Fayez Gebali

# Analysis of Computer Networks

*Second Edition*

Springer

# Analysis of Computer Networks

Fayez Gebali

# Analysis of Computer Networks

Second Edition

Springer

Fayez Gebali
Department of Electrical and Computer
Engineering
University of Victoria
Victoria, BC, Canada

ISBN 978-3-319-15656-9 ISBN 978-3-319-15657-6 (eBook)
DOI 10.1007/978-3-319-15657-6

Library of Congress Control Number: 2015934004

Springer Cham Heidelberg New York Dordrecht London


Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

# Preface

The purpose of this book is to give the reader two things, to paraphrase Mark Twain: Roots to know the basics of modeling networks and Wings to fly away and attempt modeling other proposed systems of interest.

The Internet phenomenon is affecting us all in the way we communicate, conduct business, and access information and entertainment. More unforeseen applications are still to come. All of this is due to the existence of an efficient global high-performance network that connects millions of users and moves information at a high rate with small delay.

## High-Performance Networks

A high-performance network is characterized by two performance measures **bandwidth** and **delay**. Traditional network design focused mainly on bandwidth planning; the solution to network problems was to add more bandwidth. Nowadays, we have to consider message delay particularly for delay-sensitive applications such as voice and real-time video. Both bandwidth and delay contribute to the performance of the network. Bandwidth can be easily increased by compressing the data, by using links with higher speed, or by transmitting several bits in parallel using sophisticated modulation techniques. Delay, however, is not so easily improved. It can only be reduced by the use of good scheduling protocols, very fast hardware and switching equipment throughout the network. The increasing use of optical fibers means that the transmission channel is close to ideal with extremely high bandwidth and low delay (speed of light). The areas that need optimization are the interfaces and devices that connect the different links together such as hubs, switches, routers, and bridges. The goal of this book is to explore the design and analysis techniques of these devices. There are indications, however, that the optical fiber channel is becoming less than ideal due to the increasing bit rates. Furthermore, the use of wireless mobile networking is becoming very popular. Thus new and improved techniques for transmitting across the noisy, and band-limited, channel become very essential.

The work to be done to optimize the physical level of communication is devising algorithms and hardware for adaptive data coding, modulation, and compression. Thus digital signal processing is finding an increasing and pivotal role in the area of networking and communications.

## Scope

The three main building blocks of high-performance networks are the links, the switching equipment connecting the links together, and the software employed at the nodes and switches. The purpose of this book is to provide the basic techniques for modeling and analyzing the last component: the software and protocols. For this purpose, different topics are covered in the book such as Markov chains and queuing analysis, traffic modeling, and protocol analysis.

There are many books and articles dealing with continuous-time Markov chains and queuing analysis. This is because continuous-time systems were thought to be easily modeled and analyzed. However, digital communications are discrete in nature. Luckily, discrete-time Markov chains are simple, if not even easier, to analyze. The first edition of this book in 2008 contributed to supporting this idea as can be ascertained by the nature of recent publications on the topic. The approach we chose to present Markov chains and queuing analysis is to start with explaining the basic concepts, then explain the analytic and numerical techniques that could be used to study the system. We introduce many worked examples throughout to get a feel as to how to apply discrete-time Markov chains to many communication systems.

We employ MATLAB® throughout this book for its toolboxes and its simplicity and popularity. It is widely used by engineers and engineering students. There are many equally useful mathematical packages available nowadays on many workstations and personal computers such as the Canadian Maple® and Mathematica®.

## Organization

This book covers the mathematical theory and techniques necessary for analyzing telecommunication systems. Queuing and Markov chain analyses are provided for many protocols that are used in networking. The book then discusses in detail applications of Markov chains and queuing analysis to model communications protocols. Several appendices are also provided that round up the discussion and provide a handy reference for the necessary background material.

*Chapter 1* discusses probability theory and random variables. There is a discussion of sample spaces and how to count the number of outcomes of a random experiment. Also discussed is probability density function and expectations. Important distributions are discussed since they will be used for describing traffic in our analysis. The Pareto distribution is discussed in this chapter, which is usually

not discussed in standard engineering texts on probability. Perhaps what is new in this chapter is the review of techniques for generating random numbers that obey a desired probability distribution. Inclusion of this material rounds up the chapter and helps the designer or researcher to generate the network traffic data needed to simulate a system under specified conditions.

*Chapter 2* discusses random processes and in particular Poisson and exponential processes. The chapter also discusses concepts associated with random processes such as ensemble average, time average, autocorrelation function, and cross-correlation function.

*Chapter 3* discusses discrete-time Markov chains. Techniques for constructing the state transition matrix are explored in detail as well as how the time step is determined since all discrete-time Markov chains require awareness of the time step value. The chapter discusses also transient behaviour of Markov chains and explains the various techniques for studying it such as diagonalization, expansion of the initial distribution vector, Jordan canonic form, and using the $z$-transform.

*Chapter 4* is a very useful and important chapter since it discusses Markov chains at equilibrium, or steady state. Analytic techniques for finding the equilibrium distribution vector are explained such as finding the eigenvalues and eigenvectors of the state transition matrix, solving difference equations, and the $z$-transform technique. Several numerical techniques for finding the steady-state distribution are discussed such as use of forward substitution and backward substitution, and iterative equations. The concepts of balance equations and flow balance are also explained.

*Chapter 5* discusses reducible Markov chains and explains the concept of closed and transient states. The transition matrix for a reducible Markov chain is partitioned into blocks, and the closed and transient states are related to each partitioning block. An expression is derived for the state of a Markov chain at any time instant $n$ and also at equilibrium. The chapter also discusses how a reducible Markov chain could be identified by studying its eigenvalues and eigenvectors. It is shown that the eigenvectors enable us to identify all sets of closed and transient states.

*Chapter 6* discusses periodic Markov chains. Two types of periodic Markov chains are identified and discussed separately. The eigenvalues of periodic Markov chains are discussed and related to the periodicity of the system. What is novel in this chapter is the ability to recognize and characterize a periodic Markov chain by inspection of its eigenvalues. Transient analysis of a periodic Markov chain is discussed in detail and asymptotic behaviour is analyzed.

*Chapter 7* discusses discrete-time queues and queuing analysis. Kendall's notation is explained and several discrete-time queues are analyzed such as the infinite-sized $M/M/1$ queue and the finite-sized $M/M/1/B$ queue. Equally important queues encountered in this book are also considered such as $M^m/M/1/B$ and $M/M^m/1/B$ queues. The important performance parameters considered for each queue are the throughput, delay, average queue size, loss probability, and efficiency. The chapter also discusses how to analyze networks of queues using two techniques: the flow balance approach and the merged approach.

*Chapter 8* discusses the modeling of several flow control protocols using Markov chains and queuing analysis. Three traffic management protocols are considered and analyzed for the first time: leaky bucket, token bucket, and the virtual scheduling (VS) algorithm.

*Chapter 9* discusses the modeling of several error control protocols using Markov chains and queuing analysis. Three error control techniques using automatic repeat request algorithms are considered: stop-and-wait (SW ARQ), go-back-$N$ (GBN ARQ), and selective repeat protocol (SRP ARQ).

*Chapter 10* discusses the modeling of several medium access control protocols using Markov chains and queuing analysis. Several media access protocols are discussed: IEEE Standard 802.1p (static priority), pure and slotted ALOHA, IEEE Standard 802.3 (CSMA/CD, Ethernet), and Carrier sense multiple access with collision avoidance (CSMA/CA).

*Chapter 11* discusses and analyzes several variants of the IEEE 802.11 protocol such as IEEE Standard 802.11 basic distributed coordination function for ad hoc networks, IEEE Standard 802.11 RTS/CTS protocol for wireless ad hoc networks, Enhanced Distributed Channel Access (EDCA) for ad hoc networks, and IEEE 802.11e Hybrid Coordination function Control Channel Access (HCCA) for wireless infrastructure networks.

*Chapter 12* discusses the IEEE 802.16 WiMAX and provides a simple analytical model for the Access Point (AP) and the individual users.

*Chapter 13* provides a quick review of the physical wireless channel fading phenomena and develops a Markov chain model for fading.

*Chapter 14* discusses cognitive radio (CR) and develops a model for opportunistic behaviour.

*Chapter 15* discusses realistic statistical descriptors of network traffic and their effect on packet queues or buffers. Different traffic distributions are considered such as on–off traffic, Poisson traffic, Bernoulli traffic, Self-similar traffic, and the general case of traffic that follows an arbitrary distribution. Other topics are also analyzed such as traffic destination, packet length, and transmission errors. The interarrival times for different traffic distributions are discussed in detail and realistic models are proposed.

*Chapter 16* discusses scheduling algorithms. The differences and similarities between scheduling algorithms and media access protocols are discussed. Scheduler performance measures are explained and scheduler types or classifications are explained. The concept of max–min fairness is explained since it is essential for the discussion of scheduling algorithms. Twelve scheduling algorithms are explained and analyzed: first-in/first-out (FIFO), static priority, round robin (RR), weighted round robin (WRR), processor sharing (PS), generalized processor sharing (GPS), fair queuing (FQ), packet by packet GPS (PGPS), weighted fair queuing (WFQ), frame-based fair queuing (FFQ), core-stateless fair queuing (CSFQ), and finally random early detection (RED).

*Appendix A* provides a handy reference for many formulas that are useful while modeling the different queues considered here. The reader should find this information handy since it was difficult to find all of formulas in a single source.

*Appendix B* discusses techniques for solving difference equations or recurrence relations. These recurrence relations crop up in the analysis of queues and Markov chains.

*Appendix C* discusses how the z-transform technique could be used to find a closed-form expression for the distribution vector $\mathbf{s}(n)$ at any time value through finding the z-transform of the transition matrix $\mathbf{P}$.

*Appendix D* discusses vectors and matrices. Several concepts are discussed such as matrix inverse, matrix nullspace, rank of a matrix, matrix diagonalization, and eigenvalues and eigenvectors of a matrix. Techniques for solving systems of linear equations are discussed since these systems are encountered in several places in the book. Many special matrices are discussed such as circulant matrix, diagonal matrix, echelon matrix, Hessenberg matrix, identity matrix, nonnegative matrix, orthogonal matrix, plane rotation, stochastic (Markov) matrix, substochastic matrix, and tridiagonal matrix.

## Advanced Topics

I invested special effort in making this book useful to practicing engineers and students. There are many interesting examples and models throughout the book. However, I list here some interesting topics:

- Chapter 1 discusses heavy-tailed distribution in Sect. 1.20 and generation of random numbers in Sect. 1.36.
- Chapter 3 discusses techniques for finding higher powers for Markov chain state transition matrix in Sects. 3.13 and 3.14.
- Chapter 5 discusses reducible Markov chains at steady state in Sect. 5.7 and transient analysis of reducible Markov chains in Sect. 5.6. Also there is a discussion on how to identify a reducible Markov chain by examining its eigenvalues and eigenvectors.
- Chapter 6 discusses transient analysis of periodic Markov chains in Sect. 6.14 and asymptotic behavior of periodic Markov chains in Sect. 6.15. Also there is a discussion on how to identify a periodic Markov chain and how to determine its period by examining its eigenvalues.
- Chapter 7 discusses developing performance metrics for the major queue types.
- Chapter 8 discusses how to model three flow control protocols dealing with traffic management.
- Chapter 9 discusses how to model three flow control protocols dealing with error control.
- Chapter 10 discusses how to model three flow control protocols dealing with medium access control.
- Chapter 15 discusses developing realistic models for source traffic using Poisson description (Sect. 15.3.2), Bernoulli (Sect. 15.4.2), and Pareto traffic (Sect. 15.8). There is also a discussion on Packet destination and length modeling.

## Web Resources

A website is provided, *http://www.ece.uvic.ca/~fayez/Book*, that will contain information about the textbook and any related web resources.

## Errors

This book covers a wide range of topics related to communication networks and provides an extensive set of analyses and worked examples. It is "highly probable" that it contains errors and omissions. Other researchers and/or practicing engineers might have other ideas about the content and organization of this book. We welcome receiving any constructive comments and suggestions for inclusion in the next edition. If you find any errors, we would appreciate hearing from you. We also welcome ideas for examples and problems (along with their solutions if possible) to include in the next edition with proper citation.

You can send your comments and bug reports electronically to fayez@uvic.ca or you can fax or mail the information to:

Dr. Fayez Gebali
Elec. and Comp. Eng. Dept.
University of Victoria Victoria, BC, Canada V8W 2Y2
Tel: (250)721-6509
Fax: (250) 721-6052.

# Acknowledgments

I was motivated to write this book as a result of reading two very good works. One book is of course the famous *Queueing Systems – Volume 1: Theory* by Leonard Kleinrock. The other book is *Communication and Computer Networks: Modeling with Discrete-Time Queues* by Michael E. Woodward. The title of the last book sums it all. Discrete-time queuing systems proved to be very useful in dealing with communications systems.

I felt that there were more practical topics and important definitions that either were not discussed or were a bit vague. This book provides a minor contribution to this area by summarizing in book form the techniques I advise my students to use while doing their graduate degrees under my supervision. I would like to thank Dr. Mohamed Watheq El-Kharashi for suggesting that I put everything in book format. I also value his excellent suggestions and help. His vision, suggestions, and his time were absolutely helpful and appreciated.

I would like to thank Dr. W.-S. Lu of the Electrical and Computer Engineering Department of the University of Victoria for his extremely valuable help and insight in all the matrix topics covered in this book. I would like also to thank Dr. G.A. Shoja and Dr. D. Olesky of the Computer Science Department at the University of Victoria for their suggestions, the references they provided, and their encouragement during preparing the material for this book. The constant help and feedback of some of my past and current graduate students was particularly helpful. In particular, I note Mr. Maher Fahmi, Dr. A. Sabba, and Dr. Esam Abdel-Raheem.

Gratitude must go to PMC-Sierra, Inc., of Vancouver for the meetings and consultations I regularly had with their directors, managers, and engineering staff. The financial support of PMC-Sierra throughout the years has contributed enormously to my research program. Without that support I would not have been able to build my research group or acquire the necessary equipment.

The MathWorks, Inc., has provided me with their MATLAB, Simulink, and Stateflow software packages. This is of great help in developing the material and presentations for the examples and problems of this book. For that I am obliged.

Thanks must go to my children Michael Monir, Tarek Joseph, Aleya Lee, and Manel Alia for being such responsible and outstanding young men and women.

Victoria, BC, Canada Fayez Gebali

# Contents

# About the Author

**Dr. Fayez Gebali** is currently the Chair of the Electrical and Computer Engineering Department at the University of Victoria. Previously he has been the Associate Dean (undergraduate programs) at the Faculty of Engineering, University of Victoria. He is also a professor at the Electrical and Computer Engineering Department. Before that he was a Research Associate with Dr. M.I. Elmasry in the Department of Electrical Engineering at the University of Waterloo. He was also a Research Associate with Dr. K. Colbow in the Physics Department of Simon Fraser University.

Dr. Gebali's research interests include parallel algorithms, hardware security, and computer arithmetic. He has published over 280 articles in refereed journals and conferences in his areas of expertise.

Dr. Gebali received his B.Sc. in Electrical Engineering (first-class honours) from Cairo University, his B.Sc. (first-class honours) in Applied Mathematics from Ain Shams University, and his Ph.D. degree in Electrical Engineering from the University of British Columbia. He is a life senior member of the IEEE, a member of the Society of Professional Engineers and Geoscientists of British Columbia, and a member of the Egyptian Professional Engineers of Egypt.

# Chapter 1
# Probability

## 1.1 Introduction

The goal of this chapter is to provide a review of the principles of probability, random variables, and distributions. Probability is associated with conducting a **random experiment** or **trial** and checking the resulting **outcome**.

**Definition 1.1 (Outcome).** An *outcome* is any possible observation of a random experiment.

For example, the random experiment might be flipping a coin and the outcome would be heads or tails depending whether the coin lands face up or down. The possible outcomes of a random experiment could be discrete or continuous. An example of a random experiment with discrete outcomes is rolling a die since the possible outcomes would be the numbers 1, 2, $\cdots$, 6. An example of a random experiment with continuous outcomes is spinning a pointer and measuring its angle relative to some reference direction. The angle we measure could be anything between $0^\circ$ and $360^\circ$.

**Definition 1.2 (Sample Space).** The collection of all possible, mutually exclusive outcomes is called the *sample space* $S$.

For the random experiment of rolling a die, the sample space will be the set:

$$S = \{1, 2, 3, 4, 5, 6\}$$

This sample space is discrete since the possible outcomes were discrete. For the example of spinning a pointer, the sample space is specified by the equation:

$$S = \{\theta \mid 0^\circ \leq \theta < 360^\circ\}$$

**Definition 1.3 (Event).** An *event* is a set of outcomes sharing a common characteristic.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_1

In that sense, an event could correspond to one or several outcomes of an experiment. An event could be thought of as a *subset* of the sample space $S$. On the other hand, an outcome is an *element* of the subset space.

**Definition 1.4 (Event Space).** The *event space* is the collection of all possible, mutually exclusive events of a random experiment.

In that sense, the event space is the partitioning of $S$ into subsets that do not share their elements.

For the case of die rolling experiment, we might be interested in the event $E$ that the outcome is an even number—Thus we can specify $E$: the number is even or $\overline{E}$: the number is odd. In that case there are six outcomes and two events. The event space will be composed of two events:

$$S = \{E, \overline{E}\}$$

Another event could be that the outcome is greater than 2, say. In that case there are two events again for the experiment—$E$: the number $= 1$ or 2 and $\overline{E}$: the number $> 2$. The event space will also consist of two events.

## 1.2 Applying Set Theory to Probability

We saw above that sets are used to describe aspects of random experiments such as sample space, outcomes, and events. Table 1.1 shows the correspondence between set theory terminology and probability definitions.

Assume $A$ and $B$ are two events in a random experiment. These two events might be graphically shown using the Venn diagram of Fig. 1.1. The event defined as any outcome that belongs to either $A$ or $B$ is called the **union** of $A$ and $B$ and is represented by the expression:

$$A \cup B$$

Figure 1.1a shows the union operation as the shaded area.

The event defined as any outcome that belongs to both $A$ and $B$ is called the **intersection** of $A$ and $B$ and is represented by the expression:

**Table 1.1** Correspondence between set theory and probability definitions [1]

| Probability | Set theory |
|---|---|
| Outcome | Element of a set |
| Sample space $S$ | Universal set $U$ |
| Event | Subset |
| Impossible event | Null set $\emptyset$ |

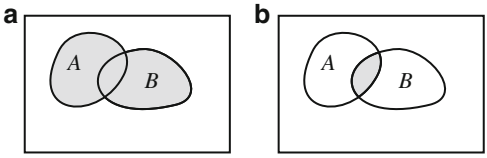



**Fig. 1.1** Venn diagram for two events $A$ and $B$ in a sample space $S$. (**a**) The union operation $A \cup B$. (**b**) The intersection operation $A \cap B$

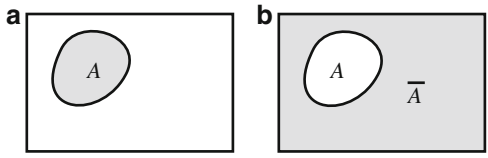



**Fig. 1.2** Event $A$ and its complementary event $\overline{A}$

$$A \cap B$$

Figure 1.1b shows the intersection operation as the shaded area.

**Definition 1.5 (Complementary Event).** Given an event $A$, the *complementary event* $\overline{A}$ is the set of all outcomes that do not belong to the set $A$.

Figure 1.2 shows that the universal set $U$ is partitioned into the two sets $A$ and $\overline{A}$. These two sets are not overlapping in the sense that there is not a single outcome that belongs to both $A$ and $\overline{A}$ simultaneously.

We can write the following equations describing operations on events:

$$\overline{A} = U - A$$
$$A \cup \overline{A} = U$$
$$A \cap \overline{A} = \emptyset$$

De Morgan's law for sets applies also for events and we can write:

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

*Example 1.1.* Let $U = \{a, b, c, d, e, f, g, h, i, j, k\}$, $A = \{a, c, e, h, j\}$, and $B = \{c, d, e, f, k\}$. Find the events:

1. $A \cup B$
2. $A \cap B$

Prove also that $\overline{A \cap B} = \overline{A} \cup \overline{B}$

$$A \cup B = \{a, c, d, e, f, h, j, k\}$$
$$A \cap B = \{c, e\}$$

We also have:

$$\overline{A \cap B} = \{a,\ b,\ d,\ f,\ g,\ h,\ i,\ j,\ k\}$$
$$\overline{A} = \{b,\ d,\ f,\ g,\ i,\ k\}$$
$$\overline{B} = \{a,\ b,\ g,\ h,\ i,\ j\}$$
$$\overline{A} \cup \overline{B} = \{a,\ b,\ d,\ f,\ g,\ h,\ i,\ j,\ k\}$$

Hence De Morgan's law is proved by direct counting. ■

**Definition 1.6 (Mutually Exclusive Events).** Events $A$ and $B$ are said to be **mutually exclusive** or **disjoint events** if they have no elements in common.

From that definition, we can write:

$$A \cap B = \phi \tag{1.1}$$

## 1.3 Counting Sample Space Points

In many discussions of probability, we need to determine the number of possible outcomes of a given experiment or trial. The multiplication principle, permutations, and combinations, to be discussed in the following sections, will prove useful.

## 1.4 The Multiplication Principle

The fundamental principle of counting is useful in finding the number of points in the sample space. Suppose that the number of outcomes for doing experiment $E_1$ is $x$ and the number of outcomes for doing experiment $E_2$ is $y$. Then the number of outcomes for doing experiment $E_1$ followed by experiment $E_2$ is the product $x\ y$.

*Example 1.2.* Assume there are 3 different ways for a data packet to travel through network $A$ and 15 different ways to travel through network $B$. Use the multiplication principle to find the number of ways the packet could travel through network $A$ followed by network $B$.

The multiplication principle states that there are $3 \times 15$ or 45 ways for the packet to travel through network $A$ followed by network $B$. ■

We can generalize the multiplication principle as follows. Suppose that $N_1$ is the number of outcomes for doing experiment $E_1$, and $N_2$ is the number of outcomes for doing experiment $E_2, \cdots$, and $N_n$ is the number of outcomes for doing experiment $E_n$. The number of outcomes of performing experiments $E_1,\ E_2, \cdots,\ E_n$ is given by the product:

$$N = N_1\ N_2\ \cdots\ N_n$$

*Example 1.3.* A die is thrown three times and the sequence of numbers is recorded. Determine the number of 3-digit sequences that could result.

We perform three experiments where each one is the act of throwing the die. One possible outcome would be the number sequence: 312; which corresponds to obtaining 3 on the first throw, 1 on the second, and 2 on the third. The number of outcomes of each experiment is 6. Therefore, the total number of outcomes is:

$$N = 6 \times 6 \times 6 = 216$$ ■

*Example 1.4.* In time division multiplexing (also known as synchronous transmission mode) each slot in a frame is reserved to a certain user. That slot could be occupied or empty when the user is busy or idle, respectively. Assuming the frame consists of 10 slots, how many slot occupancy patterns can be received in a single frame?

Each time slot can be treated as an experiment with only two outcomes, busy or idle. The experiment is repeated 10 times to form the frame. The total number of possible outcomes is:

$$N = 2^{10} = 1024$$ ■

## 1.5 Permutations

Permutations arise when we are given $n$ objects and we want to arrange them according to a certain order. In other words, we arrange the objects by randomly picking one, then the next, and so on. We have $n$ choices for picking the first item, $n - 1$ choices of picking the second item, and so on.

### *1.5.1 n Distinct Objects Taken n at a Time*

The number of permutations of $n$ distinct objects taken $n$ at a time is denoted by $P(n, n)$ and is given by:

$$P(n, n) = n! = n \times (n - 1) \times (n - 2) \times \cdots \times 3 \times 2 \times 1$$

The function $n!$ is called *Factorial-n* and can be obtained using the MATLAB function `factorial(n)`.

*Example 1.5.* Packets are sent through the Internet in sequence; however, they are received out of sequence since each packet could be sent through a different route. How many ways can a 5-packet sequence be received?

If we number our packets as 1, 2, 3, 4, and 5, then one possible sequence of received packets could be: 12543 when packet 1 arrives first followed by packet 2 then packet 5, and so on. The number of possible received packet sequences is given by:

$$P(5,5) = 5! = 120$$ ■

### 1.5.2 n Distinct Objects Taken k at a Time

A different situation arises when we have $n$ distinct objects but we only pick $k$ objects to arrange. In that case, the number of permutations of $n$ distinct objects taken $k$ at a time is given by:

$$\begin{aligned} P(n,k) &= n \times (n-1) \times \cdots \times (n-k+1) \\ &= \frac{n!}{(n-k)!} \end{aligned} \tag{1.2}$$

*Example 1.6.* Assume that 10 packets are sent in sequence, but they are out of sequence when they are received. If we observe a three-packet sequence only, how many 3-packet sequences could we observe?

A possible observed packet arrival sequence could be 295. Another might be 024, and so on. We have $n = 10$ and $k = 3$:

$$P(10,3) = \frac{10!}{7!} = 720$$ ■

## 1.6 Permutations of Objects in Groups

Now, assume we have $n$ objects in which $n_1$ objects are alike, $n_2$ objects are alike, $\cdots$, and $n_k$ objects are alike such that:

$$n = \sum_{i=1}^{k} n_i \tag{1.3}$$

Here we classify the objects not by their individual labels but by the group in which they belong. An output sequence will be distinguishable from another if the objects picked happen to belong to different groups. As a simple example, suppose we have 20 balls that could be colored red, green, or blue. We are now interested not in picking a particular ball, but in picking *a* ball of a certain color.

In that case, the number of permutations of these $n$ objects taken $n$ at a time is given by:

$$x = \frac{n!}{n_1!\, n_2! \cdots n_k!} \tag{1.4}$$

This number is smaller than $P(n, n)$ since several of the objects are alike and this reduces the number of **distinguishable** combinations.

*Example 1.7.* Packets arriving at a terminal could be one of three possible service classes: class A, class B, or class C. Assume that we received 10 packets and we found out that there were 2 packets in class A, 5 in class B, and 3 in class C. How many possible service class arrival order could we have received?

We are not interested here in the sequence of received packets. Instead, we are interested only in the arrival order of the service classes.

We have $n_1 = 2$, $n_2 = 5$, and $n_3 = 3$ such that $n = 10$. The number of service class patterns is:

$$x = \frac{10!}{2!\, 5!\, 3!} = 2,520$$

In other words, there are 10 possibilities for receiving 10 packets such that exactly two of them belonged to class A, five belonged to class B, and three belonged to class C. ■

*Example 1.8.* In time division multiplexing each time slot in a frame is reserved to a certain user. That time slot could be occupied or empty when the user is busy or idle, respectively. If we know that each frame contains ten time slots and four users are active and six are idle, how many possible active slot patterns could have been received?

We are interested here in finding the different ways we could have received 4 active slots out of 10 possible slots. Thus we "color" our slots as active or idle without regard to their location in the frame.

We have $n_1 = 4$ and $n_2 = 6$ such that $n = 10$:

$$x = \frac{10!}{4!\, 6!} = 210$$

■

*Example 1.9.* A bucket contains 10 marbles. There are 5 red marbles, 2 green marbles, and 3 blue marbles. How many different color permutations could result if we arranged the marbles in one straight line?

We have $n_1 = 5$, $n_2 = 2$, and $n_3 = 3$ such that $n = 10$. The number of different color permutations is:

$$x = \frac{10!}{5!\, 2!\, 3!} = 2,520$$

■

## 1.7 Combinations

The above permutations took the **order** of choosing the objects into consideration. If the order of choosing the objects is not taken into consideration, then **combinations** are obtained.

The number of combinations of $n$ objects taken $k$ at a time is called the **binomial coefficient** and is given by:

$$C(n,k) = \binom{n}{k} = \frac{n!}{k!\,(n-k)!} \tag{1.5}$$

MATLAB has the function `nchoosek(n,k)` for evaluating the above equation where $0 \leq k \leq n$.

*Example 1.10.* Assume 10 packets are received with 2 packets in error. How many combinations are there for this situation?

We have $n = 10$ and $k = 2$:

$$C(10,2) = \binom{10}{2} = \frac{10!}{2!\,8!} = 45$$

■

*Example 1.11.* Assume 50 packets are received but 4 packets are received out of sequence. How many combinations are there for this situation?

We have $n = 50$ and $k = 4$:

$$C(50,4) = \binom{50}{4} = \frac{50!}{4!\,46!} = 230,300$$

■

## 1.8 Probability

We define probability using the **relative-frequency approach**. Suppose we perform an experiment like the tossing of a coin for $N$ times. We define event $A$ is when the coin lands head up. Define $N_A$ as the number of times that event $A$ occurs when the coin tossing experiment is repeated $N$ times. Then the probability that we will get a head when the coin is tossed is given by:

$$p(A) = \lim_{N \to \infty} \frac{N_A}{N} \tag{1.6}$$

This equation defines the relative frequency that event $A$ happens.

## 1.9 Axioms of Probability

We defined our sample space $S$ as the set of all possible outcomes of an experiment. An impossible outcome defines the empty set or null event $\emptyset$. Based on this we can state four basic axioms for the probability:

1. The probability $p(A)$ of an event $A$ is a nonnegative fraction in the range:

$$0 \leq p(A) \leq 1$$

   This can be deduced from the basic definition of probability in (1.6).
2. The probability of the null event $\emptyset$ is zero $p(\emptyset) = 0$.
3. The probability of all possible events $S$ is unity $p(S) = 1$.
4. If $A$ and $B$ are **mutually exclusive** events (cannot happen at the same time), then the probability that event $A$ **or** event $B$ occurs is given by:

$$p(A \cup B) = p(A) + p(B) \tag{1.7}$$

Event $E$ and its complement $E^c$ are mutually exclusive. Applying the above axioms of probability we can write:

$$p(E) + p(E^c) = 1 \tag{1.8}$$

$$p(E \cap E^c) = 0 \tag{1.9}$$

## 1.10 Other Probability Relationships

If $A$ and $B$ are two events (they need not be mutually exclusive), then the probability that event $A$ **or** event $B$ occurring is given by:

$$p(A \cup B) = p(A) + p(B) - p(A \cap B) \tag{1.10}$$

The probability that event $A$ occurs **given that** event $B$ occurred is denoted by $P(A|B)$ and is sometimes referred to as the probability of $A$ conditioned by $B$. This is given by:

$$p(A|B) = \frac{p(A \cap B)}{p(B)} \tag{1.11}$$

Now, if $A$ and $B$ are two **independent** events, then we can write $p(A|B) = p(A)$ because the probability of event $A$ taking place will not change whether event $B$ occurs or not. From the above equation we can now write the probability that event $A$ **and** event $B$ occurs is given by:

$$p(A \cap B) = p(A) \times p(B) \tag{1.12}$$

provided that the two events are **independent**.

The probability of the complement of an event is given by:

$$p(A^c) = 1 - p(A) \tag{1.13}$$

## 1.11 Random Variables

Many systems based on random phenomena are best studied using the concept of random variables. A random variable allows us to employ mathematical and numerical techniques to study the phenomenon of interest. For example, measuring the length of packets arriving at random at the input of a switch produces as outcome a number that corresponds to the length of that packet.

According to references [2–5], a **random variable** is simply a numerical description of the outcome of a random experiment. We are free to choose the **function** that maps or assigns a numerical value to each outcome depending on the situation at hand. Later we shall see that the choice of this function is rather obvious in most situations. Figure 1.3 graphically shows the steps leading to assigning a numerical value to the outcome of a random experiment. First we run the experiment then we observe the resulting outcome. Each outcome is assigned a numerical value.

Assigning a numerical value to the outcome of a random experiment allows us to develop uniform analysis for many types of experiments independent of the nature of their specific outcomes [2].

We denote a random variable by a capital letter (the name of the function) and any particular value of the random variable is denoted by a lowercase letter (the value of the function).

Examples of random variables, and their numerical values, could be:

1. Number of arriving packets at a given time instance is an example of a **discrete random variable** $N$ with possible values $n = 0, 1, 2, \cdots$.
2. Tossing a coin and assigning 0 when a tail is obtained and 1 when a head is obtained is an example of a discrete random variable $X$ with values $x \in \{0, 1\}$.
3. The weight of a car in kilograms is an example of a **continuous random variable** $W$ with values in the range $1,000 \leq w \leq 2,000$ kg typically.
4. The temperature of a day at noon is an example of random variable $T$. This random variable could be discrete or continuous depending on the type of the thermometer (analog or digital).

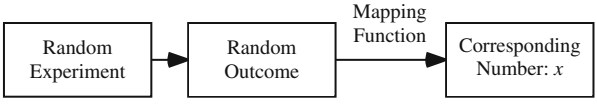



**Fig. 1.3** The steps leading to assigning a numerical value to the outcome of a random experiment

5. The atmospheric pressure at a given location is an example of a random variable $P$. This random variable could be discrete or continuous depending on the accuracy of the barometer (analog or digital).

## 1.12 Cumulative Distribution Function

The cumulative distribution function (CDF) for a random variable $X$ is denoted by $F_X(x)$ and is defined as the **probability** that the random variable is less than or equal to $x$. Thus the event of interest is $X \leq x$ and we can write:

$$F_X(x) = p(X \leq x) \tag{1.14}$$

The subscript $X$ denotes the random variable associated with the function while the argument $x$ denotes a numerical value. For simplicity we shall drop the subscript and write $F(x)$ when we are dealing with a single random variable and there is no chance of confusion. Because $F(x)$ is a probability, it must have the same properties of probabilities. In addition, $F(x)$ has other properties as shown below:

$$F(-\infty) = 0 \tag{1.15}$$

$$F(\infty) = 1 \tag{1.16}$$

$$0 \leq F(x) \leq 1 \tag{1.17}$$

$$F(x_1) \leq F(x_2) \qquad \text{when } x_1 \leq x_2 \tag{1.18}$$

$$p(x_1 < X \leq x_2) = F(x_2) - F(x_1) \tag{1.19}$$

The CDF is a monotonically increasing function of $x$. From the last equation, the probability that $x$ lies in the region $x_0 < x \leq x_0 + \epsilon$ (where $\epsilon$ is arbitrarily small) is given by:

$$p(x_0 < X \leq x_0 + \epsilon) = F(x_0 + \epsilon) - F(x_0) \tag{1.20}$$

**Thus the amount of jump in CDF at $x = x_0$ is the probability that $x = x_0$.**

*Example 1.12.* Consider the random experiment of spinning a pointer around a circle and measuring the angle it makes when it stops. Plot the CDF $F_\Theta(\theta)$.

Obviously the random variable $\Theta$ is continuous since the pointer could point at any angle. The range of values for $\theta$ is between $0^\circ$ and $360^\circ$. Thus the function $F_\Theta(\theta)$ has the following extreme values:

$$F_\Theta\left(-0^\circ\right) = p\left(\theta \leq -0^\circ\right) = 0$$

$$F_\Theta\left(360^\circ\right) = p\left(\theta \leq 360^\circ\right) = 1$$

There is no preference for the pointer to settle at any angle in particular and the CDF will have the distribution shown in Fig. 1.4. ■

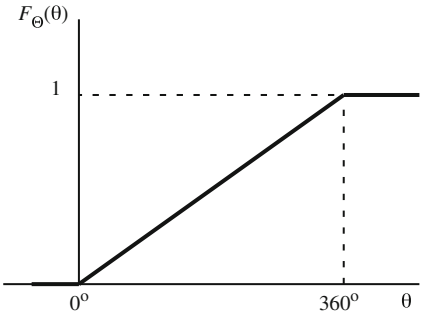



**Fig. 1.4** Cumulative distribution function for a continuous random variable

### *1.12.1 CDF in the Discrete Case*

For the case of a discrete random variable we make use of the CDF property in (1.20). The CDF for a discrete random variable will be a staircase as illustrated in the following example.

*Example 1.13.* Consider again the case of the spinning pointer experiment but define the discrete random variable $Q$ which identifies the quadrant in which the pointer rests in. The quadrants are assigned the numerical values 1, 2, 3, and 4. Thus the random variable $Q$ will have the values $q = 1, 2, 3,$ or 4.

Since the pointer has equal probability of stopping in any quadrant, we can write:

$$p(q = 1) = \frac{1}{4}$$

$$p(q = 2) = \frac{1}{4}$$

$$p(q = 3) = \frac{1}{4}$$

$$p(q = 4) = \frac{1}{4}$$

The CDF for this experiment is shown in Fig. 1.5. ■

## 1.13 Probability Density Function

The probability density function (pdf) for a **continuous random variable** $X$ is denoted by $f_X(x)$ and is defined as the derivative of $F_X(x)$:

$$f_X(x) = \frac{dF_X(x)}{dx} \tag{1.21}$$

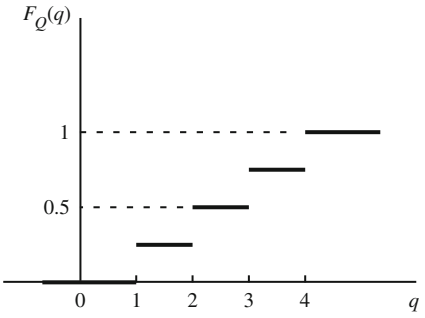



**Fig. 1.5** Cumulative distribution function for a discrete random variable

Because $F_X(x)$ is a monotonically increasing function of $x$, we conclude that $f_X(x)$ will never be negative. It can, however, be zero or even bigger than 1.

We will follow our simplifying convention of dropping the subscript when there is no chance of confusion and write the pdf as $f(x)$ instead of $f_X(x)$. Integrating the above equation we obtain:

$$\int_{x_1}^{x_2} f(x)\,dx = F(x_2) - F(x_1) \tag{1.22}$$

Thus we can write:

$$p(x_1 < X \leq x_2) = \int_{x_1}^{x_2} f(x)\,dx \tag{1.23}$$

**The area under the pdf curve is the probability** $p(x_1 < X \leq x_2)$

$f(x)$ has the following properties:

$$f(x) \geq 0 \qquad \text{for all } x \tag{1.24}$$

$$\int_{-\infty}^{\infty} f(x)\,dx = 1 \tag{1.25}$$

$$\int_{-\infty}^{x} f(y)\,dy = F(x) \tag{1.26}$$

$$\int_{x_1}^{x_2} f(x)\,dx = p(x_1 < X \leq x_2) \tag{1.27}$$

$$f(x)\,dx = p(x < X \leq x + dx) \tag{1.28}$$

## 1.14 Probability Mass Function

For the case of a **discrete random variable** the cdf is discontinuous in the shape of a staircase. Therefore, its slope will be zero everywhere except at the discontinuities where it will be infinite.

The pdf in the discrete case is called the **Probability Mass Function** (pmf) [1]. The pmf is defined as the probability that the random variable $X$ has the value $x$ and is denoted by $p_X(x)$. We can write:

$$p_X(x) \equiv p(X = x) \tag{1.29}$$

where the expression on the right-hand side indicates the probability that the random variable $X$ has the value $x$.

We will follow our simplifying convention of dropping the subscript when there is no chance of confusion and write the pmf as $p(x)$ instead of $p_X(x)$. $p_X(x)$ has the following properties:

$$p_X(x) \geq 0 \qquad \text{for all } x \tag{1.30}$$

$$\sum_x p_X(x) = 1 \tag{1.31}$$

*Example 1.14.* The pointer spinning experiment was considered for the continuous case (Example 1.12) and the discrete case (Example 1.13), plot the pdf for the continuous random variable $\Theta$ and the corresponding pmf for the discrete random variable $Q$.

Figure 1.6a shows the pdf for the continuous case where the random variable $\Theta$ measures the angle of the pointer. Figure 1.6b shows the pmf for the discrete case where the random variable $Q$ measures the quadrant where the pointer is located. ■

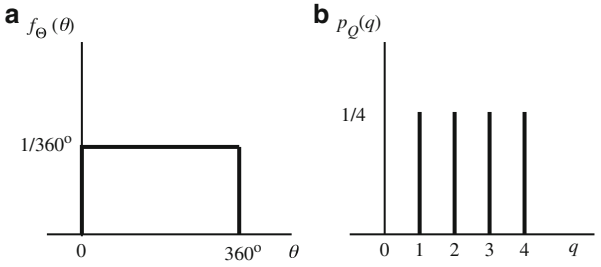



**Fig. 1.6** Continuous and discrete random variables. (**a**) pdf for the continuous case. (**b**) pmf for the discrete case

## 1.15 Expected Value and Variance

The pdf and pmf we obtained above help us find the expected value $E[X]$ of a random variable $X$. For the continuous case, the expected value is given by:

$$E[X] = \int_{-\infty}^{\infty} x \ f(x) \, dx \tag{1.32}$$

For the discrete case, the expected value is given by the weighted sum:

$$E[X] = \sum_{i} x_i \ p(x_i) \tag{1.33}$$

The expectation is sometimes referred to as the first moment of the random variable. Sometimes $\mu$ is used as another symbol for the expected value:

$$\mu = E[X]$$

The **mean** ($m$) of a set of random variable samples is defined as:

$$m = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{1.34}$$

The mean is not exactly equal to the expected value $\mu$ since $m$ changes its value depending on how many samples we take. However, as $n \to \infty$, the two quantities become equal [1].

Higher moments are also useful and we define the **variance**, or second central moment, of the random variable as:

$$\sigma^2 = E\left[(X - \mu)^2\right] \tag{1.35}$$

The variance describes how much of the mass of the distribution is close to the expected value. A small value for $\sigma^2$ indicates that most of the random variable values lie close to the expected value $\mu$. In other words, small variance means that the pdf is large only in regions close to the expected value $\mu$. For an archery target practice experiment, this might mean that most of the arrows were clustered together and landed very close at some spot on the target (not necessarily dead center).

Conversely, a large variance means that the pdf is large for values of $X$ far away from $\mu$. Again for the archery experiment, this means that most of the arrows were not clustered together and landed at different spots on the target.

The **standard deviation** $\sigma$ is simply the square root of the variance.

*Example 1.15.* Assume a random variable $A$ from a binary random experiment in which only two events result. $A$ has the two values $a$ and 0. The probability that the value $a$ is obtained is $p$ and the probability that the value 0 is obtained is $q = 1 - p$. Find the expected value of $A$.

This is a discrete random variable and the pmf for $A$ is:

$$p(a) = \begin{cases} q & \text{when } A = 0 \\ p & \text{when } A = a \end{cases} \tag{1.36}$$

The expected value is obtained from (1.33) as:

$$E[A] = q \times 0 + p \times a = p\, a \tag{1.37}$$

Notice that the expected value will be between 0 and $a$ since $p$ is a positive fraction. ■

## 1.16 Common Continuous RV Distributions

We discuss in the following sections some continuous random variables that are useful for network simulations. Discussion of common discrete random variables is found in later sections.

## 1.17 Continuous Uniform (Flat) Distribution

The uniform random variable, or uniform distribution, usually arises in physical situations where there is no preferred value for the random variable. For example, the value of a signal during analog-to-digital conversion could lie anywhere within each quantization level. This distribution is also useful in our studies because it is often used to obtain random numbers that obey the more sophisticated distributions to be discussed below. These random numbers are then considered to be the "traffic" generated at the inputs of our communication networks.

A uniform distribution is characterized by a random variable that spans the range $a$ to $b$ such that $a < b$. $f(x)$ can be written as:

$$f(x) = \begin{cases} 1/(b-a) & a \le x < b \\ 0 & \text{otherwise} \end{cases} \tag{1.38}$$

and the corresponding cdf is given by:

$$F(x) = \begin{cases} 0 & x < a \\ (x-a)/(b-a) & a \le x < b \\ 1 & x \ge b \end{cases} \tag{1.39}$$

Typically $a = 0$ and $b = 1$. Figure 1.7a shows the pdf for the uniform distribution and Fig. 1.7b shows the corresponding cdf. The mean and variance for $X$ are:

$$\mu = \frac{b+a}{2} \tag{1.40}$$

$$\sigma^2 = \frac{(b-a)^2}{12} \tag{1.41}$$

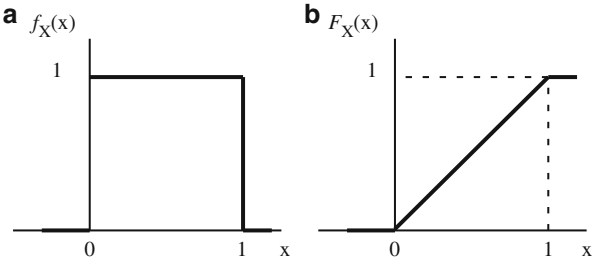



**Fig. 1.7** The uniform distribution for a continuous random variable. (**a**) The pdf; and (**b**) is the corresponding cdf

The following MATLAB code generates and plots a random variable having a uniform distribution in the range $0 \leq x < 1$.

```
%uniform.m
n=1000 % number of samples is 1000
x=rand(1,n)
subplot(1,2,1)
plot(x,'k')
box off, axis square
xlabel('Sample index'), ylabel('Random number value')
subplot(1,2,2)
hist(x)
title('pdf plot')
box off, axis square
xlabel('Bins'), ylabel('Number of samples)
```

Figure 1.8 shows the result of running the code. The figure on the left shows the samples and the figure on the right shows the histogram of the random variable. Notice that the distribution of the samples in the different bins is almost equal. If we chose the number of samples to be larger than 1,000, the histogram would show more equal distribution among the bins.

## 1.18 Gaussian Distribution

This distribution arises in many random variables used in electrical engineering such as the noise in a wireless channel. The Gaussian distribution applies for the case of a continuous random variable $X$ that is allowed to have the values ranging from $-\infty$ to $+\infty$. The pdf for this distribution is given by:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)} \tag{1.42}$$

where $\mu$ is the mean and $\sigma$ is the standard deviation of the distribution.

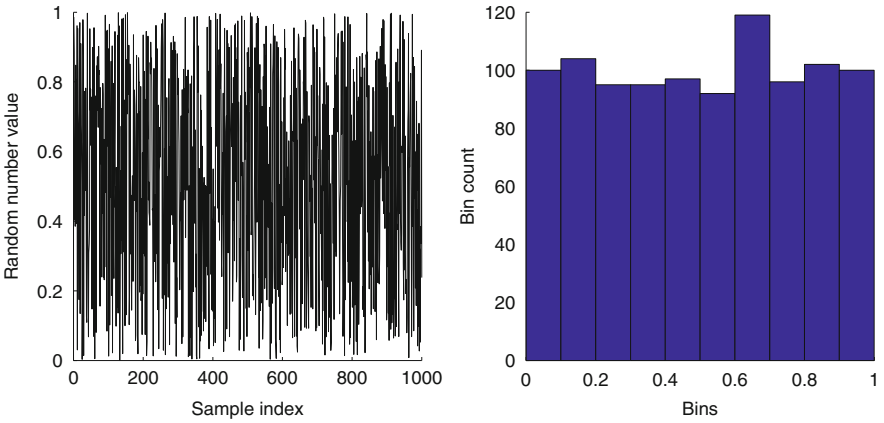



**Fig. 1.8** One thousand samples of a random variable having the uniform distribution in the range 0 to 1 (*left*). Histogram for the samples showing a uniform distribution (*right*)

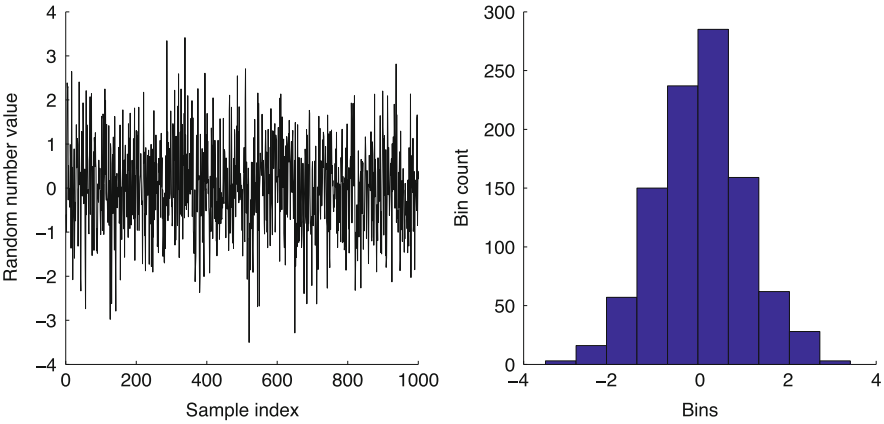



**Fig. 1.9** Random numbers generated using the Gaussian distribution with zero mean and unity variance. Figure on the left shows the random samples and figure on the right shows their histogram

The cdf for this distribution is given by:

$$F(x) = \int_{-\infty}^{x} f\,(y)\;dy \tag{1.43}$$

There is no closed-form formula for the cdf associated with the Gaussian distribution but that function is tabulated in many textbooks on statistics.

The **standard** or **normal** random variable is a Gaussian RV with $\mu = 0$ and $\sigma = 1$ [1]. Figure 1.9 shows the output of a Gaussian random variable with zero mean and unity variance using the `randn` function of MATLAB. 1,000 samples were generated in this experiment.

## 1.19 Exponential Distribution

The exponential distribution applies for the case of a continuous random variable $X$ that is allowed to have the values ranging from 0 to $+\infty$. The pdf for this distribution is given by:

$$f(x) = be^{-b\,x} \qquad x \geq 0 \tag{1.44}$$

where $b > 0$.

The corresponding cdf is given by:

$$F(x) = 1 - e^{-b\,x} \tag{1.45}$$

The mean and variance for $X$ are:

$$\mu = \frac{1}{b} \tag{1.46}$$

$$\sigma^2 = \frac{1}{b^2} \tag{1.47}$$

A famous example of exponential RV is the radioactive decay where we have:

$$f(t) = \lambda e^{-\lambda t} \qquad t \geq 0 \tag{1.48}$$

Here $\lambda$ is the rate of decay of an element. In that case $1/\lambda$ is called the *lifetime* when the radioactive material decreases by the ratio $1/e$.

## 1.20 Pareto Distribution

The Poisson and binomial distributions have been traditionally employed to model traffic arrival at networks. Recent work has shown that such models may be inadequate because the traffic may exhibit periods of high data rates even when the average data rate is low. This type of traffic is described as being self-similar (fractal) [6–8]. Self-similar traffic has distributions with very high variance. Sometimes such distributions are described as being **heavy-tailed** since the pdf has large values for $x$ far away from the mean $\mu$. This type of distribution might then prove more accurate in describing the pdf for the rate of data produced by a bursty source.

The Pareto distribution could be made to be a heavy-tailed distribution by proper choice of its parameters. The Pareto distribution is described by the pdf:

$$f\,(x) = \frac{b\,a^b}{x^{b+1}} \qquad \text{with } a \leq x < \infty \tag{1.49}$$

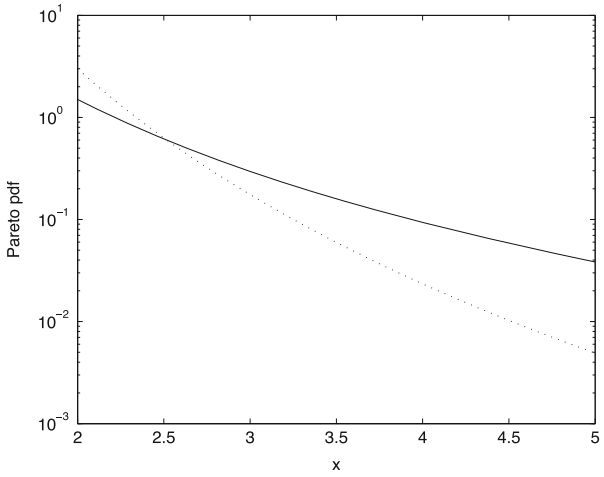



**Fig. 1.10** Pareto pdf distribution for the case when $a = 2$ and $b = 3$ (*solid line*) and $b = 5$ (*dashed line*)

where $a$ is the position parameter and $b > 0$ is the shape parameter. Figure 1.10 shows the pdf distribution for the case when $a = 2$ and $b = 3$ (solid line) and $b = 5$ (dashed line). For the smaller value of shape parameter $b$ the pdf becomes flatter and has higher values at larger values of $x$. This results in larger variance for $X$.

The corresponding cdf is:

$$F(x) = 1 - \left(\frac{a}{x}\right)^b \tag{1.50}$$

The mean and variance for $X$ are:

$$\mu = \frac{ba}{b-1} \tag{1.51}$$

$$\sigma^2 = \frac{b\,a^2}{(b-1)^2\,(b-2)} \tag{1.52}$$

The mean is always positive as long as $b > 1$. The variance is meaningful only when $b > 2$.

From (1.50) we can write:

$$\begin{aligned} p(X > x) &= 1 - p(X \leq x) \\ &= \left(\frac{a}{x}\right)^b \end{aligned} \tag{1.53}$$

which means that the probability that the random variable has a value greater than $x$ decreases geometrically [9].

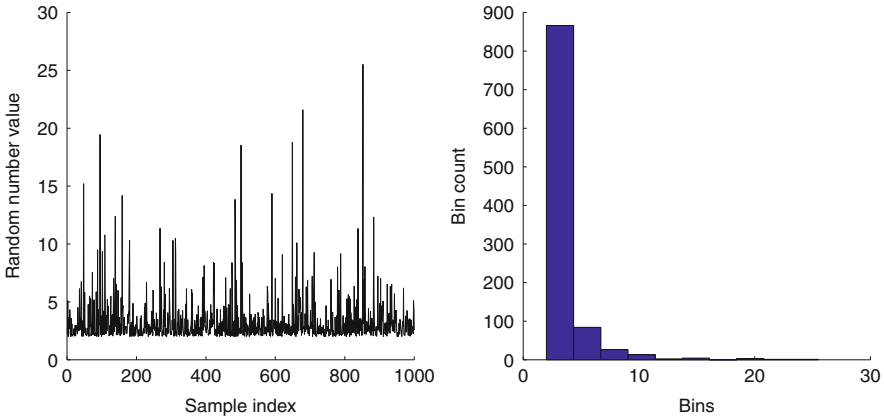



**Fig. 1.11** Random numbers generated using the Pareto distribution with $a = 2$ and $b = 2.5$. The figure on the left shows the random samples and the figure on the right shows their histogram

Pareto distribution is typically used to generate network traffic that shows bursty behavior. This means that when a traffic burst is encountered, it is very probable that the burst will continue. For such traffic, the shape parameter $b$ is typically chosen in the range 1.4–1.6.

Figure 1.11 shows the output of a Pareto random variable with position parameter $a = 2$ and shape parameter $b = 2.5$. 1,000 samples were generated in this experiment using the inversion method as described later in Sect. 1.36.2.

## 1.21 Rayleigh Distribution

The Rayleigh distribution is used to model the phenomenon of fading in a wireless communication channel. The pdf of the Rayleigh distribution is given by the formula:

$$f_X(x) = \frac{x}{a^2} e^{-x^2/(2a^2)}, \qquad x \geq 0, \quad a > 0 \tag{1.54}$$

where $a$ is the shape parameter. Figure 1.12a shows the Rayleigh pdf distribution for three different values of the shape parameter $a$. The corresponding cdf is given by:

$$F_X(x) = 1 - e^{-x^2/(2a^2)} \tag{1.55}$$

This can be proven using the formulas in Appendix A.

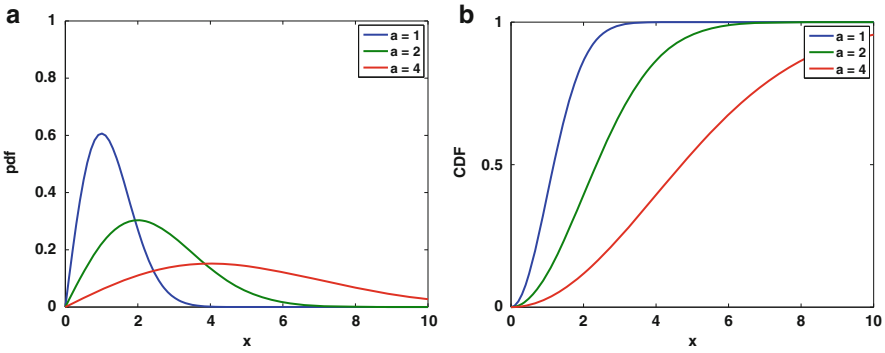



**Fig. 1.12** The Rayleigh distribution for three different values of the shape parameter $a$. (**a**) Probability density function (pdf). (**b**) Cumulative distribution function (CDF)

The mean of the Rayleigh distribution, using the formulas in Appendix A, is given by:

$$\begin{aligned}\mu &= \int_0^\infty \frac{x^2}{a^2} e^{-x^2/(2a^2}\\ &= a\sqrt{\frac{\pi}{2}}\end{aligned} \tag{1.56}$$

The standard deviation, using the formulas in Appendix A, is given by:

$$\sigma = a\sqrt{\frac{4-\pi}{2}} \tag{1.57}$$

## 1.22 Common Discrete Distributions

We discuss in the following sections some discrete random variables that are useful for network simulations.

## 1.23 Discrete Uniform Distribution

Assume $N$ is a random variable such that there are $k$ distinct sample points. The pmf for the discrete uniform RV is defined by:

$$p(n) = \begin{cases} 1/k & 1 \le n \le k \\ \\ 0 & \text{otherwise} \end{cases}$$

where $n$ is the sample index.

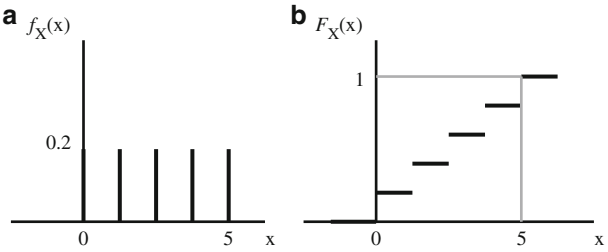



**Fig. 1.13** The uniform distribution for a discrete random variable whose values are 0, 1, ⋯, 5. (**a**) The pmf; and (**b**) is the corresponding cdf

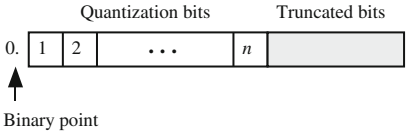



**Fig. 1.14** Truncation of a fractional number from $n + m$ bits to $n$ bits

Alternatively, the pmf can be expressed in the form:

$$p(n) = \frac{1}{k} \sum_{i=1}^{k} \delta(n - i) \tag{1.58}$$

where $\delta(j)$ is the Dirac delta function which is one when $j = 0$ and zero for all other values of $j \neq 0$.

Figure 1.13 shows the pmf for a random variable consisting of six samples which are assumed to take the values 0, 1, ⋯, 5.

*Example 1.16.* Study the statistical distribution of rounding errors in computer arithmetic.

The truncation or rounding operation is required after fixed and floating point number multiplication and also after floating point number addition. Rounding or truncation is employed to reduce the number of bits back to their original size $n$, say. Without loss of generality we assume the inputs to be fractions with the binary point at the left such that the weight of the most significant bit is $2^{-1}$ and the weight of the least significant bit (LSB) is $2^{-n}$. Figure 1.14 shows the location of the binary point for the fixed-point number and also shows the bits that will be truncated or rounded.

Truncating the extra bits to the right of LSB results in an error $e$ whose magnitude varies approximately in the range:

$$-2^{-n} \leq e \leq 2^{-n} \tag{1.59}$$

Assume the number of bits to be truncated is $m$. In that case our truncated data has $m$ bits and the number of possible error samples is $k = 2 \times 2^m - 1$. The factor 2 comes from the fact that the error could be positive or negative.

Define the discrete random variable $E$ that corresponds to the rounding or truncation error. Since the probability of any truncation error is equally likely, we have:

$$p(e) = \frac{1}{k} = \frac{1}{2^{(m+1)} - 1}$$

■

## 1.24 Bernoulli (Binary) Distribution

Many systems in communications have two outcomes. For example, a received packet might be error free or it might contain an error. For a router or a switch, a packet might arrive at a given time step or no packet arrives. Consider a chance experiment that has two mutually exclusive outcomes $A$ and $\overline{A}$ that occur with probabilities $p$ and $q$, respectively. We define the discrete random variable $X$ where $X = 1$ when $A$ occurs and $X = 0$ when $\overline{A}$ occurs. We can write:

$$p(1) = p \tag{1.60}$$

$$p(0) = q \tag{1.61}$$

where $q = 1 - p$.

Alternatively, $p(x)$ can be expressed by a single equation in the form:

$$p(x) = q\ \delta(x) + p\ \delta(x - 1) \tag{1.62}$$

The mean and variance for $X$ are:

$$\mu = p \tag{1.63}$$

$$\sigma^2 = p\ (1 - p) \tag{1.64}$$

Figure 1.15a shows the pmf for the binary distribution and Fig. 1.15b shows the cdf.

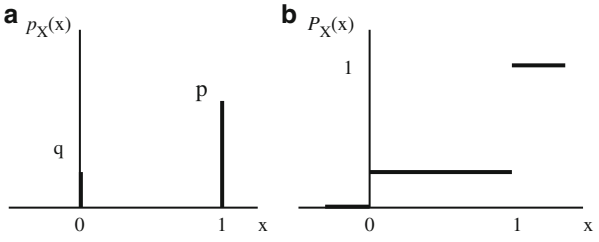



**Fig. 1.15** The binary distribution: (**a**) The pmf; and (**b**) is the corresponding cdf

*Example 1.17.* The 50/50 draw is one of the traditions of a typical minor lacrosse or baseball sports events. Spectators purchase numbered tickets. One of the tickets is picked at random and half the proceeds goes to the winner and the rest goes to support the team (or the executive council might just use the money for their own purposes). Assume you purchased one ticket and there was a total of 100 entries at the start of the draw, what are your chances of winning or losing? How much would your winnings be?

The total number of entries is 100, and the probabilities of winning or losing are:

$$p = \frac{1}{100} = 0.01$$

$$q = 1 - p = 0.99$$

Assuming the purchase price of the ticket is \$1, the winner takes \$50. The money won would be \$49. ■

## 1.25 Geometric Distribution

This distribution is encountered when success, event $A$, occurs after $n$ failures. This is the case when several devices are attached to a bus and compete for access. The probability of success is $a$ and the probability of failure is $b = 1 - a$. The pmf is the probability of success after $n$ repeated failures and is given by:

$$p(N = n) = a\, b^n \qquad \text{for } n \geq 0 \tag{1.65}$$

Alternatively, $p(n)$ can be expressed by a single equation in the form:

$$p(n) = \sum_{i \geq 0} a\, b^i\, \delta(n - i) \qquad \text{for } n \geq 0 \tag{1.66}$$

The mean and variance for $N$ are:

$$\mu = \frac{b}{a} \tag{1.67}$$

$$\sigma^2 = \frac{b}{a^2} \tag{1.68}$$

*Example 1.18.* Assume packets arrive at a certain device with probability $a$ at a given time slot. The probability that a packet does not arrive at a time slot is $b = 1 - a$.

(a) What is the probability that we have to wait for $n$ time slots before a packet arrives?
(b) What is the average number of time slots between packet arrivals?

(a) The probability that we have to wait for $n$ time slots before a packet arrives is:

$$p(n) = a\ b^n$$

(b) The average number of time slots between packet arrivals is given by:

$$E[n] = \sum_{i=0}^{\infty} n\ p(n) = \sum_{i=0}^{\infty} n\ a\ b^n$$

From Appendix A, the above equation becomes:

$$E[n] = \frac{a\ b}{(1-b)^2} = \frac{b}{a}$$

■

## 1.26 Binomial Distribution

Consider a chance experiment that has two mutually exclusive outcomes $A$ and $\overline{A}$ that occur with probabilities $a$ and $b$, respectively. We define the discrete random variable $K$ which equals the number of times that $A$ occurs in any order in $N$ trials or repetitions of the random experiment. We can write the pmf for this distribution as:

$$p(K = k) = \binom{N}{k} a^k\ b^{N-k} \qquad \text{for } 0 \leq k \leq N \tag{1.69}$$

where $b = 1 - a$. Basically, the probability of event $A$ occurring $k$ times during $N$ trials and not occurring for $N - k$ times is $a^k\ b^{N-k}$ and the number of ways of this event taking place is the binomial coefficient $C(N, k)$.

Alternatively, $p(k)$ can be expressed by a single equation in the form:

$$p(k) = \sum_{i=0}^{N} \binom{N}{i} a^i\ b^{N-i}\ \delta(k-i) \qquad \text{for } 0 \leq k \leq N \tag{1.70}$$

The mean and variance for $K$ are:

$$\mu = N\ a \tag{1.71}$$

$$\sigma^2 = N\ a\ b \tag{1.72}$$

The binomial distribution is sometimes referred to as sampling with replacement as for example when we have $N$ objects and each one has a certain property (color for example). We pick an object, inspect its property, then place it back in the population. If the selected object is removed from the population after inspection, then we would have the case of the **hypergeometric distribution**, which is also known as sampling without replacement, but this is outside our present interest.

*Example 1.19.* Assume a classroom has $n$ students. What are the chances that $m$ students have a birthday today?

The probability that a student has a birthday on a given day is $a = 1/365$ and the probability that a student has does not have a birthday on a given day is $b = 1 - a = 364/365$. The probability that $m$ students have a birthday today is:

$$p(m) = \binom{n}{m} a^m \, b^{n-m}$$

For example, when $n = 20$ and $m = 2$, we get:

$$p(2) = \binom{20}{2} a^2 \, b^{18} = 1.35 \times 10^{-3} \tag{1.73}$$

■

### 1.26.1 Approximating the Binomial Distribution

We saw in the binomial distribution that $p(k)$ is given by:

$$p(k) = \binom{N}{k} a^k \, b^{N-k} \tag{1.74}$$

where $b = 1 - a$. When $N$ is large, it becomes cumbersome to evaluate the above equation. Several approximations exist for evaluating the binomial distribution using simpler expressions. We discuss two techniques in the following two subsections.

### 1.26.2 DeMoivre–Laplace Theorem

We can approximate $p(k)$ by the Gaussian distribution provided that the following condition is satisfied [3]:

$$n \, a \, b \gg 1 \tag{1.75}$$

When this condition is true we can write:

$$p(k) \approx \frac{1}{\sigma \sqrt{\pi}} e^{-(k-\mu)^2/\sigma^2} \tag{1.76}$$

where we have to choose the two parameters $\mu$ and $\sigma$ according to the following two equations:

$$\mu = N\ a \tag{1.77}$$

$$\sigma = \sqrt{2N\ a\ b} \tag{1.78}$$

The above approximation is known as the DeMoivre–Laplace Theorem and is satisfactory even for moderate values of $n$ such as when $n \approx 5$.

### *1.26.3 Poisson Theorem*

If $Na$ is of the order of one (i.e., $Na \approx 1$), then DeMoivre–Laplace approximation is no longer valid. We can still obtain a relatively simple expressions for the binomial distribution if the following condition applies [3]:

$$n\ a \approx 1 \tag{1.79}$$

When this condition is true we can write:

$$p(k) \approx \frac{(n\ a)^k}{k!} e^{-n\ a} \tag{1.80}$$

Thus we are able to replace the binomial distribution with the Poisson distribution which is discussed in Sect. 1.27.

*Example 1.20.* A file is being downloaded from a remote site and 500 packets are required to transmit the entire file. It has been estimated that on the average 5 % of received packets through the channel are in error. Determine the probability that 10 received packets are in error using the binomial distribution and its approximations using DeMoivre–Laplace and Poisson approximations approximation.

The parameters for our binomial distribution are:

$$N = 500$$

$$a = 0.05$$

$$b = 0.95$$

The probability that 5 packets are in error is:

$$p(10) = \binom{500}{10} (0.05)^{10}\ (0.95)^{490} = 2.9165 \times 10^{-4}$$

Use the DeMoivre–Laplace Theorem with the parameters:

$$\mu = N\ a = 25$$
$$\sigma = \sqrt{N\ a\ b} = 6.892$$

In that case, the required probability is:

$$p(10) = 7.1762 \times 10^{-4}$$

Use the Poisson Theorem which results in the probability:

$$p(10) = 3.6498 \times 10^{-4}$$

Under the above parameters, the Poisson approximation gives better results than the DeMoivre–Laplace approximation. ■

## 1.27 Poisson Distribution

The Poisson distribution defines the probability $p(k)$ that an event $A$ occurs $k$ times during a certain interval. The probability is given by:

$$p(K = k) = \frac{a^k\ e^{-a}}{k!} \tag{1.81}$$

where $a > 0$ and $k = 0, 1, \cdots$.

The parameter $a$ in the above formula is usually expressed as:

$$a = \lambda\ t$$

where $\lambda$ is the rate of event $A$ and $t$ is usually thought of as time. Because we talk about rates, we usually associate Poisson distribution with time or with average number of occurrences of an event. So let us derive the expression for Poisson distribution based on this method of thinking.

Consider a chance experiment where an event $A$ occurs at a rate $\lambda$ events/second. In a small time interval $(\Delta t)$ the probability that the event $A$ occurs is $p = \lambda \Delta t$. We chose $\Delta t$ so small so that event $A$ occurs at most only once. For example, $\lambda$ might indicate the average number of packets arriving at a link per unit time. In that case the variable $t$ will indicate time. $\lambda$ might also refer to the average number of bits in error for every 1,000 bits, say. In that case, the variable $t$ would indicate the number of bits under consideration. In these situations we express the parameter $a$ in the form $a = \lambda t$ where $\lambda$ expresses the rate of some event and $t$ expresses the size or the period under consideration.

Typically the Poisson distribution is concerned with the probability that a specified **number** of occurrences of event $A$ take place in a specified interval $t$. Assuming a discrete random variable $K$ that takes the values 0, 1, 2, $\cdots$ , then the probability that $k$ events occur in a time $t$ is given by:

$$p(k) = \frac{(\lambda t)^k \, e^{-\lambda t}}{k!} \tag{1.82}$$

Alternatively, $p(k)$ can be expressed by a single equation in the form:

$$p(k) = \sum_{i \geq 0} \frac{(\lambda \, t)^i e^{-\lambda \, t}}{i!} \delta(k - i) \qquad \text{for } k \geq 0 \tag{1.83}$$

The mean and variance for $K$ are:

$$\mu = \lambda \, t \tag{1.84}$$

$$\sigma^2 = \lambda \, t \tag{1.85}$$

### *1.27.1 Deriving Poisson Distribution from Binomial Distribution*

We saw in the binomial distribution that $p(k)$ is given by:

$$p(k) = \binom{N}{k} a^k \, b^{N-k} \tag{1.86}$$

where $b = 1 - a$. When $N$ is large, it becomes cumbersome to evaluate the above equation. We can easily evaluate $p(k)$ in the special case when the number of trials $N$ becomes very large and $a$ becomes very small such that:

$$N \, a = \mu \tag{1.87}$$

where $\mu$ is nonzero and finite. This gives rise to Poisson distribution:

$$p(k) = \lim_{N \to \infty, a \to 0} \binom{N}{k} a^k \, b^{N-k} \approx \frac{(\mu)^k \, e^{-\mu}}{k!} \tag{1.88}$$

We can prove the above equation using the following two simplifying expressions. We start by using Stirling's formula:

$$N! \approx \sqrt{2\pi} N^N e^{-N} \tag{1.89}$$

and the following limit from calculus:

$$\lim_{N\to\infty}(1-\frac{a}{N})^N = e^{-a} \tag{1.90}$$

Using the above two expressions, we can write [10]:

$$p(k) = \binom{N}{k} a^k\ b^{N-k} \tag{1.91}$$

$$= \frac{N!}{k!(N-k)!} a^k\ (1-b)^{N-k} \tag{1.92}$$

$$= \frac{N^N e^{-N}}{k!\,(N-k)^{N-k}\,e^{-(N-k)}} \left(\frac{\mu}{N}\right)^k \left(1-\frac{\mu}{N}\right)^{N-k} \tag{1.93}$$

$$= \frac{\mu^k}{k!} \left(\frac{N}{N-k}\right)^{N-k} \left(1-\frac{\mu}{N}\right)^{N-k} \tag{1.94}$$

$$\approx \frac{\mu^k}{k!} e^{-\mu} \tag{1.95}$$

where $\mu = N\ a$. This gives the expression for the Poisson distribution. This is especially true when $N > 50$ and $a < 0.1$ in the binomial distribution.

*Example 1.21.* Packets arrive at a device at an average rate of 500 packets per second. Determine the probability that four packets arrive during 3 ms.

We have $\lambda = 500$, $t = 3 \times 10^{-3}$, and $k = 4$:

$$p(4) = \frac{(1.5)^4\, e^{-1.5}}{4!} = 4.7 \times 10^{-2}$$

■

## 1.28 Systems with Many Random Variables

We reviewed in Sect. 1.11 the concept of a random variable where the outcome of a random experiment is mapped to a single number. Here, we consider random experiments whose output is mapped to two or more numbers. Many systems based on random phenomena are best studied using the concept of multiple random variables. For example, signals coming from a Quadrature Amplitude Modulation (QAM) system are described by the equation:

$$v(t) = a\cos(\omega t + \phi) \tag{1.96}$$

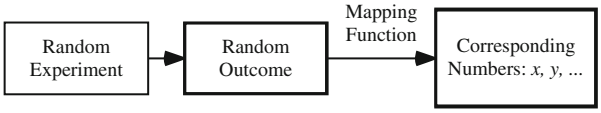



**Fig. 1.16** The sequence of events leading to assigning multiple numerical values to the outcome of a random experiment

Incoming digital signals are modulated by assigning different values to $a$ and $\phi$. In that sense, QAM modulation combines amplitude and phase modulation techniques. The above signal contains two pieces of information: viz, the amplitude $a$ and the phase $\phi$ that correspond to two random variables $A$ and $\Phi$. So every time we sample the signal $v(t)$, we have to find two values for the associated random variables $A$ and $\Phi$.

Figure 1.16 graphically shows the sequence of events leading to assigning multiple numerical values to the outcome of a random experiment. First we run the experiment then we observe the resulting outcome. Each outcome is assigned multiple numerical values.

As an example, we could monitor the random events of packet arrival at the input of a switch. Several outcomes of this random event could be observed such as: (1) packet length; (2) packet type (voice, video, data, etc.); (3) interarrival time—i.e., the time interval between arriving packets; (4) destination address. In these situations, we might want to study the relationships between these random variables in order to understand the underlying characteristics of the random experiment we are studying.

## 1.29 Joint cdf and pdf

Assume our random experiment gives rise to two discrete random variables $X$ and $Y$. We define the joint cdf of $X$ and $Y$ as:

$$F_{XY}(x, y) = p(X \leq x, Y \leq y) \tag{1.97}$$

When the two random variables are independent, we can write:

$$p(X \leq x, Y \leq y) = p(X \leq x)\ p(Y \leq y) \tag{1.98}$$

Thus for independent random variables the joint cdf is simply the product of the individual cdf's:

$$F_{XY}\,(x, y) = F_X\,(x)\ F_Y\,(y) \tag{1.99}$$

For continuous RVs, the joint pdf is defined as:

$$f_{XY}(x, y) = \frac{\partial^2 F_{XY}}{\partial x \, \partial y} \tag{1.100}$$

The joint pdf satisfies the following relation:

$$\int_{x,y} f_{XY}(x, y) \, dx \, dy = 1 \tag{1.101}$$

The two random variables are **independent** when the joint pdf can be expressed as the product of the individual pdf's:

$$f_{XY}(x, y) = f_X(x) \; f_Y(y) \tag{1.102}$$

For discrete RVs, the joint pmf is defined as the probability that $X = x$ and $Y = y$:

$$p_{XY}(x, y) = p(X = x, \; Y = y) \tag{1.103}$$

The joint pmf satisfies the following relation:

$$\sum_x \sum_y p_{XY}(x, y) = 1 \tag{1.104}$$

Two random variables are **independent** when the joint pmf can be expressed as the product of the individual pmf's:

$$p_{XY}(x, y) = p(X = x) \; p(Y = y) \tag{1.105}$$

*Example 1.22.* Assume arriving packets are classified according to two properties: packet length (short or long) and packet type (voice or data). Random variable $X = 0, 1$ is used to describe packet length and random variable $Y = 0, 1$ is used to describe packet type. The probability that the received packet is short is 0.9 and probability that it is long is 0.1. When a packet is short, the probability that it is a voice packet is 0.4 and probability that it is a data packet is 0.6. When a packet is long, the probability that it is a voice packet is 0.05 and probability that it is a data packet is 0.95. Find the joint pmf of $X$ and $Y$.

We have the mapping:

$$X = \begin{cases} 0 & \text{short packet} \\ 1 & \text{long packet} \end{cases}$$

$$Y = \begin{cases} 0 & \text{voice packet} \\ 1 & \text{data packet} \end{cases}$$

Based on the above mapping, and assuming independent RVs, we get:

$$\begin{aligned} p_{XY}(0,0) &= 0.9 \times 0.4 = 0.36 \\ p_{XY}(0,1) &= 0.9 \times 0.6 = 0.54 \\ p_{XY}(1,0) &= 0.1 \times 0.05 = 0.005 \\ p_{XY}(1,1) &= 0.1 \times 0.95 = 0.095 \end{aligned}$$

Note that the sum of all the probabilities must add up to 1. ■

## 1.30 Individual pmf from a Given Joint pmf

Sometimes we want to study an individual random variable even though our random experiment generates multiple RVs. An individual random variable is described by its pmf which is obtained as:

$$p_X(x) = p(X = x) \tag{1.106}$$

If our random experiment generates two RVs $X$ and $Y$, then we have two individual pmf's given by:

$$p_X(x) = \sum_y p_{XY}(x, y) \tag{1.107}$$

$$p_Y(y) = \sum_x p_{XY}(x, y) \tag{1.108}$$

From the definition of pmf we must have:

$$\sum_x p_X(x) = 1 \tag{1.109}$$

$$\sum_y p_Y(y) = 1 \tag{1.110}$$

*Example 1.23.* Assume a random experiment that generates two random variables $X$ and $Y$ with the given joint pmf.

| $p_{XY}(x, y)$ | $x = 1$ | $x = 2$ | $x = 3$ |
|---|---|---|---|
| $y = 0$ | 0.1 | 0.05 | 0.1 |
| $y = 3$ | 0.1 | 0.05 | 0 |
| $y = 7$ | 0 | 0.1 | 0.1 |
| $y = 9$ | 0.2 | 0 | 0.2 |

Find the individual pmf for $X$ and $Y$. Are $X$ and $Y$ independent RVs?
We have

| $p_X(x)$ | $x = 1$ | $x = 2$ | $x = 3$ |
|---|---|---|---|
| | 0.4 | 0.2 | 0.4 |

and

| $p_Y(y)$ | $y = 0$ | $y = 3$ | $y = 7$ | $y = 9$ |
|---|---|---|---|---|
| | 0.25 | 0.15 | 0.2 | 0.4 |

■

## 1.31 Expected Value

The joint pmf helps us find the expected value of one of the random variables.

$$\mu_X = E(X) = \sum_x \sum_y x \; p_{XY}(x, y) \tag{1.111}$$

*Example 1.24.* In Example 1.23 above, find the expected values for the random variables $X$ and $Y$.
We can write:

$$\begin{aligned} \mu_X &= 1 \times 0.4 + 2 \times 0.2 + 3 \times 0.4 && = 2 \\ \mu_y &= 0 \times 0.25 + 3 \times 0.15 + 7 \times 0.2 + 9 \times 0.4 && = 5.45 \end{aligned}$$

■

## 1.32 Correlation

The correlation between two random variables is defined as:

$$\begin{aligned} r_{XY} &= E[XY] \\ &= \sum_x \sum_y x \; y \; p_{XY}(x, y) \end{aligned} \tag{1.112}$$

We say that the two random variables $X$ and $Y$ are **orthogonal** when $r_{XY} = 0$.

## 1.33 Variance

The variance of a random variable is defined as:

$$\begin{aligned}\sigma_X^2 &= E[(X-\mu_X)^2] \\ &= \sum_x \sum_y (x-\mu_X)^2 \; p_{XY}(x,y)\end{aligned} \tag{1.113}$$

The following equation can be easily proven:

$$\sigma_X^2 = E[X^2] - \mu_X^2 \tag{1.114}$$

## 1.34 Covariance

The covariance between two random variables is defined as:

$$\begin{aligned}c_{XY} &= E[(X-\mu_X)(Y-\mu_Y)] \\ &= \sum_x \sum_y (x-\mu_X)(y-\mu_Y) \; p_{XY}(x,y)\end{aligned} \tag{1.115}$$

The following equation can be easily proven:

$$c_{XY} = r_{XY} - \mu_X \; \mu_Y \tag{1.116}$$

We say that the two random variables $X$ and $Y$ are **uncorrelated** when $c_{XY} = 0$.

The **correlation coefficient** $\rho_{XY}$ is defined as:

$$\rho_{XY} = c_{XY} / \sqrt{\sigma_X^2 \; \sigma_Y^2} \tag{1.117}$$

When we are dealing with two random variables obtained from the same random process, the correlation coefficient would be written as:

$$\rho_{X(n)} = c_X(n) / \sigma_X^2 \tag{1.118}$$

We expect that the correlation coefficient would decrease as the value of $n$ becomes large to indicate that the random process "forgets" its past values.

## 1.35 Transforming Random Variables

Mathematical packages usually have functions for generating random numbers following the uniform and normal distributions only. However, when we are simulating communication systems, we need to generate network traffic that follows other types of distributions. How can we do that? Well, we can do that through

transforming random variables which is the subject of this section. Section 1.36 will show how to actually generate the random numbers using the techniques of this section.

### 1.35.1 Continuous Case

Suppose we have a random variable $X$ with known pdf and cdf and we have another random variable $Y$ that is a function of $X$:

$$Y = g(X) \tag{1.119}$$

$X$ is named the **source** random variable and $Y$ is named the **target** random variable. We are interested in finding the pdf and cdf of $Y$ when the pdf and cdf of $X$ are known. The probability that $X$ lies in the range $x$ and $x+dx$ is given from (1.28) by:

$$p(x \leq X \leq x + dx) = f_X(x)\, dx \tag{1.120}$$

But this probability must equal the probability that $Y$ lies in the range $y$ and $y+dy$. Thus we can write:

$$f_Y(y)\, dy = f_X(x)\, dx \tag{1.121}$$

where $f_Y(y)$ is the pdf for the random variable $Y$ and it was assumed that the function $g$ was monotonically increasing with $x$. If $g$ was monotonically decreasing with $x$, then we would have:

$$f_Y(y)\, dy = -f_X(x)\, dx \tag{1.122}$$

The above two equations define the **fundamental law of probabilities**, which is given by:

$$|f_Y(y)\, dy| = |f_X(x)\, dx| \tag{1.123}$$

or:

$$f_Y(y) = f_X(x) \left| \frac{dx}{dy} \right| \tag{1.124}$$

since $f_Y(y)$ and $f_X(x)$ are always positive.

In the discrete case the fundamental law of probability gives:

$$p_Y(y) = p_X(x) \tag{1.125}$$

where $p_X(x)$ is the given pmf of the source random variable and $p_Y(y)$ is the pmf of the target random variable.

*Example 1.25.* Given the random variable $X$ whose pdf has the form:

$$f_X(x) = e^{-x^2} \qquad x \geq 0 \tag{1.126}$$

Find the pdf for the random variable $Y$ that is related to $X$ by the relation:

$$Y = X^2 \tag{1.127}$$

We have:

$$x = \pm\sqrt{y} \tag{1.128}$$

$$\frac{dx}{dy} = \pm\frac{1}{2\sqrt{y}} \tag{1.129}$$

From (1.124) we can write:

$$f_Y(y) = \frac{1}{2\sqrt{y}} \times e^{-x^2} \tag{1.130}$$

Substituting the value of $x$ in terms of $y$, we get:

$$f_Y(y) = \frac{1}{2\sqrt{y}} e^{-y} \qquad y \geq 0 \tag{1.131}$$

■

*Example 1.26.* Assume the sinusoidal signal:

$$x = \cos \omega t$$

where the signal has a random frequency $\omega$ that varies uniformly in the range $\omega_1 \leq \omega \leq \omega_2$. The frequency is represented by the random variable $\Omega$. What are the expected values for the random variables $\Omega$ and $X$?

We can write:

$$f_\Omega = 1/(\omega_2 - \omega_1)$$

and $E[\Omega]$ is given by the expression:

$$E[\Omega] = \frac{1}{\omega_2 - \omega_1} \int_{\omega_1}^{\omega_2} \omega \, d\omega \tag{1.132}$$

$$= \frac{\omega_2 + \omega_1}{2} \tag{1.133}$$

We need to find $f_X$ and $E[X]$. For that we use the fundamental law of probability. First, we know that $-1 \leq x \leq 1$ from the definition of the sine function, so $f_X(x) = 0$ for $|x| > 1$. Now we can write:

$$f_X(x) = f_\Omega(\omega) \left| \frac{d\omega}{dx} \right| \qquad |x| \leq 1 \tag{1.134}$$

Now we have:

$$\omega = \frac{1}{t}\cos^{-1} x \tag{1.135}$$

and:

$$\left|\frac{d\omega}{dx}\right| = \frac{1}{t\sqrt{1-x^2}} \qquad |x| \leq 1 \tag{1.136}$$

Thus we get:

$$f_X(x) = \frac{1}{\omega_2 - \omega_1} \times \frac{1}{t\sqrt{1-x^2}} \qquad |x| \leq 1 \tag{1.137}$$

The expected value for $X$ is given by:

$$E[X] = \frac{1}{(\omega_2 - \omega_1)\,t} \int_{-1}^{1} \frac{x}{\sqrt{1-x^2}}\, dx = 0 \tag{1.138}$$

due to the odd symmetry of the function being integrated. ■

*Example 1.27.* Suppose our source random variable is uniformly distributed and our target random variable $Y$ is to have a pdf given by:

$$f_Y(y) = \lambda e^{-\lambda y} \qquad \lambda > 0 \tag{1.139}$$

Derive $Y$ as a function of $X$.

This example is fundamentally important since it shows how we can find the *transformation* that allows us to obtain random numbers following a desired pdf given the random numbers for the uniform distribution. This point will be further discussed in the next section.

We use (1.124) to write:

$$\lambda e^{-\lambda y} = f_X(x)\left|\frac{dx}{dy}\right| \tag{1.140}$$

Assume the source random variable $X$ is confined to the range 0 to 1. From Sect. 1.17 we have $f_X(x) = 1$ and we can write:

$$\lambda e^{-\lambda y} = \frac{dx}{dy} \tag{1.141}$$

Integrating, we get:

$$e^{-\lambda y} = x \tag{1.142}$$

and we obtain the dependence of $Y$ on $X$ as:

$$Y = g(X) = -\frac{\ln X}{\lambda} \qquad 0 \leq x \leq 1 \tag{1.143}$$

■

### *1.35.2 Discrete Case*

In the discrete case the fundamental law of probability gives:

$$p_Y(y) = p_X(x) \tag{1.144}$$

where $p_X(x)$ is the given pmf of the source random variable and $p_Y(y)$ is the pmf of the target random variable. The values of $Y$ are related to the values of $X$ by the equation:

$$Y = g(X) \tag{1.145}$$

The procedure for deriving the pmf for $Y$ given the pmf for $X$ is summarized as follows:

1. For each value of $x$ obtain the corresponding value $p(X = x)$ through observation or modeling.
2. Calculate $y = g(x)$.
3. Associate $y$ with the probability $p(X = x)$.

## 1.36 Generating Random Numbers

We review briefly in this section how to generate sequences of random numbers obeying one of the distributions discussed in the previous section. This background is useful to know even if packages exist that fulfill our objective.

Before we start, we are reminded of Von Neumann's remark on the topic:

> Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin. (John Von Neumann 1951)

### *1.36.1 Uniform Distribution*

To generate a sequence of integer random numbers obeying the uniform distribution using C programming, one invokes the function `srand(seed)`. This function

returns an integer in the range 0 to `RAND_MAX` [11]. When a continuous random number is desired, `drand` is used. The function `rand` in MATLAB is used to generate random numbers having uniform distribution.

### 1.36.2 Inversion Method

To generate a number obeying a given distribution, we rely on the fundamental law of probabilities summarized by (1.124). When $X$ has a uniform distribution over the range $0 \leq x \leq 1$, we can use the following procedure for obtaining $y$. We have:

$$\frac{dx}{dy} = f(y) \tag{1.146}$$

where $f(x) = 1$ and $f(y)$ is the function describing the pdf of the target distribution. Integrating, we get:

$$\int_0^y f(z)\, dz = x \tag{1.147}$$

Thus we have:

$$F(y) = x \tag{1.148}$$

$$y = F^{-1}(x) \tag{1.149}$$

The procedure then to obtain the random numbers corresponding to $y$ is to generate $x$ according to any standard random number generator producing a uniform distribution. Then use the above equation to provide us with $y$. Thus the target random number value $y$ is obtained according to the following steps:

1. Obtain the source random number $x$ having a uniform distribution.
2. Lookup the value of $F(y)$ that is numerically equal to $x$ according to (1.148). $F(y)$ is either computed or stored in a lookup table.
3. Find the corresponding value of $y$ according to (1.149). If the inverse function is not available, then a lookup table is used and $y$ is chosen according to the criterion $F(y) \leq x \leq F(y + \epsilon)$, where $y + \epsilon$ denotes the next entry in the table.

Graphically, the procedure is summarized in Fig. 1.17.

The inversion technique was used to generate random numbers that obey the Pareto distribution using MATLAB. In that case, when $x$ is the trial source input, the output $y$ is given from the $F^{-1}$ by the equation:

$$y = -\frac{a}{\exp\left[(1/b)\ln(1-x)\right]} \tag{1.150}$$

Figure 1.11 shows the details of the method for generating random numbers following the Pareto distribution. The Pareto parameters chosen were $a = 2$,

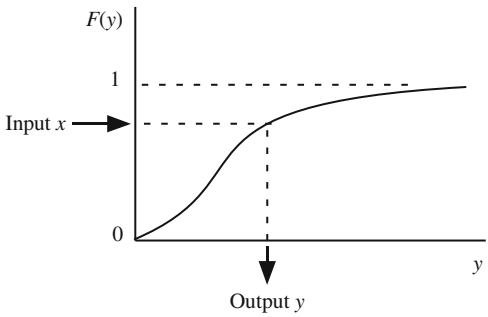



**Fig. 1.17** Transformation method for finding $y$ given $x$

$b = 2.5$, minimum value for data was $x_{\min} = a$. One thousand samples were chosen to generate the data. Note that the minimum value of the data equals $a$ according to the restrictions of the Pareto distribution.

### *1.36.3 Rejection Method*

The previous method requires that the target cdf be known and computable such that the inverse of the function can be determined either analytically or through using a lookup table. The rejection method is more general since it only requires that the target pdf is known and computable. We present here a simplified version of this method.

Assume we want to generate the random numbers $y$ that lie in the range $a \leq y < b$ and follow the target pdf distribution specified by $f(y)$. We choose the uniform distribution $g(y)$ that covers the same range $a$–$b$ such that the following condition is satisfied for all the range of $x$:

$$g(y) = \frac{1}{b-a} > f(y) \tag{1.151}$$

If this condition cannot be satisfied, then a different $g(y)$ must be chosen that might not follow the uniform distribution. We proceed as follows.

1. Obtain the source random number $x$ using any random number generator having the uniform distribution.
2. Accept the candidate value $x$ as the target value $y = x$ with probability:

$$p\ (\text{accept } x) = \frac{f(y)}{g(y)} \tag{1.152}$$

$$= (b-a)\ f(y) \tag{1.153}$$

We are assured by (1.151) that the above expression for the probability is always valid.

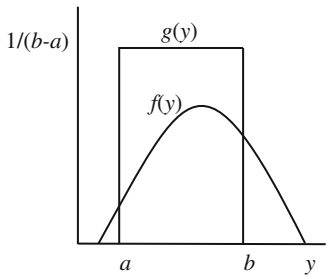



**Fig. 1.18** The rejection method

This technique is not efficient when $f(y)$ is mostly small with few large peaks. Since most of the candidate points will be rejected, as can be seen from Fig. 1.18.

### *1.36.4 Importance Sampling Method*

Importance sampling is a generalization of the rejection method. A trial pdf, $g(y)$, is chosen as in the rejection method. It is **not** necessary here to choose $g(y)$ such that it is larger than $f(y)$. Each point $y$ has a weight associated with it given by:

$$w(y) = \frac{f(y)}{g(y)} \tag{1.154}$$

Based on this array of weights, we choose a weight $W$ that is slightly larger than the maximum value of $w(y)$:

$$W = \max[w(y)] + \epsilon \qquad \epsilon > 0 \tag{1.155}$$

The method is summarized in the following steps.

1. Obtain the source random number $x$ that has a uniform distribution.
2. Apply the inversion method using $g(y)$ to obtain a candidate value for $y$.
3. Accept the candidate $y$ value with probability:

$$p(\text{accept } y) = \frac{w(y)}{W} \tag{1.156}$$

## 1.37 Problems

### *1.37.1 The Multiplication Principle*

**1.1.** A pair of fair dice is rolled once. Identify one possible outcome and identify the sample space of this experiment.

**1.2.** Consider the above experiment where we are interested in the event that "seven" will show. What are the outcomes that constitute the event?

**1.3.** A student has to take three courses from three different fields. Field $A$ offers 5 courses, field $B$ offers 10 courses, and field $C$ offers 3 courses. In how many ways can the student select the course load?

**1.4.** A car dealer specializes in selling three types of vehicles: sedans, trucks, and vans. Each vehicle could be rated as excellent, okay, lemon, or "bring your own jumper cables." How many different ways can a customer buy a vehicle?

**1.5.** Car license plates in British Columbia consist of three letters followed by three numbers. How many different license plate numbers could be formed?

### *1.37.2 Permutations*

**1.6.** Internet packets have different lengths. Assume 10 packets have been received such that two are over-length, four have medium length, and four are short. How many different packet patterns are possible?

**1.7.** In asynchronous transfer mode, a time frame is divided into ten time slots such that each time slot can be used by any user. Suppose that a frame is received that contains three packets due to user 1, two due to user 2 and the rest of the time slots were empty. How many frame patterns are possible?

**1.8.** A router receives 15 packets from 15 different sources. How many ways could these packets be received?

**1.9.** In Problem 1.8, of the 15 packets received, five were due to one user and the rest were due to 10 different users. How many ways could the packets types be received?

### *1.37.3 Combinations*

**1.10.** A packet buffer can store 50 packets. If it is known that 15 of those packets belong to a certain user. The rest belong to different users. How many possibilities can these packets can be stored in the buffer?

**1.11.** A router receives 10 packets such that four of them are in error. How many different packet patterns are possible?

**1.12.** In a cellular phone environment, a cell has 100 users and there are only 16 available channels. How many different possibilities exist for choosing among these users?

### 1.37.4 Probability

**1.13.** A signal source randomly moves from being active to being idle. The active period lasts 5 s and the idle period lasts 10 s. What is the probability that the source will be active when it is sampled?

**1.14.** In a wireless channel a certain user found that for each 1,000 packets transmitted 10 were lost, 100 were in error, and 50 were delayed. Find

(a) Probability that a packet is lost.
(b) Probability that a packet is received without delay.
(c) Probability that a packet is received without delay and without errors.
(d) Probability that a packet is received without delay or without errors.

**1.15.** Assume a gambler plays double or nothing game using a fair coin and he/she starts with one dollar. What is the probability that he/she will wind up winning \$1,024?

**1.16.** Four friends decide to play the following game. A bottle is spun and the person that the bottle points to is unceremoniously thrown out of the game. What is the probability that you are still in the game after $n$ spins? Is there an upper limit on the value of $n$?

**1.17.** A bird breeder finds that the probability that a chick is male is 0.2 and a female is 0.8. If the nest has three eggs, what is the probability that two male chicks will be produced?

### 1.37.5 Random Variables

**1.18.** Indicate the range of values that the random variable $X$ may assume and classify the random variable as finite/infinite and continuous/discrete in the following experiments.

(a) The number of times a coin is thrown until a head appears.
(b) The wait time in minutes until a bus arrives at the bus stop.
(c) The duration of a telephone conversation.
(d) The number of students in a classroom.
(e) The number of retransmissions until a packet is received error free.

**1.19.** A packet is received either correctly or in error on a certain channel. A random variable $X$ is assigned a value equal to the number of error-free packets in a group of three packets. Assume that the error in a packet occurs independent of the other packets.

(a) List all the possible outcomes for the reception of three packets.
(b) List all the possible values of $X$.
(c) Express the probability for the event $x = 2$ in terms of $f_X(x)$ and $F_X(x)$.

**1.20.** In some communication scheme, when a packet is received in error, a request for retransmission is issued until the packet is received error free. Let the random variable $Y$ denote the number of retransmission requests. What are the values of $Y$?

**1.21.** Packets arrive at a certain input randomly at each time step (a time step is defined here as the time required to transmit or receive one complete packet). Let the random variable $W$ denote the wait time (in units of time steps) until a packet is received. What values may $W$ assume?

### *1.37.6 The Cumulative Distribution Function*

**1.22.** Assume a random variable $X$ whose cdf is $F(x)$. Expresses the probability $p(X > x)$ as a function of $F(x)$.

**1.23.** A system monitors the times between packet arrivals starting at time $t = 0$. This time is called the *interarrival time* of packets. The interarrival time is a random variable $T$ with cdf $F_T(t)$. The probability that the system receives a packet in the time interval $(t, t + \delta t)$ is given by $p(t)\delta t$.

(a) Find the probability that the system receives a packet in a time less than or equal to $t$.
(b) Find the probability that the system receives a packet in a time greater than $t$.

**1.24.** Plot the cdf for the random variable in Problem 1.19.

**1.25.** Explain the meaning of Eq. (1.19) for the cdf function. Note that (1.19) is really a restatement of (1.7) since the events $X \leq x_1$ and $x_1 \leq X \leq x_2$ are mutually exclusive.

**1.26.** Plot the cdf for the random variable in Problem 1.19

**1.27.** A buffer contains ten packets. Four packets contain an error in their payload and six are error-free. Three packets are picked at random for processing. Let the random variable $X$ denote the number of error free packets selected.

(a) List all possible outcomes of the experiment.
(b) Find the value assigned $X$ for each outcome.
(c) Find the probability associated with each value of $X$.
(d) Plot the cdf for this random variable.

Note that this problem deals with sampling without replacement; i.e., we pick a packet but do not put it back in the buffer. Hence the probability of picking an error-free packet will vary depending on whether it was picked first, second, or third.

**1.28.** Sketch the pdf associated with the random variable in Problem 1.23.

**1.29.** Sketch the pdf associated with the random variable in Problem 1.31.

**1.30.** Sketch the pdf associated with the random variable in Problem 1.32.

**1.31.** A router has ten input ports and at a given time instance each input could receive a packet with probability $a$ or it could be idle (with probability $b = 1 - a$). Let the random variable $X$ denote the number of active input ports.

(a) List all possible outcomes of the experiment.
(b) Find the value assigned $X$ for each outcome.
(c) Find the probability associated with each value of $X$.
(d) Sketch the cdf for this random variable.

**1.32.** A traffic source generates a packet at a certain time step with probability $a$. Let the random variable $X$ denote the number of packets produced in a period $T = 5$ time steps.

(a) List all possible outcomes of the experiment.
(b) Find the value assigned $X$ for each outcome.
(c) Find the probability associated with each value of $X$.
(d) Sketch the cdf for this random variable.

### *1.37.7 The Probability Density Function*

**1.33.** Sketch the pdf and cdf for the binomial distribution. Assume for $a = 0.3$ and $N = 5$.

**1.34.** Sketch the pdf associated with the random variable in Problem 1.19.

**1.35.** Sketch the pdf associated with the random variable in Problem 1.27.

**1.36.** Sketch the pdf associated with the random variable in Problem 1.31.

**1.37.** Sketch the pdf associated with the random variable in Problem 1.32.

### *1.37.8 Expected Value*

**1.38.** Prove that (1.34) on page 15 converges to (1.33) as $n \to \infty$. Start your analysis by (a) assuming that $n$ samples are grouped such that $n_j$ samples produce the same outcome. (b) Use the definition of probability in (1.6) on page 8 to complete your proof.

**1.39.** What are the expected values for the random variables $\Theta$ and $Q$ in Example 1.14?

**1.40.** Assume a Poisson process, rate parameter is $\lambda$, that gives rise to two random variables $X_1$ and $X_2$ which correspond to $k$ packets received at times $t_1$ and $t_2$, respectively, where $t_2 \geq t_1$. (a) Find the mean and variance for these two random variables. (b) Now define a new random variable $Y = X_2 - X_1$ and find its mean and variance.

**1.41.** Find the expected value for the random variable in Problem 1.19.

**1.42.** Find the expected value for the random variable in Problem 1.27.

**1.43.** Find the expected value for the random variable in Problem 1.31.

**1.44.** Find the expected value for the random variable in Problem 1.32.

**1.45.** Repeat Example 1.16 when the random variable $E$ is treated as a discrete random variable.

### *1.37.9 The Uniform Distribution*

**1.46.** Write down the pdf and cdf for the uniform distribution of a continuous random variable $X$ that spans the range $a \leq x < b$.

**1.47.** Repeat the above problem when the random variable is discrete and has $n$ discrete values in the same range given above.

**1.48.** Find the average value and variance for the random variable in Problem 1.46.

**1.49.** Find the average value and variance for the random variable in Problem 1.47.

### *1.37.10 The Binomial Distribution*

**1.50.** Prove that the mean and standard deviation of the binomial distribution are $Na$ and $\sqrt{Nab}$, respectively.

**1.51.** Sketch the pdf for the binomial distribution. Assume values for $a$ and $N$.

**1.52.** The probability of error in a single packet is $10^{-4}$. What is the probability that three or less errors occur in 1,000 packets assuming binomial distribution.

**1.53.** Assume $q$ as the probability that people making reservations on a certain flight will not show up. The airline then sells $t$ tickets for a flight that takes only $s$ passengers ($t > s$). Write an expression for the probability that there will be a seat available for every passenger that shows up. What is that probability for the special case when only one seat is over sold (i.e., $t = s + 1$)?

### *1.37.11 The Poisson Distribution*

**1.54.** Verify that the mean and standard deviation for the binomial and Poisson distributions become almost identical for large $N$ and small $p$ as was discussed in Sect. 1.27.

**1.55.** The probability of a defective electronic component is 0.1. Find

(a) The mean and standard deviation for the distribution of defective components in a batch of 500 components using the Poisson and binomial distributions.
(b) The probability that 2 components are defective in a batch of 500 components.

**1.56.** Sketch the pdf for the Poisson distribution for different values of $\lambda t$ and $k$. Comment on your results.

**1.57.** Sketch on one graph the binomial and Poisson distributions for the case $N = 5$ and $p = 0.1$. Assume $\lambda t = 0.5$.

**1.58.** Repeat question 1.52 assuming Poisson distribution with $\lambda t = 0.1$ where we assumed the "rate" $\lambda$ of occurrence of error per packet is $10^{-4}$ errors/packet and the "duration" of interest is $t = 1{,}000$ packets.

**1.59.** Five percent of the rented videos are worth watching. Find the probability that in a sample of 10 videos chosen at random, exactly two will be worth watching using (a) binomial distribution, (b) Poisson distribution.

### *1.37.12 The Exponential Distribution*

**1.60.** Find the cdf for the exponential distribution.

**1.61.** Prove that the Pareto distribution formula given by (1.49) is a valid pdf (i.e., the area under the curve is 1).

### *1.37.13 Joint pmf*

**1.62.** Consider the random experiment of throwing a dart at a target. The point of impact is characterized by two random variables $X$ and $Y$ to indicate the location of the dart assuming the center is the point of origin. We can justifiably assume that $X$ and $Y$ are statistically independent.

(a) Write down the joint cdf $F_{XY}(x, y)$ as a function of the individual cdf's $F_X(x)$ and $F_Y(y)$.
(b) Write down the joint pdf $f_{XY}(x, y)$ as a function of the individual pdf's $f_X(x)$ and $f_Y(y)$.
(c) Write down an expressions for $f_X(x)$ and $f_Y(y)$ assuming that each follows the normal (Gaussian) distribution.

**1.63.** Find the correlation between the two random variables $X$ and $Y$ in Example 1.22 on page 33.

**1.64.** Find the variance of random variables $X$ in Example 1.22 on page 33.

**1.65.** Find the covariance between the two random variables $X$ and $Y$ in Example 1.22 on page 33.

**1.66.** Prove (1.114) on page 36.

**1.67.** Find the correlation coefficient for the two random variables $X$ and $Y$ in Example 1.22 on page 33.

### *1.37.14 Random Numbers*

**1.68.** Use the inversion method to generate $y$ in the range 1 to 5 such that the target pdf is $f(y) \propto 1/\sqrt{y}$.

**1.69.** Generate the random number $y$ that has pdf $f(y) = (1 + y)/\sqrt{y}$ using the importance sampling method.

## References

1. R.D. Yates, D.J. Goodman, *Probability and Stochastic Processes* (Wiley, New York, 1999)
2. P.Z. Peebles, *Probability, Random Variables, and Random Signal Principles* (McGraw-Hill, New York, 1993)
3. A. Papoulis, *Probability, Random Variables, and Stochastic Processes* (McGraw-Hill, New York, 1984)
4. G.R. Cooper, C.D. McGillem, *Probability Methods of Signal and System Analysis* (Oxford University Press, New York, 1999)
5. "statistics" Encyclopedia Britannica Online. http://search.eb.com/bol/topic?eu=115242&sctn=8
6. A. Erramilli, G. Gordon, W. Willinger, Applications of fractals in engineering for realistic traffic processes. Int. Teletraffic Conference **14**, 35–44 (1994)
7. W. Leland, M. Taqqu, W. Willinger, D. Wilson, On the self-similar nature of Ethernet traffic. IEEE/ACM Trans. Networking **2**, 1–15 (1994)
8. W. Willinger, M.Taqqu, R. Sherman, D. Wilson, Self-similarity through high variability statistical analysis of Ethernet LAN traffic at the source level. IEEE/ACM Trans. Networking **2** (1997)
9. T. Hettmansperger, M. Keenan, Tailweight, statistical interference and families of distributions-A brief survey in *Statistical Distributions in Scientific Work*, vol. 1, ed. by G.P. Patil et al. (Kluwer, Boston, 1980)
10. R. Syski, *Random Processes: A First Look* (Marcel Dekker, New York, 1979)
11. W.H. Press, S.T. Teukolsky, W.T. Vetterling, B.P. Fhannery, *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge University Press, Cambridge, 1992)

# Chapter 2
# Random Processes

## 2.1 Introduction

We saw in Sect. 1.11 on page 10 that many systems are best studied using the concept of random variables where the outcome of a random experiment was associated with some numerical value. Next, we saw in Sect. 1.28 on page 31 that many more systems are best studied using the concept of multiple random variables where the outcome of a random experiment was associated with multiple numerical values. Here we study *random processes* where the outcome of a random experiment is associated with a *function of time* [1]. Random processes are also called *stochastic processes*. For example, we might study the output of a digital filter being fed by some random signal. In that case, the filter output is described by observing the output waveform.

Thus a *random process* assigns a random *function of time* as the outcome of a random experiment. Figure 2.1 graphically shows the sequence of events leading to assigning a function of time to the outcome of a random experiment. First we run the experiment then we observe the resulting outcome. Each outcome is associated with a time function $x(t)$.

A random process $X(t)$ is described by:

- The *sample space* $S$ which includes all possible outcomes $s$ of a random experiment.
- The *sample function* $x(t)$ which is the time function associated with an outcome $s$. The values of the sample function could be discrete or continuous.
- The *ensemble* which is the set of all possible time functions produced by the random experiment.
- The time parameter $t$ which could be continuous or discrete.
- The statistical dependencies among the random processes $X(t)$ when $t$ is changed.

Based on the above descriptions, we could have four different types of random processes:



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_2

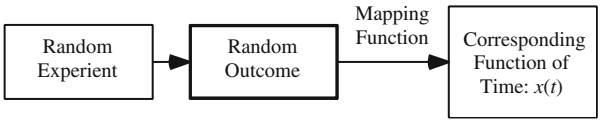



**Fig. 2.1** The sequence of events leading to assigning a time function $x(t)$ to the outcome of a random experiment

1. Discrete time, discrete value: we measure time at discrete values $t = nT$ with $n = 0, 1, 2, \cdots$. As an example, at each value of $n$ we could observe the number of cars on the road $x(n)$. In that case, $x(n)$ is an integer between 0 and 10, say. Each time we perform this experiment, we would get a totally different sequence for $x(n)$.
2. Discrete time, continuous value: we measure time at discrete values $t = nT$ with $n = 0, 1, 2, \cdots$. As an example, at each value of $n$ we measure the outside temperature $x(n)$. In that case, $x(n)$ is a real number between $-30°$ and $+45°$, say. Each time we perform this experiment, we would get a totally different sequence for $x(n)$.
3. Continuous time, discrete value: we measure time as a continuous variable $t$. As an example, at each value of $t$ we store an eight-bit digitized version of a recorded voice waveform $x(t)$. In that case, $x(t)$ is a binary number between 0 and 255, say. Each time we perform this experiment, we would get a totally different sequence for $x(t)$.
4. Continuous time, continuous value: we measure time as a continuous variable $t$. As an example, at each value of $t$ we record a voice waveform $x(t)$. In that case, $x(t)$ is a real number between 0 and 5 V, say. Each time we perform this experiment, we would get a totally different sequence for $x(t)$.

Figure 2.2 shows a discrete time, discrete value random process for an observation of ten samples where only three random functions are generated. We find that for $n = 2$, the values of the functions correspond to the random variable $X(2)$.

Therefore, random processes give rise to random variables when the time value $t$ or $n$ is fixed. This is equivalent to sampling all the random functions at the specified time value which is equivalent to taking a vertical slice from all the functions shown in Fig. 2.2.

*Example 2.1.* A time function is generated by throwing a die three consecutive throws and observing the number on the top face after each throw. Classify this random process and estimate how many sample functions are possible.

This is a discrete time, discrete value process. Each sample function will have three samples and each sample value will be from the set of integers 1–6. For

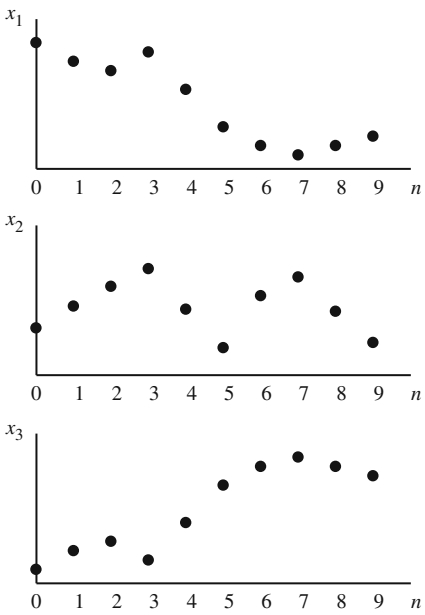



**Fig. 2.2** An example of a discrete time, discrete value random process for an observation of ten samples where only three random functions are possible

example, one sample function might be 4, 2, 5. Using the multiplication principle for probability, the total number of possible outputs is $6^3 = 216$. ■

## 2.2 Notation

We use the notation $X(t)$ to denote a continuous-time random process and also to denote the random variable measured at time $t$. When $X(t)$ is continuous, it will have a PDF $f_X(x)$ such that the probability that $x \leq X \leq x + \varepsilon$ is given by:

$$p(X = x) = f_X(x)\, dx \tag{2.1}$$

When $X(t)$ is discrete, it will have a PMF $p_X(x)$ such that the probability that $X = x$ is given by:

$$p(X = x) = p_X(x) \tag{2.2}$$

Likewise, we use the notation $X(n)$ to denote a discrete-time random process and also to denote the random variable measured at time $n$. That random variable is statistically described by a PDF $f_X(x)$ when it is continuous, or it is described by a PMF $p_X(x)$ when it is discrete.

## 2.3 Poisson Process

We shall encounter Poisson processes when we describe communication traffic. A Poisson process is a stochastic process in which the number of events occurring in a given period of time depends only on the length of the time period [2]. This number of events $k$ is represented as a random variable $K$ that has a Poisson distribution given by:

$$p(k) = \frac{(\lambda\, t)^k e^{-\lambda\, t}}{k!} \tag{2.3}$$

where $\lambda > 0$ is a constant representing the rate of arrival of the events and $t$ is the length of observation time.

## 2.4 Exponential Process

The exponential process is related to the Poisson process. The exponential process is used to model the interarrival time between occurrence of random events. Examples that lead to an interarrival time include the time between bus arrivals at a bus stop, the time between failures of a certain component; and the time between packet arrival at the input of a router.

The random variable $T$ could be used to describe the interarrival time. The probability that the interarrival time lies in the range $t \leq T \leq t + dt$ is given by:

$$p(t \leq T \leq t + dt) = \lambda e^{-\lambda\, t}\, dt \tag{2.4}$$

where $\lambda$ is the average rate of the event under consideration.

## 2.5 Deterministic and Nondeterministic Processes

A *deterministic* random process is one where future values of the sample function are known if the present value is known. An example of a deterministic random process is the modulation technique known as Quadrature Amplitude Modulation (QAM) for transmitting groups of binary data. The transmitted analog waveform is given by:

$$v(t) = a\cos(\omega t + \phi) \tag{2.5}$$

where the signal amplitude $a$ and phase angle $\phi$ change their value depending on the bit pattern that has been received. The analog signal is transmitted for the time period $0 \leq t < T_0$. Since the arriving bit pattern is random, the values of the corresponding two parameters $a$ and $\phi$ are random. However, once $a$ and $\phi$ are determined, we are able to predict the shape of the resulting waveform.

A *nondeterministic* random process is one where future values of the sample function cannot be known if the present value is known. An example of a nondeterministic random process is counting the number of packets that arrives at the input of a switch every one second and this observation is repeated for a certain time. We are unable to predict the pattern even if we know the present number of arriving packets.

## 2.6 Ensemble Average

The random variable $X(n_1)$ represents all the possible values $x$ obtained when time is frozen at the value $n_1$. In a sense, we are sampling the ensemble of random functions at this time value.

The expected value of $X(n_1)$ is called the *ensemble average* or statistical average $\mu(n_1)$ of the random process at $n_1$. The ensemble average is expressed as

$$\mu_X(t) = E[X(t)] \qquad \text{continuous-time process} \tag{2.6}$$

$$\mu_X(n) = E[X(n)] \qquad \text{discrete-time process} \tag{2.7}$$

The ensemble average could itself be another random variable since its value could change at random with our choice of the time value $t$ or $n$.

*Example 2.2.* The modulation scheme known as Frequency-Shift Keying (FSK) can be modeled as a random process described by:

$$X(t) = a \cos \omega t$$

where $a$ is a constant and $\omega$ corresponds to the random variable $\Omega$ that can have one of two possible values $\omega_1$ and $\omega_2$ that correspond to the input bit being zero or one, respectively. Assuming that the two frequencies are equally likely, find the expected value $\mu(t)$ of this process.

Our random variable $\Omega$ is discrete with probability 0.5 when $\Omega = \omega_1$ or $\Omega = \omega_2$. The expected value for $X(t)$ is given by

$$\begin{aligned} E[X(t)] &= 0.5\, a\, \cos \omega_1 t + 0.5\, a\, \cos \omega_2 t \\ &= a\, \cos\left[\frac{(\omega_1 + \omega_2)\, t}{2}\right] \times \cos\left[\frac{(\omega_1 - \omega_2)\, t}{2}\right] \end{aligned}$$

■

*Example 2.3.* The modulation scheme known as Pulse Amplitude Modulation (PAM) can be modeled as a random process described by:

$$X(n) = \sum_{i=0}^{\infty} g(n)\, \delta(n - i)$$

where $g(n)$ is the amplitude of the input signal at time $n$. $g(n)$ corresponds to the random variable $G$ that is uniformly distributed in the range 0–A. Find the expected value $\mu(t)$ of this process.

This is a discrete time, continuous value random process. Our random variable $G$ is continuous and the expected value for $X(n)$ is given by

$$E[X(n)] = \frac{1}{A}\int_0^A g\, dg$$
$$= \frac{A}{2}$$

■

## 2.7 Time Average

Figure 2.2 helps us find the *time average* of the random process. The time average is obtained by finding the average value for *one* sample function such as $X_1(n)$ in the figure. The time average is expressed as

$$\overline{X} = \frac{1}{T}\int_0^T X(t)\, dt \qquad \text{continuous-time process} \tag{2.8}$$

$$\overline{X} = \frac{1}{N}\sum_0^{N-1} X(n)] \qquad \text{discrete-time process} \tag{2.9}$$

In either case we assumed we sampled the function for a period $T$ or we observed $N$ samples.

The time average $\overline{X}$ could itself be a random variable since its value could change with our choice of the random function under consideration.

## 2.8 Autocorrelation Function

Assume a discrete time random process $X(n)$ which produces two random variables $X_1 = X(n_1)$ and $X_2 = X(n_2)$ at times $n_1$ and $n_2$, respectively. The *autocorrelation function* for these two random variables is defined by the following equation:

$$r_{XX}(n_1, n_2) = E[X_1\, X_2] \tag{2.10}$$

In other words, we consider the two random variables $X_1$ and $X_2$ obtained from the same random process at the two different time instances $n_1$ and $n_2$.

*Example 2.4.* Find the autocorrelation function for a second-order Finite-Impulse Response (FIR) digital filter, sometimes called Moving Average (MA) filter, whose output is given by the equation:

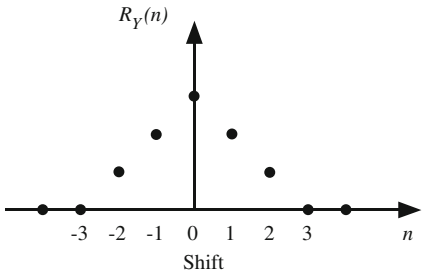



**Fig. 2.3** Autocorrelation function of a second-order digital filter whose input is uncorrelated samples

$$y(n) = a_0 x(n) + a_1 x(n-1) + a_2 x(n-2) \tag{2.11}$$

where the input samples $x(n)$ are assumed to be zero mean Independent and Identically Distributed (IID) random variables.

We assign the random variable $Y_n$ to correspond to output sample $y(n)$ and $X_n$ to correspond to input sample $x(n)$. Thus we can have the following autocorrelation function:

$$r_{YY}(0) = E[Y_n Y_n] = a_0^2 E\left[X_0^2\right] + a_1^2 E\left[X_1^2\right] + a_2^2 E\left[X_2^2\right] E\left[X_0^2\right] \tag{2.12}$$

$$= \left(a_0^2 + a_1^2 + a_2^2\right)\sigma^2 \tag{2.13}$$

Similarly we can write

$$r_{YY}(1) = E\left(Y_n Y_{n+1}\right) = 2a_0 a_1\, \sigma^2 \tag{2.14}$$

$$r_{YY}(2) = E\left(Y_n Y_{n+2}\right) = a_0 a_2\, \sigma^2 \tag{2.15}$$

$$r_{YY}(k) = 0; \qquad k > 2 \tag{2.16}$$

where $\sigma^2$ is the input sample variance. Figure 2.3 shows a plot of the autocorrelation assuming all the filter coefficients are equal. ■

## 2.9 Stationary Processes

A *wide-sense stationary* random process has the following two properties [3]

$$E[X(t)] = \mu = \text{constant} \tag{2.17}$$

$$E[X(t)\, X(t+\tau)] = r_{XX}(t, t+\tau) = r_{XX}(\tau) \tag{2.18}$$

Such a process has a constant expected value and the autocorrelation function depends on the time difference between the two random variables.

The above equations apply to a continuous-time random process. For a discrete time random process, the equations for a wide-sense stationary random process become

$$E\left[X(n)\right] = \mu = \text{constant} \tag{2.19}$$

$$E\left[X\left(n_1\right)\ X\left(n_1+n\right)\right] = r_{XX}\left(n_1, n_1+n\right) = r_{XX}(n) \tag{2.20}$$

The autocorrelation function for a wide-sense stationary random process exhibits the following properties [1].

$$r_{XX}(0) = E\left[X^2(n)\right] \geq 0 \tag{2.21}$$

$$\left|r_{XX}(n)\right| \leq r_{XX}(0) \tag{2.22}$$

$$r_{XX}(-n) = r_{XX}(n) \qquad \text{even symmetry} \tag{2.23}$$

A stationary random process is *ergodic* if all time averages equal their corresponding statistical averages [3]. Thus if $X(n)$ is an ergodic random process, then we could write

$$\overline{X} = \mu \tag{2.24}$$

$$\overline{X^2} = r_{XX}(0) \tag{2.25}$$

*Example 2.5.* The modulation scheme known as Phase-Shift Keying (PSK) can be modeled as a random process described by:

$$X(t) = a\cos(\omega t + \phi)$$

where $a$ and $\omega$ are constant and $\phi$ corresponds to the random variable $\Phi$ with two values 0 and $\pi$ which are equally likely. Find the autocorrelation function $r_{XX}(t)$ of this process.

The phase PMF is given by

$$p(0) = 0.5$$

$$p(\pi) = 0.5$$

The autocorrelation is found as

$$\begin{aligned} r_{XX}(\tau) &= E\left[a\cos(\omega t + \Phi)\ a\cos(\omega t + \omega\tau + \Phi)\right] \\ &= 0.5\,a^2\ \cos(\omega\tau)\ E\left[\cos(2\omega t + \omega\tau + 2\Phi)\right] \\ &= 0.5\,a^2\ \cos(\omega\tau)\ \cos(2\omega t + \omega\tau) \end{aligned}$$

We notice that this process is not wide-sense stationary since the autocorrelation function depends on $t$. ■

## 2.10 Cross-Correlation Function

Assume two discrete time random processes $X(n)$ and $Y(n)$ which produce two random variables $X_1 = X(n_1)$ and $Y_2 = Y(n_2)$ at times $n_1$ and $n_2$, respectively. The *cross-correlation function* is defined by the following equation:

$$r_{XY}(n_1, n_2) = E[X_1 Y_2] \tag{2.26}$$

If the cross-correlation function is zero, i.e., $r_{XY} = 0$ then we say that the two processes are *orthogonal*.

If the two processes are *statistically independent*, then we have:

$$r_{XY}(n_1, n_2) = E[X(n_1)] \times E[Y(n_2)] \tag{2.27}$$

*Example 2.6.* Find the cross-correlation function for the two random processes:

$$X(t) = a \cos \omega t$$
$$Y(t) = b \sin \omega t$$

where $a$ and $b$ are two IID random variables with mean $\mu$ and variance $\sigma^2$.

The cross-correlation function is given by:

$$\begin{aligned} r_{XY}(t, t+\tau) &= E[A \cos \omega t \; B \sin(\omega t + \omega \tau)] \\ &= 0.5[\sin \omega \tau + \sin(2\omega t + \omega \tau)] \; E[A] \; E[B] \\ &= 0.5 \, \mu^2 [\sin \omega \tau + \sin(2\omega t + \omega \tau)] \end{aligned}$$

■

## 2.11 Covariance Function

Assume a discrete time random process $X(n)$ which produces two random variables $X_1 = X(n_1)$ and $X_2 = X(n_2)$ at times $n_1$ and $n_2$, respectively. The *autocovariance function* is defined by the following equation:

$$c_{XX}(n_1, n_2) = E[(X_1 - \mu_1)(X_2 - \mu_2)] \tag{2.28}$$

The autocovariance function is related to the autocorrelation function by the following equation:

$$c_{XX}(n_1, n_2) = r_X(n_1, n_2) - \mu_1 \mu_2 \tag{2.29}$$

For a wide-sense stationary process, the autocovariance function depends on the difference between the time indices $n = n_2 - n_1$:

$$c_{XX}(n) = E\left[(X_1 - \mu)(X_2 - \mu)\right] = r_{XX}(n) - \mu^2 \tag{2.30}$$

*Example 2.7.* Find the autocovariance function for the random process $X(t)$ given by:

$$X(t) = a + b\cos\omega t$$

where $\omega$ is a constant and $a$ and $b$ are IID random variables with zero mean and variance $\sigma^2$.

We have:

$$\begin{aligned} c_{XX} &= E\left\{(A + B\cos\omega t)[A + B\cos\omega(t+\tau)]\right\} \\ &= E\left[A^2\right] + E[AB]\ [\cos\omega t + \cos\omega(t+\tau)] + E\left[B^2\right]\cos^2\omega(t+\tau) \\ &= \sigma^2 + E[A]\ E[B]\ [\cos\omega t + \cos\omega(t+\tau)] + \sigma^2\cos^2\omega(t+\tau) \\ &= \sigma^2\left[1 + \cos^2\omega(t+\tau)\right] \end{aligned}$$

■

The *cross-covariance function* for two random processes $X(n)$ and $Y(n)$ is defined by

$$\begin{aligned} c_{XY}(n) &= E\left[(X(n_1) - \mu_X)(Y(n_1 + n) - \mu_Y)\right] \\ &= r_{XY}(n) - \mu_X\mu_Y \end{aligned} \tag{2.31}$$

Two random processes are called *uncorrelated* when their cross-covariance function vanishes.

$$c_{XY}(n) = 0 \tag{2.32}$$

*Example 2.8.* Find the cross-covariance function for the two random processes $X(t)$ and $Y(t)$ given by

$$X(t) = a + b\cos\omega t$$

$$Y(t) = a + b\sin\omega t$$

where $\omega$ is a constant and $a$ and $b$ are IID random variables with zero mean and variance $\sigma^2$.

We have

$$\begin{aligned} c_{XY}(n) &= E\left\{(A + B\cos\omega t)[A + B\sin\omega(t+\tau)]\right\} \\ &= E\left[A^2\right] + E[AB]\ [\cos\omega t + \sin\omega(t+\tau)] + E\left[B^2\right]\cos\omega t\ \sin\omega(t+\tau) \\ &= \sigma^2 + E[A]\ E[B]\ [\cos\omega t + \sin\omega(t+\tau)] + \sigma^2\cos\omega t\ \sin\omega(t+\tau) \\ &= \sigma^2\left[1 + \cos\omega t\ \sin\omega(t+\tau)\right] \end{aligned}$$

■

## 2.12 Correlation Matrix

Assume we have a discrete time random process $X(n)$. At each time step $i$ we define the random variable $X_i = X(i)$. If each sample function contains $n$ components, it is convenient to construct a vector representing all these random variables in the form:

$$\mathbf{x} = \begin{bmatrix} X_1 & X_2 & \cdots & X_n \end{bmatrix}^t \tag{2.33}$$

Now we would like to study the correlation between each random variable $X_i$ and all the other random variables. This would give us a comprehensive understanding of the random process. The best way to do that is to construct a *correlation matrix*.

We define the $n \times n$ correlation matrix $\mathbf{R}_X$, which gives the correlation between all possible pairs of the random variables, as:

$$\mathbf{R}_X = E\left[\mathbf{x}\,\mathbf{x}^t\right] = E\begin{bmatrix} X_1X_1 & X_1X_2 & \cdots & X_1X_n \\ X_2X_1 & X_2X_2 & \cdots & X_2X_n \\ \vdots & \vdots & \ddots & \vdots \\ X_nX_1 & X_nX_2 & \cdots & X_nX_n \end{bmatrix} \tag{2.34}$$

We can express $\mathbf{R}_X$ in terms of the individual correlation functions:

$$\mathbf{R}_X = \begin{bmatrix} r_{XX}(1,1) & r_{XX}(1,2) & \cdots & r_{XX}(1,n) \\ r_{XX}(1,2) & r_{XX}(2,2) & \cdots & r_{XX}(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ r_{XX}(1,n) & r_{XX}(2,n) & \cdots & r_{XX}(n,n) \end{bmatrix} \tag{2.35}$$

Thus we see that the correlation matrix is symmetric. For a wide-sense stationary process, the correlation functions depend only on the difference in times and we get an even simpler matrix structure:

$$\mathbf{R}_X = \begin{bmatrix} r_{XX}(0) & r_{XX}(1) & \cdots & r_{XX}(n-1) \\ r_{XX}(1) & r_{XX}(0) & \cdots & r_{XX}(n-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_{XX}(n-1) & r_{XX}(n-2) & \cdots & r_{XX}(0) \end{bmatrix} \tag{2.36}$$

Each diagonal in this matrix has identical elements and our correlation matrix becomes a *Toeplitz matrix*.

*Example 2.9.* Assume the autocorrelation function for a stationary random process is given by:

$$r_{XX}(\tau) = 5 + 3e^{-|\tau|}$$

Find the autocorrelation matrix for $\tau = 0$, 1, and 2.

The autocorrelation matrix is given by:

$$\mathbf{R}_{XX} = \begin{bmatrix} 8 & 6.1036 & 5.4060 \\ 6.1036 & 8 & 6.1036 \\ 5.4060 & 6.1036 & 6 \end{bmatrix}$$

■

## 2.13 Covariance Matrix

In a similar fashion, we can define the *covariance matrix* for many random variables obtained from the same random process as:

$$\mathbf{C}_{XX} = E\left[(\mathbf{x} - \overline{\mu})\,(\mathbf{x} - \overline{\mu})^t\right] \tag{2.37}$$

where $\overline{\mu} = \left[\mu_1\ \mu_2 \cdots \mu_n\right]^t$ is the vector whose components are the expected values of our random variables. Expanding the above equation we can write:

$$\mathbf{C}_{XX} = E\left[\mathbf{X}\mathbf{X}^{\mathbf{t}}\right] - \overline{\mu}\,\overline{\mu}^t \tag{2.38}$$

$$= \mathbf{R}_X - \overline{\mu}\,\overline{\mu}^t \tag{2.39}$$

When the process has zero mean, the covariance matrix equals the correlation matrix:

$$\mathbf{C}_{XX} = \mathbf{R}_{XX} \tag{2.40}$$

The covariance matrix can be written explicitly in the form:

$$\mathbf{C}_{XX} = \begin{bmatrix} C_{XX}(1,1) & C_{XX}(1,2) & \cdots & C_{XX}(1,n) \\ C_{XX}(1,2) & C_{XX}(2,2) & \cdots & C_{XX}(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ C_{XX}(1,n) & C_{XX}(2,n) & \cdots & C_{XX}(n,n) \end{bmatrix} \tag{2.41}$$

Thus we see that the covariance matrix is symmetric. For a wide-sense stationary process, the covariance functions depend only on the difference in times and we get an even simpler matrix structure:

$$\mathbf{C}_{XX} = \begin{bmatrix} C_{XX}(0) & C_{XX}(1) & \cdots & C_{XX}(n-1) \\ C_{XX}(1) & C_{XX}(0) & \cdots & C_{XX}(n-2) \\ \vdots & \vdots & \ddots & \vdots \\ C_{XX}(n-1) & C_{XX}(n-2) & \cdots & C_{XX}(0) \end{bmatrix} \tag{2.42}$$

Using the definition for covariance in (1.118) on page 36, we can write the above equation as:

$$\mathbf{C}_{XX} = \sigma_X^2 \begin{bmatrix} 1 & \rho(1) & \rho(2) & \cdots & \rho(n-1) \\ \rho(1) & 1 & \rho(1) & \cdots & \rho(n-2) \\ \rho(2) & \rho(1) & 1 & \cdots & \rho(n-3) \\ \vdots & & & & \vdots \\ \rho(n-1) & \rho(n-2) & \rho(n-3) & \cdots & 1 \end{bmatrix} \tag{2.43}$$

*Example 2.10.* Assume the autocovariance function for a wide-sense stationary random process is given by:

$$c_{XX}(\tau) = 5 + 3e^{-|\tau|}$$

Find the autocovariance matrix for $\tau = 0$, 1, and 2.

Since the process is wide-sense stationary, the variance is given by:

$$\sigma^2 = c_{XX}(0) = 8$$

The autocovariance matrix is given by:

$$\mathbf{C}_{XX} = 8 \begin{bmatrix} 1 & 0.7630 & 0.6758 \\ 0.7630 & 1 & 0.7630 \\ 0.6758 & 0.7630 & 1 \end{bmatrix}$$

■

## 2.14 Problems

### *2.14.1 Random Processes*

**2.1.** Define deterministic and nondeterministic random processes. Give an example of each type.

**2.2.** Let $X$ be the random process corresponding to observing the temperature throughout a day. The number of sample functions is 365 corresponding to each day of the year. Classify this process.

**2.3.** Let $X$ be the random process corresponding to observing the number of defective lights in a building versus time for a period of one month. Each month we would get a different pattern. Classify this process.

**2.4.** Let $X$ be the random process corresponding to measuring the total tonnage (weight) of ships going through the Suez canal in one day. The data is plotted for a period of one year. Each year will produce a different pattern. Classify this process.

**2.5.** Let $X$ be the random process corresponding to observing the number of cars crossing a busy intersection in one hour. The number of sample functions is 24 corresponding to each hour of the day. Classify this process.

**2.6.** Let $X$ be the random process corresponding to observing the bit pattern in an ATM cell. Classify this process.

### *2.14.2 Expected Value*

**2.7.** Amplitude-Shift Keying (ASK) can be modeled as a random process described by

$$X(t) = a \cos \omega t$$

where $\omega$ is constant and $a$ corresponds to the random variable $A$ with two values 0 and $a_0$ which occur with equal probability. Find the expected value $\mu(t)$ of this process.

**2.8.** A modified ASK uses two bits of the incoming data to generate a sinusoidal waveform and the corresponding random process is described by

$$X(t) = a \cos \omega t$$

where $\omega$ is a constant and $a$ is a random variable with four values $a_1$, $a_2$, $a_3$, and $a_4$. Assuming that the four possible bit patterns are equally likely find the expected value $\mu(t)$ of this process.

**2.9.** PSK can be modeled as a random process described by

$$X(t) = a \cos(\omega t + \phi)$$

where $a$ and $\omega$ are constant and $\phi$ corresponds to the random variable $\Phi$ with two values 0 and $\pi$ which occur with equal probability. Find the expected value $\mu(t)$ of this process.

**2.10.** A modified FSK uses two bits of the incoming data to generate a sinusoidal waveform and the corresponding random process is described by

$$X(t) = a \cos \omega t$$

where $a$ is a constant and $\omega$ is a random variable with four values $\pi/4$, $3\pi/4$, $-3\pi/4$, and $-\pi/4$ [4]. Assuming that the four possible bit patterns occur with equal probability find the expected value $\mu(t)$ of this process.

**2.11.** A modified FSK uses three bits of the incoming data to generate a sinusoidal waveform and the random process is described by

$$X(t) = a\cos\omega t$$

where $a$ is a constant and $\omega$ corresponds to the random variable $\Omega$ with four values $\omega_1, \omega_2, \cdots, \omega_8$. Assuming that the eight frequencies are equally likely find the expected value $\mu(t)$ of this process.

**2.12.** A deterministic discrete-time random process $X(n)$ produces the random variable $X(n)$ given by

$$X(n) = a^n$$

where $a$ is a uniformly distributed random variable in the range 0–1. Find the expected value for this random variable at any time instant $n$.

### *2.14.3 Autocorrelation Function*

**2.13.** Define a wide-sense stationary random process.

**2.14.** Prove (2.23) on page 58.

**2.15.** Define an ergodic random process.

**2.16.** Explain which of the following functions represent a valid autocorrelation function.

$r_{XX}(n) = a^n \quad 0 \le a < 1$ $\qquad$ $r_{XX}(n) = |a|^n \quad 0 \le a < 1$

$r_{XX}(n) = a^{n^2} \quad 0 \le a < 1$ $\qquad$ $r_{XX}(n) = |a|^{n^2} \quad 0 \le a < 1$

$r_{XX}(n) = \cos n$ $\qquad$ $r_{XX}(n) = \sin n$

**2.17.** A random process described by

$$X(t) = a\cos(\omega t + \phi)$$

where $a$ and $\omega$ are constant and $\phi$ corresponds to the random variable $\Phi$ is uniformly distributed in the interval 0–$2\pi$. Find the autocorrelation function $r_{XX}(t)$ of this process.

### *2.14.4 Cross-Correlation Function*

**2.18.** Define what is meant by two random processes being orthogonal.

**2.19.** Define what is meant by two random processes being statistically independent.

**2.20.** Find the cross-correlation function for the following random process and its delayed version

$$X(t) = a\cos\omega t$$
$$Y(t) = \alpha\, a\cos(\omega t + \theta)$$

where $a$ $\theta$ are zero two mean random variable and $\alpha$ is a constant.

### *2.14.5 Covariance Function*

**2.21.** Given two random processes $X$ and $Y$, when are they uncorrelated?

**2.22.** Write down expressions for the cross-correlation and cross-covariance for two uncorrelated random processes.

**2.23.** Find the covariance function for the random process

$$X(t) = a\cos\omega t$$

where $a$ is a random variable with mean $\mu$ and variance $\sigma^2$.

**2.24.** Find the autocovariance function for the random process in Example 2.7 on page 60 when $\omega$ is a uniformly distributed random variable in the range 0–$\omega_0$ and $a$ and $b$ are IID random variables with zero mean and $\sigma^2$ variance.

**2.25.** Find the autocovariance function for the random process in Example on page 60 when $\omega$ is a uniformly distributed random variable in the range 0–$\omega_0$ and $a$ and $b$ are IID random variables with zero mean and $\sigma^2$ variance.

## References

1. R.D. Yates, D.J. Goodman, *Probability and Stochastic Processes* (Wiley, New York, 1999)
2. Thesaurus of Mathematics. http://thesaurus.maths.org/dictionary/map/word/1656
3. P.Z. Peebles, *Probability, Random Variables, and Random Signal Principles* (McGraw-Hill, New York, 1993)
4. S. Haykin, *An Introduction to Analog and Digital Communications* (Wiley, New York, 1989)

# Chapter 3
# Markov Chains

## 3.1 Introduction

We explained in Chap. 1 that in order to study a stochastic system we map its random output to one or more random variables. In Chap. 2 we studied other systems where the output was mapped to random processes which are functions of time. In either case we characterized the system using the expected value, variance, correlation, and covariance functions. In this chapter we study stochastic systems that are best described using *Markov processes*. A Markov process is a random process where the value of the random variable at instant $n$ depends *only* on its immediate past value at instant $n - 1$. The way this dependence is defined gives rise to a family of sample functions just like in any other random process. In a Markov process the random variable represents the *state* of the system at a given instant $n$. The state of the system depends on the nature of the system under study as we shall see in that chapter. We will have a truly rich set of parameters that describe a Markov process. This will be the topic of the next few chapters.

We see examples of Markov processes in many real-life situations

1. Telecommunication protocols and hardware systems.
2. Customer arrivals and departures at banks.
3. Checkout counters at supermarkets.
4. Mutation of a virus or DNA molecule.
5. Random walk such as Brownian motion.
6. Arrival of cars at an intersection.
7. Bus rider population during the day, week, month, etc.
8. Machine breakdown and repair during use.
9. The state of the daily weather.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_3

## 3.2 Markov Chains

If the state space of a Markov process is discrete, the Markov process is called a *Markov chain*. In that case the states are labeled by the integers 0, 1, 2, and so on. We will be concerned here with discrete-time Markov chains since they arise naturally in many communication systems.

## 3.3 Selection of the Time Step

A Markov chain stays in a particular state for a certain amount of time called the *hold time*. At the end of the hold time, the Markov chain moves to another state at random where the process repeats again. We have two broad classifications of Markov chains that are based on how we measure the hold time.

**Discrete-Time Markov Chain** In a discrete-time Markov chain the hold time assumes discrete values. As a result, changes in the states occur at discrete time values. In that case time is measured at specific instances:

$$t = T_0,\ T_1,\ T_2,\ \cdots$$

The spacing between the time steps need not be equal in the general case. Most often, however, the discrete time values are equally spaced and we can write

$$t = nT \tag{3.1}$$

$$n = 0, 1, 2, \cdots \tag{3.2}$$

The time step value $T$ depends on the system under study as will be explained below.

**Continuous-Time Markov Chain** In a continuous-time Markov chain the hold time assumes continuous values. As a result, changes in the states occur at any time value. The time value $t$ will be continuous over a finite or infinite interval.

### *3.3.1 Discrete-Time Markov Chains*

This type of Markov chains changes state at regular intervals. The time step could be a clock cycle, start of a new day, or a year, etc.

*Example 3.1.* Consider a packet buffer where packets arrive at each time step with probability $a$ and depart with probability $c$. Identify the Markov chain and specify the possible buffer states.

**Table 3.1** States of buffer occupancy

| State | Significance |
|---|---|
| 0 | Buffer is empty |
| 1 | Buffer has one packet |
| 2 | Buffer has two packets |
| ⋮ | ⋮ |
| B | Buffer has B packets (full) |

We choose the time step in this example to be equal to the time required to receive or transmit a packet (transmission delay). At each time step we have two independent events: packet arrival and packet departure. We model the buffer as a Markov chain where the states of the system indicate the number of packets in the buffer. Assuming the buffer size is $B$, then the number of states of the buffer is $B+1$ as identified in Table 3.1. ■

*Example 3.2.* Suppose that packets arrive at random on the input port of a router at an average rate $\lambda_a$ (packets/s). The maximum data rate is assumed to be $\sigma$ (packets/s), where $\sigma > \lambda_a$. Study the packet arrival statistics if the port input is sampled at a rate equal to the average data rate $\lambda_a$.

The time step (seconds) is chosen as:

$$T = \frac{1}{\lambda_a}$$

In one time step we could receive 0, 1, 2, $\cdots$, $N$ packets where $N$ is the maximum number of packets that could arrive:

$$N = \lceil \sigma \times T \rceil = \lceil \frac{\sigma}{\lambda_a} \rceil$$

where the ceiling function $f(x) = \lceil x \rceil$ gives the smallest integer larger than or equal to $x$.

The statistics for packet arrival follow the binomial distribution and the probability of receiving $k$ packets in time $T$ is:

$$p(k) = \binom{N}{k} a^k \; b^{N-k}$$

where $a$ is the probability that a packet arrives and $b = 1 - a$. Our job in almost all situations will be to find out the values of the parameters $N$, $a$ and $b$ in terms of the given data rates.

The packet arrival probability $a$ could be obtained using the average value of the binomial distribution. The average input traffic $N_a(in)$ is given from the binomial distribution by:

$$N_a(in) = N \; a$$

But $N_a(in)$ is also determined by the average data rate as:

$$N_a(in) = \lambda_a \times T = 1$$

From the above two equations we get:

$$a = \frac{1}{N} \leq \frac{\lambda_a}{\sigma}$$ ■

*Example 3.3.* Consider the previous Example 3.2 when the input port is sampled at the rate $\sigma$.

The time step is now given by:

$$T' = \frac{1}{\sigma}$$

In one time step we either get one packet or we get no packets. There is no chance to get more than one packet in one time step since packets cannot arrive at a rate higher than $\sigma$. Therefore, the packet arrival statistics follow the Bernoulli distribution.

For a time period $t$, the average number of packets that arrives is:

$$N_a(in) = \lambda_a \, t$$

From the Bernoulli distribution that average is given by:

$$N_a(in) = a' \, \frac{t}{T'}$$

The fraction on RHS indicates the number of time steps spanning the time period $t$. From the above two equations we get:

$$a' = \lambda_a \, T' = \frac{\lambda_a}{\sigma}$$ ■

## 3.4 Memoryless Property of Markov Chains

In a discrete-time Markov chain, the value of the random variable $S(n)$ represents the state of the system at time $n$. The random variable $S(n)$ is a function of its immediate past value—i.e., $S(n)$ depends on $S(n-1)$. This is referred to as the *Markov property* or *memoryless property* of the Markov chain where the present state of the system depends only on its immediate past state [4, 5]. Alternatively, we can say that the Markov property of the Markov chain implies that the future state of the system depends only on the present sate and not on its past states [3].

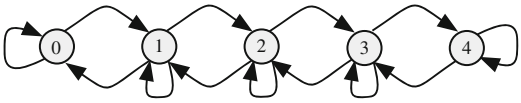



**Fig. 3.1** The occupancy states of a buffer of size four and the possible transitions between the states

The probability that the Markov chain is in state $s_i$ at time $n$ is a function of its past state $s_j$ at time $n-1$ only. Mathematically this statement is written as:

$$p\left[S(n) = s_i\right] = f\left(s_j\right) \tag{3.3}$$

for all $i \in S$ and $j \in S$ where $S$ is the set of all possible states of the system.

Transition from a state to the next state is determined by a *transition probability* only with no regard to how the system came to be in the present state. Many communication systems can be modeled as Markov or memoryless systems using several techniques such as introducing extra transitions, defining extra states, and adjusting the time step value. This effort is worthwhile since the memoryless property of a Markov chain will result in a *linear system* that can be easily studied.

*Example 3.4.* Consider a data buffer in a certain communication device such as a network router for example. Assume the buffer could accommodate at most four packets. We say the buffer size is $B = 4$. Identify the states of this buffer and show the possible transitions between states assuming at any time step at most one packet can arrive or leave the buffer. Finally explain why the buffer could be studied using Markov chain analysis.

Figure 3.1 shows the occupancy states of a buffer of size four and the possible transitions between the states. The buffer could be empty or it could contain 1, 2, 3, or 4 packets. Furthermore, the assumptions indicate that the size of the buffer could remain unchanged or it could increase or decrease by one.

The transition from one state to another does not depend on how the buffer happened to be in the present state. Thus the system is memoryless and could be modeled as a Markov chain. ■

## 3.5 Markov Chain Transition Matrix

Let us define $p_{ij}(n)$ as the probability of finding our system in state $i$ at time step $n$ given that the past state was state $j$. We equate $p_{ij}$ to the conditional probability that the system is in state $i$ at time $n$ given that it was in state $j$ at time $n-1$. Mathematically, we express that statement as follows:

$$p_{ij}(n) = p\left[S(n) = i \mid S(n-1) = j\right] \tag{3.4}$$

The situation is further simplified if the transition probability is independent of the time step index $n$. In that case we have a *homogeneous* Markov chain and the above equation becomes:

$$p_{ij} = p\,[S(n) = i \mid S(n-1) = j\,] \tag{3.5}$$

Let us define the probability of finding our system in state $i$ at the $n$th step as:

$$s_i(n) = p\,[X(n) = i] \tag{3.6}$$

where the subscript $i$ identifies the state and $n$ denotes the time step index.

Using (3.4), we can express the above equation as:

$$s_i(n) = \sum_{j=1}^{m} p_{ij} \times s_j(n-1) \tag{3.7}$$

where we assumed the number of possible states to be $m$ and the indices $i$ and $j$ lie in the range $1 \leq i \leq m$ and $1 \leq j \leq m$. We can express the above equation in matrix form as:

$$\mathbf{s}(n) = \mathbf{P}\,\mathbf{s}(n-1) \tag{3.8}$$

where $\mathbf{P}$ is the *state transition matrix* of dimension $m \times m$:

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1,m} \\ p_{21} & p_{22} & \cdots & p_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m,1} & p_{m,2} & \cdots & p_{m,m} \end{bmatrix} \tag{3.9}$$

and $\mathbf{s}(n)$ is the *distribution vector* (or *state vector*) defined as the probability the system being in each state at time step $n$:

$$\mathbf{s}(n) = \begin{bmatrix} s_1(n) & s_2(n) & \cdots & s_m(n) \end{bmatrix}^t \tag{3.10}$$

The component $s_i(n)$ of the distribution vector $\mathbf{s}(n)$ at time $n$ indicates the *probability* of finding our system in state $s_i$ at that time. Because it is a probability, our system could be in any of the $m$ states. However, the probabilities only indicate the likelihood of being in a particular state. Because $\mathbf{s}$ describes probabilities of all possible $m$ states, we must have:

$$\sum_{i=1}^{m} s_i(n) = 1 \qquad n = 0, 1, 2, \cdots \tag{3.11}$$

We say that our vector is normalized when it satisfies (3.11). We call such a vector a *distribution vector*. This is because the vector describes the distribution of probabilities among the different states of the system.

Soon we shall find out that describing the transition probabilities in matrix form leads to great insights about the behavior of the Markov chain. To be specific, we will find that we are interested in more than finding the values of the transition probabilities or entries of the transition matrix **P**. Rather, we will pay great attention to the eigenvalues and eigenvectors of the transition matrix.

Since the columns of **P** represent transitions *out of* a given state, the sum of each column must be one since this covers all the possible transition events out of the state. Therefore we have, for all values of $j$,:

$$\sum_{i=1}^{m} p_{ij} = 1 \tag{3.12}$$

The above equation is always valid since the sum of each column in **P** is unity. For example, a $2 \times 2$ transition matrix **P** would be set up as in the following diagram:

$$\begin{array}{ccc} & & \text{Present State} \\ & & \begin{array}{cc} 1 & \quad 2 \end{array} \\ & & \downarrow \\ \text{Next State} & \leftarrow & \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} \end{array} \begin{array}{l} \\ \\ \\ 1 \\ 2 \end{array}$$

The columns represent the present state while the rows represent the next state. Element $p_{ij}$ represents the transition probability from state $j$ to state $i$. For example, $p_{12}$ is the probability that the system makes a transition from state $s_2$ to state $s_1$.

*Example 3.5.* An on–off source is often used in telecommunications to simulate voice traffic. Such a source has two states: The silent state $s_1$ where the source does not send any data packets and the active state $s_2$ where the source sends one packet per time step. If the source were in $s_1$, it has a probability $s$ of staying in that state for one more time step. When it is in state $s_2$, it has a probability $a$ of staying in that state. Obtain the transition matrix for describing that source.

The next state of the source depends only on its present state. Therefore, we can model the state of the source as a Markov chain. The state diagram for such source is shown in Fig. 3.2 and the transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix} s & 1-a \\ 1-s & a \end{bmatrix}$$

■

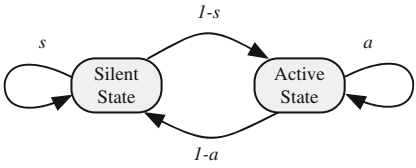



**Fig. 3.2** Transition diagram for an on–off source

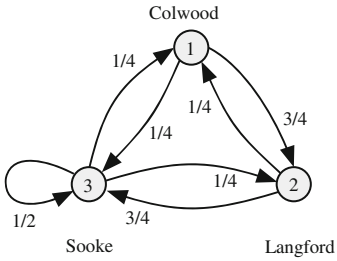



**Fig. 3.3** A Markov chain involving three states

*Example 3.6.* Assume that the probability that a delivery truck moves between three cities at the start of each day is shown in Fig. 3.3. Write down the transition matrix and the initial distribution vector assuming that the truck was initially in Langford.

We assume the city at which the truck is located is the state of the truck. The next state of the truck depends only on its present state. Therefore, we can model the state of the truck as a Markov chain. We have to assign indices to replace city names. We chose the following arbitrary assignment, although any other state assignment will work as well.

| City | State index |
|---|---|
| Colwood | 1 |
| Langford | 2 |
| Sooke | 3 |

Based on the state assignment table, the transition matrix is given by:

$$\mathbf{P} \begin{bmatrix} 0 & 1/4 & 1/4 \\ 3/4 & 0 & 1/4 \\ 1/4 & 3/4 & 1/2 \end{bmatrix}$$

The initial distribution vector is given by:

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^t$$

■

*Example 3.7.* Assume an on–off data source that generates equal length packets with probability $a$ per time step. The channel introduces errors in the transmitted packets such that the probability of a packet is received in error is $e$. Model the

source using Markov chain analysis. Draw the Markov chain state transition diagram and write the equivalent state transition matrix.

The Markov chain model of the source we use has four states:

| State | Significance |
|---|---|
| 1 | Source is idle |
| 2 | Source is retransmitting a frame |
| 3 | Frame is transmitted with no errors |
| 4 | Frame is transmitted with an error |

Since the next state of the source depends only on its present state, we can model the source using Markov state analysis.

Figure 3.4 shows the Markov chain state transition diagram. We make the following observations:

- The source stays idle (state $s_1$) with probability $1 - a$.
- Transition from $s_1$ to $s_3$ occurs under two conditions: the source is active and no errors occur during transmission.
- Transition from $s_1$ to $s_4$ occurs under two conditions: the source is active and an error occurs during transmission.
- Transition from $s_2$ to $s_3$ occurs under only one condition: no errors occur.

The associated transition matrix for the system is given by:

$$\mathbf{P} = \begin{bmatrix} 1-a & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ a(1-e) & 1-e & 0 & 0 \\ a\,e & e & 0 & 0 \end{bmatrix}$$

■

*Example 3.8.* In an ethernet network based on the carrier sense multiple access with collision detection (CSMA/CD), a user requesting access to the network starts

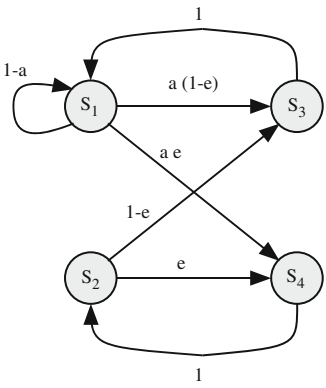



**Fig. 3.4** State transition diagram for transmitting a packet over a noisy channel

transmission when the channel is not busy. If the channel is not busy, the user starts transmitting. However, if one or more users sense that the channel is free, they will start transmitting and a collision will take place. If a collision from other users is detected, the user stops transmitting.

Assume the following probabilities

| | |
|---|---|
| $u_0$ | Probability all users are idle |
| $u_1$ | Probability only one user is transmitting |
| $1 - u_0 - u_1$ | Probability two or more users are transmitting |

(a) Justify using Markov chain analysis to describe the behavior of the channel or communication medium.
(b) Select a time step size for a discrete-time Markov chain model.
(c) Draw the Markov state transition diagram for this channel and show the state transition probabilities.

(a) A user in that system will determine its state within a time frame of twice the propagation delay on the channel. Therefore, the current state of the channel or communication medium and all users will depend only on the actions of the users in time frame of one propagation delay only. Thus our system can be described as a Markov chain.
(b) The time step $T$ we can choose is twice the propagation delay. Assume packet transmission delay requires $n$ time steps where all packets are assumed to have equal lengths.
(c) The channel can be in one of the following states:

1. $i$: Idle state
2. $t$: Transmitting state
3. $c$: Collided state

Figure 3.5 shows the state of the user and the transition probabilities between these states.

The channel remains in the idle state with probability $u_0$.

The channel moves from idle to transmitting state with probability $u_1$.

The channel moves from idle to the collided state with probability $1 - u_0 - u_1$.

The other transitions are explained in the same way. We organize our state transition matrix such that first row or column corresponds to the idle state $i$. The second row or column corresponds to the collided state. The third row or column corresponds to transmit state $t_1$, and so on. For $n$ transmit states, the transition matrix

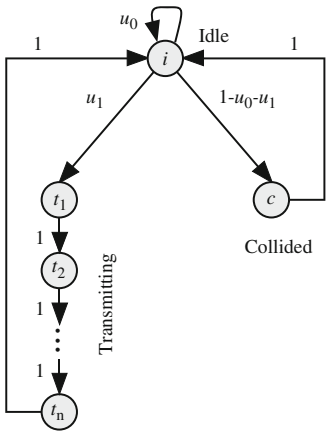



**Fig. 3.5** Markov chain state transition diagram for the CSMA/CD channel

will have the dimension $(n+2) \times (n+2)$:

$$\mathbf{P} = \begin{bmatrix} u_0 & u_0 & 0 & 0 \cdots & u_0 \\ 1-u_0-u_1 & 1-u_0-u_1 & 0 & 0 \cdots & 1-u_0-u_1 \\ u_1 & u_1 & u_1 & 0 \cdots & u_1 \\ 0 & 0 & 0 & 1 \cdots & 0 \\ 0 & 0 & 0 & 0 \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \ddots & \vdots \\ 0 & 0 & 0 & 0 \cdots & 0 \end{bmatrix} \blacksquare$$

## 3.6 Markov Matrices

The definition of the transition matrix **P** results in a matrix with peculiar properties:

1. The number of rows equals the number of columns. Thus **P** is a *square matrix*.
2. All the elements of **P** are real numbers. Thus **P** is a *real matrix*.
3. $0 \leq p_{ij} \leq 1$ for all values of $i$ and $j$. Thus **P** is a *nonnegative matrix*.
4. The sum of each column is exactly 1 (i.e. $\sum_{j=1}^{m} p_{ij} = 1$).
5. The magnitude of all eigenvalues obeys the condition $|\lambda_i| \leq 1$. Thus the spectral radius of **P** equals 1.
6. At least one of the eigenvalues of **P** equals 1.

From all the above properties we conclude that the transition matrix is square, real, and nonnegative. Such a matrix is termed *column stochastic* matrix or *Markov* matrix. Notice that all column stochastic matrices are a subset of nonnegative matrices. Thus nonnegative matrices need not be column stochastic.

Nonnegative matrices have many interesting properties related to their eigenvalues and eigenvectors but this is beyond the scope of this book [2].

The above properties have implications on the eigenvalues of the transition matrix. The following theorem indicates one such implication.

**Theorem 3.1.** *Let* $\mathbf{P}$ *be any* $m \times m$ *column stochastic matrix. Then* $\mathbf{P}$ *has 1 as an eigenvalue [2].*

*Proof.* We know that if $\lambda$ is an eigenvalue of $\mathbf{P}$, then the determinant of the characteristic equation must vanish, i.e.:

$$\det(\mathbf{P} - \lambda \mathbf{I}) = 0 \tag{3.13}$$

where $\mathbf{I}$ is an $m \times m$ unit matrix. Assume that $\mathbf{P}$ has 1 as an eigenvalue, then the determinant is given by:

$$\det(\mathbf{P} - 1 \times \mathbf{I}) = \begin{vmatrix} p_{11} - 1 & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} - 1 & \cdots & p_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} - 1 \end{vmatrix} \tag{3.14}$$

where $|\mathbf{A}|$ indicates the determinant of matrix $\mathbf{A}$. The determinant will not change if we add all the remaining rows to the first row:

$$\det(\mathbf{P} - 1 \times \mathbf{I}) = \begin{vmatrix} \sum p_{i1} - 1 & \sum p_{i2} - 1 & \cdots & \sum p_{im} - 1 \\ p_{21} & p_{22} - 1 & \cdots & p_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} - 1 \end{vmatrix} \tag{3.15}$$

But the sum of the elements in each column is 1 and the first row of the above determinant will be zero. As a result, the determinant is zero and this proves that 1 is an eigenvalue of $\mathbf{P}$.

Conversely, assume that (3.13) is satisfied for some value of $\lambda$. In that case we can write an equation similar to (3.15), namely:

$$\det(\mathbf{P} - 1 \times \mathbf{I}) = \begin{vmatrix} \sum p_{i1} - \lambda & \sum p_{i2} - \lambda & \cdots & \sum p_{im} - \lambda \\ p_{21} & p_{22} - 1 & \cdots & p_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} - 1 \end{vmatrix} \tag{3.16}$$

But we know that this determinant is zero. This is true for all values of $p_{ij}$ which implies that the elements in the first row of the matrix must be zero. Thus we must have:

$$\sum p_{i1} - \lambda = 0 \tag{3.17}$$

But the sums in the above equation is equal to 1 independent of the value of $i$. Thus we must have:

$$1 - \lambda = 0 \tag{3.18}$$

The above equation implies that $\lambda = 1$ is a root of the equation which proves the second part of the theorem. □

The following theorem will prove useful when we start multiplying Markov matrices to perform transient analysis on our Markov chain. The theorem essentially explains the effect of premultiplying any matrix by a column stochastic matrix.

**Theorem 3.2.** *The sum of columns of any matrix* **A** *will not change when it is premultiplied by a column stochastic matrix* **P**.

*Proof.* When **A** is premultiplied by **P** we get matrix **B**:

$$\mathbf{B} = \mathbf{PA} \tag{3.19}$$

Element $b_{ij}$ is given by the usual matrix product formula:

$$b_{ij} = \sum_{k=1}^{m} p_{ik}\ a_{kj} \tag{3.20}$$

The sum of the $j$th column of matrix **B** is denoted by $\sigma_j(\mathbf{B})$ and is given by:

$$\sigma_j(\mathbf{B}) = \sum_{i=1}^{m} b_{ij} = \sum_{i=1}^{m} \sum_{k=1}^{m} p_{ik}\ a_{kj} \tag{3.21}$$

Now reverse the order of summation on the right-hand side of the above equation:

$$\sigma_j(\mathbf{B}) = \sum_{k=1}^{m} a_{kj} \sum_{i=1}^{m} p_{ik} \tag{3.22}$$

Because **P** is column stochastic we have:

$$\sigma_k(\mathbf{P}) = \sum_{i=1}^{m} p_{ik} = 1 \tag{3.23}$$

Therefore, (3.22) becomes:

$$\sigma_j(\mathbf{B}) = \sum_{i=1}^{m} b_{ij} = \sum_{k=1}^{m} a_{kj} \tag{3.24}$$

$$= \sigma_j(\mathbf{A}) \tag{3.25}$$

Thus we proved that sum of columns of a matrix does not change when the matrix is premultiplied by a column stochastic matrix. □

**Table 3.2** Significance of diagonals of **P**

| Diagonal | Significance |
|---|---|
| Main | Probabilities queue will retain its size |
| $1^{st}$ upper | Probabilities queue size will decrease by one |
| $2^{nd}$ upper | Probabilities queue size will decrease by two |
| $3^{rd}$ upper | Probabilities queue size will decrease by three |
| $\vdots$ | $\vdots$ |
| $1^{st}$ lower | Probabilities queue size will increase by one |
| $2^{nd}$ lower | Probabilities queue size will increase by two |
| $3^{rd}$ lower | Probabilities queue size will increase by three |
| $\vdots$ | $\vdots$ |

### *3.6.1 The Diagonals of P*

We mentioned above the significance of each element $p_{ij}$ of the transition matrix **P** as the probability of making a transition from state $j$ to state $i$. Further insight can be gained for Markov chains representing queues. *Queuing systems* are a special type of Markov chains in which customers arrive and lineup to be serviced by servers. Thus a queue is characterized by the number of arriving customers at a given time step, the number of servers, the size of the waiting area for customers, and the number customers that can leave in one time step.

The state of a queuing system corresponds to the number of customers in the queue. If we take the lineup for a bank as an example, then the queue size increases when new customers arrive. The number of arriving customers could be one in some cases or many in others. This depends on the specifics of the situation. For example, if there is only one door to the bank, then we could expect at most one customer to arrive at any time. At the head of the queue, the number of servers varies also depending on how many bank tellers are ready, or disposed, to serve the customers. If there is only one teller, then we expect the size of the queue to decrease by at most one each time a customer is served. The duration of the service time also varies depending on the type of transaction being done.

The diagonals of **P** reflect the queuing system characteristics. Table 3.2 illustrates the significance of each diagonal of the matrix **P**

## 3.7 Eigenvalues and Eigenvectors of P

The eigenvalues and eigenvectors of the transition matrix **P** will prove to be of utmost importance in the analyses of this book.

The following theorem makes certain predictions about the eigenvector corresponding to the eigenvalue $\lambda = 1$.

**Theorem 3.3.** *Given a column stochastic matrix* $\mathbf{P}$ *and the eigenvector* $\mathbf{x}$ *corresponding to the eigenvalue* $\lambda = 1$*, the sum of the elements of* $\mathbf{x}$ *is nonzero and could be taken as unity, i.e.* $\sigma(x) = 1$.

*Proof.* The eigenvector $\mathbf{x}$ corresponding to $\lambda = 1$ satisfies the equation:

$$\mathbf{P}\,\mathbf{x} = \mathbf{x} \tag{3.26}$$

We can write the above equation as:

$$(\mathbf{P} - \mathbf{I})\ \mathbf{x} = 0 \tag{3.27}$$

This is a system of linear equation with an infinite number of solutions since the matrix $(\mathbf{P} - \mathbf{I})$ is rank deficient (i.e., rank$(\mathbf{P}) < m$). To get a unique solution for $\mathbf{x}$ we need one extra equation which we choose as:

$$\sigma(x) = 1 \tag{3.28}$$

We cannot choose the sum to be zero since this is a trivial solution. Any nonzero value is acceptable. We choose to have $\sigma(x) = 1$ for reasons that will become apparent later on. This proves the theorem. □

The following theorem makes certain predictions about the sum of elements of the other eigenvectors of $\mathbf{P}$ corresponding to the eigenvalues $\lambda < 1$.

**Theorem 3.4.** *Given a column stochastic matrix* $\mathbf{P}$ *and an eigenvector* $\mathbf{x}$ *corresponding to the eigenvalue* $\lambda \neq 1$*, the sum of the elements of* $\mathbf{x}$ *must be zero, i.e.* $\sigma(x) = 0$.

*Proof.* The eigenvector $\mathbf{x}$ satisfies the equation:

$$\mathbf{P}\,\mathbf{x} = \lambda\,\mathbf{x} \tag{3.29}$$

The sum of columns of both sides of the above equation is equal:

$$\sigma(\mathbf{P}\,\mathbf{x}) = \lambda\,\sigma(\mathbf{x}) \tag{3.30}$$

From Theorem 3.2, on page 79, we are assured that the sum of the elements of $\mathbf{x}$ will not change after being multiplied by matrix $\mathbf{P}$. Thus we can write:

$$\sigma(\mathbf{P}\,\mathbf{x}) = \sigma(\mathbf{x}) \tag{3.31}$$

From the above two equations we have:

$$\sigma(x) = \lambda\sigma(x) \tag{3.32}$$

or

$$\sigma(x)(1-\lambda) = 0 \tag{3.33}$$

Since $\lambda \neq 1$, the only possible solution to the above equation is $\sigma(x) = 0$. This proves the theorem. □

*Example 3.9.* Verify Theorems 3.3 and 3.4 for the Markov matrix:

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.1 & 0 & 0.2 \\ 0.1 & 0.6 & 0.3 & 0.1 \\ 0.4 & 0.2 & 0.4 & 0.5 \\ 0.2 & 0.1 & 0.3 & 0.2 \end{bmatrix}$$

MATLAB gives the following eigenvectors and corresponding eigenvalues

```
[X,D] = eig(P)

X =
 -0.1965   0.5887  -0.1002  -0.4286
 -0.6309   0.3983  -0.7720   0.1644
 -0.6516  -0.5555   0.5190  -0.4821
 -0.3724  -0.4315   0.3532   0.7463


D =

    1.0000         0         0         0
         0    0.2211         0         0
         0         0    0.3655         0
         0         0         0   -0.0866
```

We have to normalize our eigenvectors so that the sum of the components of the first column, which corresponds to $\lambda = 1$ is one.

```
X = X/sum(X(:,1))

X =
    0.1061   -0.3179    0.0541    0.2315
    0.3408   -0.2151    0.4169   -0.0888
    0.3520    0.3001   -0.2803    0.2604
    0.2011    0.2330   -0.1907   -0.4031
```

We can write, ignoring rounding errors:

$$\sigma_1(\mathbf{X}) = 1$$

$$\sigma_2(\mathbf{X}) = 0$$
$$\sigma_3(\mathbf{X}) = 0$$
$$\sigma_4(\mathbf{X}) = 0$$

■

## 3.8 Constructing the State Transition Matrix P

The state transition matrix **P** is the key to analyzing Markov chains and queues. To construct the matrix the following steps are usually followed [3].

1. Verify that the system under study displays the Markov property. In other words, ensure that transition to a new state depends only on the current state.
2. All possible states of the system are identified and labeled. The labeling of the states is arbitrary although some labeling schemes would render the transition matrix easier to visualize.
3. All possible transitions between the states are either drawn on the state transition diagram, or the corresponding elements of the state transition matrix are identified.
4. The probability of every transition in the state diagram is obtained.
5. The transition matrix is constructed.
6. Relabeling of the states is always possible. That will change the locations of the matrix elements and make the structure of the matrix more visible. This rearrangement will still produce a column stochastic matrix and will not disturb its eigenvalues or the *directions* of its eigenvectors.

*Example 3.10.* The closing price of a certain stock on a given weekday is either falling or rising compared to the previous day's price. If the price stays the same, then it is classified as rising if the previous day's trend was rising, and vice versa. The probabilities of price transitions between these two states are shown in Fig. 3.6. Construct state transition matrix.

The price of the stock has only two states falling ($s_1$) or rising ($s_2$). The transition matrix will be:

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.6 \\ 0.7 & 0.4 \end{bmatrix}$$

■

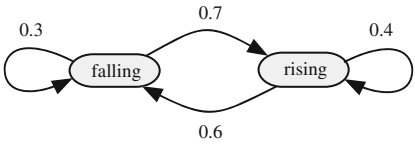



**Fig. 3.6** Day-to-day closing price fluctuations of a certain stock

## 3.9 Transient Behavior

From (3.8) on page 72 we can write the distribution vector at time step $n = 1$ as:

$$\mathbf{s}(1) = \mathbf{P}\,\mathbf{s}(0) \tag{3.34}$$

and

$$\mathbf{s}(2) = \mathbf{P}\,\mathbf{s}(1) \tag{3.35}$$

$$= \mathbf{P}\,[\mathbf{P}\,\mathbf{s}(0)] \tag{3.36}$$

$$= \mathbf{P}^2\,\mathbf{s}(0) \tag{3.37}$$

and we can generalize to express the distribution vector at step $n$ as:

$$\mathbf{s}(n) = \mathbf{P}^n\,\mathbf{s}(0) \qquad n = 0, 1, 2, \ldots \tag{3.38}$$

This equation allows us to calculate the distribution vector at the $n$th time step given the transition matrix and the initial distribution vector $\mathbf{s}(0)$.

*Example 3.11.* In Example 3.6, on page 73, what is the probability that our truck is in Colwood after five deliveries assuming that it was Langford initially?

We are looking for element $p_{12}$ in matrix $\mathbf{P}^5$. Using any mathematical tool such as MAPLE or MATLAB, we get:

$$\mathbf{P}^5 = \begin{bmatrix} 0.199 & 0.2 & 0.200 \\ 0.288 & 0.277 & 0.278 \\ 0.513 & 0.523 & 0.522 \end{bmatrix}$$

hence the required probability is 0.2. ■

*Example 3.12.* Assume a ball falls on the pegs shown in Fig. 3.7. The pegs are setup such that the ball must hit a peg at each level and bounce off one of the two pegs immediately below. The location of the ball at the bottom bucket indicates the prize to be won.

(a) Define a Markov chain describing the state of the ball.
(b) Write down the transition matrix.
(c) Write down the initial distribution vector if the ball is dropped in the middle hole.
(d) Determine the probability of hitting the middle bucket.
(e) Assume the money to be won is given by the vector:

$$\mathbf{w} = \begin{bmatrix} \$5 & \$1 & \$10 & \$1 & \$5 \end{bmatrix}^t$$

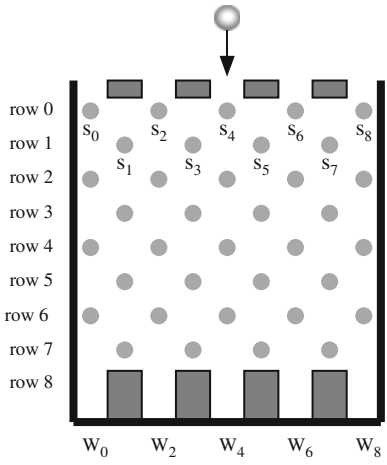



**Fig. 3.7** A ball falling through a maze of pegs

Determine the average amount of money that could be won.

Since the next location of the ball depends only on its present location, we can describe this system using Markov chains.

(a) We can model this system as a Markov chain with nine states $s_1$ to $s_9$, as indicated in the figure. State $s_i(n)$ indicates that the ball is at column $i$ and row $n$ in the game. The rows are numbered starting with zero at the top row. Thus our distribution vector could be written as:

$$= \begin{bmatrix} s_0 & s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & s_7 & s_8 \end{bmatrix}^t$$

where at even time steps the even states could be occupied and at odd time steps only the odd states could be occupied.

(b) The transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix} 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \end{bmatrix}$$

(c) The initial distribution vector is given by:

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^t$$

(d) After eight iterations, the ball finally falls into one of the buckets. Thus the distribution vector after eight iterations is given by:

$$\begin{aligned}\mathbf{s}(8) &= \mathbf{P}^8\,\mathbf{s}(0)\\ &= \begin{bmatrix}0.11 & 0 & 0.25 & 0 & 0.28 & 0 & 0.25 & 0 & 0.11\end{bmatrix}^t\end{aligned}$$

Note that the only valid states are the ones that correspond to a bucket locations at the bottom of the figure. This explains the zeros in the odd locations 1, 3, 5, and 7. The probability of settling into the middle bucket is $s_4 = 0.28$.

(e) The average winnings are given by:

$$\begin{aligned}W_a &= W_0\,s_0(8) + W_2\,s_2(8) + W_4\,s_4(8) + W_6\,s_6(8) + W_8\,s_8(8)\\ &= \$4.4\end{aligned}$$

It is interesting that if hundreds of balls are dropped at the center toward the bottom, then their distribution at the bottom barrels is bell-shaped and their number is the binomial coefficients [1] ■

*Example 3.13.* A computer memory system is composed of very fast on-chip cache, fast on-board RAM, and slow hard disk. When the computer is accessing a block from each memory system, the next block required could come form any of the three available memory systems. This is modeled as a Markov chain with the state of the system representing the memory from which the current block came from: state $s_1$ corresponds to the cache, state $s_2$ corresponds to the RAM, and state $s_3$ corresponds to the hard disk. The transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix}0.7 & 0.1 & 0\\ 0.2 & 0.7 & 0.1\\ 0.1 & 0.2 & 0.9\end{bmatrix}$$

Find the probability that after three consecutive block accesses the system will read a block from the cache.

The starting distribution vector is $\mathbf{s}(0) = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^t$ and we have:

$$\mathbf{P}^2 = \begin{bmatrix}0.51 & 0.14 & 0.01\\ 0.29 & 0.53 & 0.16\\ 0.20 & 0.33 & 0.83\end{bmatrix}$$

$$\mathbf{P}^3 = \begin{bmatrix}0.386 & 0.151 & 0.023\\ 0.325 & 0.432 & 0.197\\ 0.289 & 0.417 & 0.780\end{bmatrix}$$

The distribution vector after three iterations is:

$$\mathbf{s}(3) = \mathbf{P}^3\ \mathbf{s}(0) = \begin{bmatrix} 0.386 & 0.325 & 0.289 \end{bmatrix}^t$$

The probability that the system will read a block from the cache is 0.386. ■

### 3.9.1 *Properties of* $\mathbf{P}^n$

Matrix $\mathbf{P}^n$ has many interesting properties that we list below:

- $\mathbf{P}^n$ remains a column stochastic matrix according to Lemma 3.1 below.
- A nonzero element in $\mathbf{P}$ can increase or decrease in $\mathbf{P}^n$ but can never become zero.
- A zero element in $\mathbf{P}$ could remain zero or increase in $\mathbf{P}^n$ but can never become negative.

As a result of Theorem 3.2 on page 79, we deduce the following two lemmas.

**Lemma 3.1.** *Given a column stochastic matrix* $\mathbf{P}$*, then* $\mathbf{P}^n$*, for* $n \geq 0$*, is also column stochastic.*

The proof is easily derived from Theorem 3.2. □

**Lemma 3.2.** *The state vector* $\mathbf{s}(n)$ *at instance* $n$ *is given by:*

$$\mathbf{s}(n) = \mathbf{P}^n\ \mathbf{s}(0) \tag{3.39}$$

*This vector must be a distribution vector for all values of* $n \geq 0$*. The proof is easily derived from Theorem 3.2 after applying the theorem to the state vector* $\mathbf{s}(0)$. □

## 3.10 Finding s($n$)

We can determine the state of our Markov chain at time step $n$ if we are able to calculate $\mathbf{P}^n$. In general performing $n$ matrix multiplications is tedious and leads to computational noise. Besides, no insight can be gained from repeatedly multiplying a matrix. Alternative techniques for obtaining an expression for $\mathbf{s}(n)$ or $\mathbf{P}^n$ include:

1. Repeated multiplications to get $\mathbf{P}^n$.
2. Expanding the initial distribution vector $\mathbf{s}(0)$.
3. Diagonalizing the matrix $\mathbf{P}$.
4. Using the Jordan canonic form of $\mathbf{P}$.
5. Using the z-transform.

The first method is simple and is best done using a mathematical package such as MATLAB. We have seen examples of this technique in the previous section. We show in the following sections how the other techniques are used.

## 3.11 Finding s($n$) by Expanding s(0)

In most cases the transition matrix $\mathbf{P}$ is simple, i.e. it has $m$ distinct eigenvalues. In that case we can express our initial distribution vector $\mathbf{s}(0)$ as a linear combination of the $m$ eigenvectors:

$$\mathbf{s}(0) = c_1\,\mathbf{x}_1 + c_2\,\mathbf{x}_2 + \cdots + c_m\,\mathbf{x}_m \tag{3.40}$$

where $\mathbf{x}_i$ is the $i$th eigenvector of $\mathbf{P}$ and $c_i$ is the corresponding scalar expansion coefficients. We can write the above equation as a simple matrix expression:

$$\mathbf{s}(0) = \mathbf{X}\,\mathbf{c} \tag{3.41}$$

where $\mathbf{X}$ is an $m \times m$ matrix whose columns are the eigenvectors of $\mathbf{P}$ and $\mathbf{c}$ is an $m$-component vector of the expansion coefficients:

$$\mathbf{X} = \left[\,\mathbf{x}_1\ \mathbf{x}_2 \cdots \mathbf{x}_m\,\right] \tag{3.42}$$

$$\mathbf{c} = \left[\,c_1\ c_2 \cdots c_m\,\right]^t \tag{3.43}$$

We need not normalize the eigenvectors before we determine the coefficients vector $\mathbf{c}$ because any normalization constant for $\mathbf{X}$ will be accounted for while determining $\mathbf{c}$.

To find $\mathbf{s}(n)$ we will use the technique explained below. Equation (3.41) is a system of $m$-linear equations in $m$ unknowns, namely the components of the column vector $\mathbf{c}$. There are many numerical techniques for finding these components like Gauss elimination, Kramer's rule, etc. MATLAB has a very simple function for finding the eigenvectors of $\mathbf{P}$:

```
X = eig(P)
```
(3.44)

where matrix $\mathbf{X}$ will contain the eigenvectors in its columns. To find the coefficient vector $\mathbf{c}$ we use MATLAB to solve the system of linear equations in Eq. (3.41) by typing the MATLAB command:

```
c = X \ s
```
(3.45)

where the backslash operator "\" effectively solves for the unknown vector $\mathbf{c}$ using Gaussian elimination.

**WWW** The function `EIGPOWERS(P,s)` can be used to expand $\mathbf{s}$ in terms of the eigenvectors of $\mathbf{P}$.

*Example 3.14.* A Markov chain has the state matrix:

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.2 & 0.2 & 0.2 \\ 0.3 & 0.4 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.4 & 0.4 \end{bmatrix}$$

Check to see if this matrix has distinct eigenvalues. If so, expand the initial distribution vector:

$$\mathbf{s}(0) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^t$$

in terms of the eigenvectors of $\mathbf{P}$.

The eigenvalues of $\mathbf{P}$ are $\lambda_1 = 1$, $\lambda_2 = 0.1$, $\lambda_3 = 0.2$ and $\lambda_4 = 0$. The eigenvectors corresponding to these eigenvalues are:

$$\mathbf{X} = \begin{bmatrix} 0.439 & -0.707 & 0.0 & 0.0 \\ 0.548 & 0.707 & 0.707 & 0.0 \\ 0.395 & 0.0 & 0.0 & 0.707 \\ 0.592 & 0.0 & -0.707 & -0.707 \end{bmatrix}$$

Therefore, we can expand $\mathbf{s}(0)$ in terms of the corresponding eigenvectors:

$$\mathbf{s}(0) = \mathbf{X}\,\mathbf{c}$$

where $\mathbf{c}$ given by:

$$\mathbf{c} = \begin{bmatrix} 0.507 & -1.1 & 0.707 & 0.283 \end{bmatrix}^t \qquad \blacksquare$$

Now let us see how to get a simple expression for evaluating $\mathbf{s}(n)$ given the expansion of $\mathbf{s}(0)$. We start by using (3.41) to find $\mathbf{s}(1)$ as:

$$\begin{aligned} \mathbf{s}(1) &= \mathbf{P}\,\mathbf{s}(0) && (3.46) \\ &= \mathbf{P}\ (c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \cdots + c_m\mathbf{x}_m) && (3.47) \\ &= c_1\lambda_1\mathbf{x}_1 + c_2\lambda_2\mathbf{x}_2 + \cdots + c_m\lambda_m\mathbf{x}_m && (3.48) \end{aligned}$$

where $\lambda_i$ is the $i$th eigenvalue of $\mathbf{P}$ corresponding to the $i$th eigenvector $\mathbf{x}_i$.

We can express $\mathbf{s}(1)$ in the above equation in matrix form as follows:

$$\mathbf{s}(1) = \mathbf{X}\,\mathbf{D}\,\mathbf{c} \qquad (3.49)$$

where $\mathbf{X}$ was defined before and $\mathbf{D}$ is the diagonal matrix whose diagonal elements are the eigenvalues of $\mathbf{P}$:

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m \end{bmatrix} \tag{3.50}$$

MATLAB provides a very handy function for finding the two matrices $\mathbf{X}$ and $\mathbf{D}$ using the single command:

```
[X,D] = eig(P)
```
(3.51)

Having found $\mathbf{s}(1)$, we now try to find $\mathbf{s}(2)$:

$$\mathbf{s}(2) = \mathbf{P}\,\mathbf{s}(1) \tag{3.52}$$

$$= \mathbf{P}\,\mathbf{X}\,\mathbf{D}\,\mathbf{c} \tag{3.53}$$

but the eigenvectors satisfy the relation:

$$\mathbf{P}\mathbf{x}_i = \lambda_i \mathbf{x}_i \tag{3.54}$$

In matrix form we can write the above equation as:

$$\mathbf{P}\mathbf{X} = \mathbf{X}\,\mathbf{D} \tag{3.55}$$

Substituting (3.55) into (3.53) we get:

$$\mathbf{s}(2) = \mathbf{X}\,\mathbf{D}^2\,\mathbf{c} \tag{3.56}$$

where

$$\mathbf{D}^2 = \begin{bmatrix} \lambda_1^2 & 0 & \cdots & 0 \\ 0 & \lambda_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m^2 \end{bmatrix} \tag{3.57}$$

In general the distribution vector at time step $n$ is given by:

$$\mathbf{s}(n) = \mathbf{X}\mathbf{D}^n\mathbf{c} \tag{3.58}$$

where

$$\mathbf{D}^n = \begin{bmatrix} \lambda_1^n & 0 & \cdots & 0 \\ 0 & \lambda_2^n & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m^n \end{bmatrix} \tag{3.59}$$

It is relatively easy to find the $n$th power of a diagonal matrix by simply raising each diagonal element to the $n$th power. Then the distribution vector at time step $n$ is simply obtained from (3.58).

*Example 3.15.* Consider the Markov chain in Example 3.14. Find the values of the distribution vector at time steps 2, 5, and 20.

For the given transition matrix, we can write:

$$\mathbf{s}(n) = \mathbf{X}\mathbf{D}^n\mathbf{c}$$

where the matrices $\mathbf{X}$, $\mathbf{D}$, and the vector $\mathbf{c}$ are given by:

$$\mathbf{X} = \begin{bmatrix} 0.439 & -0.707 & 0.0 & 0.0 \\ 0.548 & 0.707 & 0.707 & 0.0 \\ 0.395 & 0.0 & 0.0 & 0.707 \\ 0.592 & 0.0 & -0.707 & -0.707 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{c} = \begin{bmatrix} 0.507 & -1.1 & 0.707 & 0.283 \end{bmatrix}^t$$

Thus we can simply write:

$$\begin{aligned} \mathbf{s}(2) &= \mathbf{X}\mathbf{D}^2\mathbf{c} \\ &= \begin{bmatrix} 0.23 & 0.29 & 0.2 & 0.28 \end{bmatrix}^t \\ \mathbf{s}(5) &= \mathbf{X}\mathbf{D}^5\mathbf{c} \\ &= \begin{bmatrix} 0.22 & 0.28 & 0.2 & 0.3 \end{bmatrix}^t \\ \mathbf{s}(20) &= \mathbf{X}\mathbf{D}^{20}\mathbf{c} \\ &= \begin{bmatrix} 0.22 & 0.28 & 0.2 & 0.3 \end{bmatrix}^t \end{aligned}$$

We see that the distribution vector settles down to a fixed value after a few iterations. The above results could be confirmed by directly finding the distribution vectors using the usual formula:

$$\mathbf{s}(n) = \mathbf{P}^n \, \mathbf{s}(0) \qquad \blacksquare$$

Sometimes two eigenvalues are the complex conjugate of each other. In that case the corresponding eigenvectors will be complex conjugate and so are the corresponding coefficient $c$ such that the end result $\mathbf{s}(n)$ is purely real.

*Example 3.16.* A Markov chain has the transition matrix:

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.4 & 0.2 \\ 0.1 & 0.4 & 0.6 \\ 0.8 & 0.2 & 0.2 \end{bmatrix}$$

Check to see if this matrix has distinct eigenvalues. If so,

(a) Expand the initial distribution vector:

$$\mathbf{s}(0) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^t$$

in terms of the eigenvectors of $\mathbf{P}$.

(b) Find the value of $\mathbf{s}(3)$.

The eigenvalues of $\mathbf{P}$ are:

$$\lambda_1 = 1$$
$$\lambda_2 = -0.15 + j0.3122$$
$$\lambda_3 = -0.15 - j0.3122$$

We see that complex eigenvalues appear as complex conjugate pairs. The eigenvectors corresponding to these eigenvalues are:

$$\mathbf{X} = \begin{bmatrix} -0.4234 & -0.1698 + j0.3536 & -0.1698 - j0.3536 \\ -0.6726 & -0.5095 - j0.3536 & -0.5095 + j0.3536 \\ -0.6005 & 0.6794 & 0.6794 \end{bmatrix}$$

We see that the eigenvectors corresponding to the complex eigenvalues appear also as complex conjugate pairs. Therefore, we can expand $\mathbf{s}(0)$ in terms of the corresponding eigenvectors:

$$\mathbf{s}(0) = \mathbf{X}\,\mathbf{c}$$

where $\mathbf{c}$ is given by:

$$\mathbf{c} = \begin{bmatrix} -5863 & -0.2591 - j0.9312 & -0.2591 + j0.9312 \end{bmatrix}^t$$

We see that the expansion coefficients corresponding to the complex eigenvectors appear as complex conjugate pairs also. This ensures that all the components of the distribution vector will always be real numbers.

The distribution vector at time step $n = 3$ is:

$$\mathbf{s}(3) = \mathbf{X}\mathbf{D}^3\mathbf{c} = \begin{bmatrix} 0.2850 & 0.3890 & 0.3260 \end{bmatrix}^t \quad \blacksquare$$

## 3.12 Finding s($n$) by Diagonalizing P

According to reference [2], matrix $\mathbf{P}$ is diagonalizable when it has $m$ distinct eigenvalues and we can write:

$$\mathbf{P} = \mathbf{X}\mathbf{D}\mathbf{X}^{-1} \tag{3.60}$$

where $\mathbf{X}$ is the matrix whose columns are the eigenvectors of $\mathbf{P}$ and $\mathbf{D}$ is a diagonal matrix whose diagonal elements are the eigenvalues arranged according to the ordering of the columns of $\mathbf{X}$. MATLAB provides a very handy function for finding the matrices $\mathbf{X}$ and $\mathbf{D}$ using the single command:

```
[X,D] = eig(P)
```

It does not matter here whether the columns of $\mathbf{X}$ are normalized or not since any scaling factor in $\mathbf{X}$ will be cancelled by $\mathbf{X}^{-1}$. MATLAB also offers a very simple function for inverting a matrix by using the command:

```
X_inv = inv(X)
```

We can calculate $\mathbf{P}^2$ as:

$$\mathbf{P}^2 = \left(\mathbf{X}\mathbf{D}\mathbf{X}^{-1}\right) \times \left(\mathbf{X}\mathbf{D}\mathbf{X}^{-1}\right) \tag{3.61}$$

$$= \mathbf{X}\mathbf{D}^2\mathbf{X}^{-1} \tag{3.62}$$

In general we have:

$$\mathbf{P}^n = \mathbf{X}\mathbf{D}^n\mathbf{X}^{-1} \tag{3.63}$$

where

$$\mathbf{D}^n = \begin{bmatrix} \lambda_1^n & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2^n & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3^n & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_m^n \end{bmatrix} \tag{3.64}$$

It is easy therefore to find $\mathbf{s}(n)$ at any value for $n$ by simply finding $\mathbf{D}^n$ and then evaluate the simple matrix multiplication expression:

$$\begin{aligned} \mathbf{s}(n) &= \mathbf{P}^n\, \mathbf{s}(0) \\ &= \mathbf{X}\mathbf{D}^n\mathbf{X}^{-1}\, \mathbf{s}(0) \end{aligned} \tag{3.65}$$

### 3.12.1 Comparing Diagonalization with Expansion of s(0)

Diagonalizing the matrix $\mathbf{P}$ is equivalent to the previous technique of expanding $\mathbf{s}(0)$ in terms of the eigenvectors of $\mathbf{P}$. As a proof, consider calculating $\mathbf{s}(n)$ using both techniques.

Using diagonalization technique, we have:

$$\mathbf{s}(n) = \mathbf{P}^n\mathbf{s}(0) \tag{3.66}$$

$$= \mathbf{X}\mathbf{D}^n\mathbf{X}^{-1}\, \mathbf{s}(0) \tag{3.67}$$

Using the expansion of $\mathbf{s}(0)$ technique in (3.41) we know that:

$$\mathbf{s}(0) = \mathbf{X}\mathbf{c} \tag{3.68}$$

Substituting this into (3.67) we get:

$$\begin{aligned} \mathbf{s}(n) &= \mathbf{X}\mathbf{D}^n\mathbf{X}^{-1}\mathbf{X}\, \mathbf{c} \\ &= \mathbf{X}\mathbf{D}^n\, \mathbf{c} \end{aligned} \tag{3.69}$$

Now the value for $\mathbf{c} = \mathbf{X}^{-1}\mathbf{s}(0)$ can be found to yield:

$$\mathbf{s}(n) = \mathbf{X}\mathbf{D}^n\mathbf{X}^{-1}\, \mathbf{s}(0) \tag{3.70}$$

This last expression for $\mathbf{s}(n)$ is identical to Eq. (3.67). Thus we see that the two techniques are equivalent.

Before we finish this section we state the following lemma which will prove useful later on.

**Lemma 3.3.** *Assume the first column of* $\mathbf{X}$ *is the normalized eigenvector corresponding to eigenvalue* $\lambda = 1$. *Then the matrix* $\mathbf{X}^{-1}$ *will have ones in its first row.*

*In general, if the first column of* $\mathbf{X}$ *is not normalized, then the theorem would state: The matrix* $\mathbf{X}^{-1}$ *will have the value* $1/\sigma(\mathbf{x}_1)$ *in the elements in its first row.*

*Proof.* Assume $\mathbf{Y} = \mathbf{X}^{-1}$ then we can write:

$$\mathbf{X} \times \mathbf{Y} = \mathbf{I} \tag{3.71}$$

where $\mathbf{I}$ is the $m \times m$ unit matrix. Let us express the above equation in terms of the elements of the two matrices $\mathbf{X}$ and $\mathbf{Y}$:

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mm} \end{bmatrix} \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mm} \end{bmatrix} = \mathbf{I} \tag{3.72}$$

The element at location $(i, j)$ is obtained from the usual formula:

$$\sum_{k=1}^{m} x_{ik}\ y_{kj} = \delta(i - j) \tag{3.73}$$

where $\delta(i - j)$ is the Dirac delta function which is one only when the argument is zero, i.e. when $i = j$. The function is zero for all other values of $i$ and $j$.

The sum of the $j$th column on both sides of (3.71) or (3.73) is given by:

$$\sum_{i=1}^{m} \sum_{k=1}^{m} x_{ik}\ y_{kj} = 1 \tag{3.74}$$

Now reverse the order of summation on the left-hand side of the above equation:

$$\sum_{k=1}^{m} y_{kj} \sum_{i=1}^{m} x_{ik} = 1 \tag{3.75}$$

Because of the properties of the eigenvectors in Theorem 3.3 on page 80 and Theorem 3.4 on page 81, the second summation becomes:

$$\sum_{k=1}^{m} y_{kj}\ \delta(k - 1) = 1 \tag{3.76}$$

$\delta(k-1)$ expresses the fact that only the sum of the first column of matrix $\mathbf{X}$ evaluates to 1. All other columns add up to zero. Thus the above equation becomes:

$$y_{1j} = 1 \tag{3.77}$$

This proves that the elements of the first row of matrix $\mathbf{X}^{-1}$ must be all ones. □

## 3.13 Expanding $\mathbf{P}^n$ in Terms of its Eigenvalues

We shall find that expanding $\mathbf{P}^n$ in terms of the eigenvalues of $\mathbf{P}$ will give us additional insights into the transient behavior of Markov chains.

Equation (3.63) related $\mathbf{P}^n$ to the $n$th powers of its eigenvalues as:

$$\mathbf{P}^n = \mathbf{X}\mathbf{D}^n\mathbf{X}^{-1} \tag{3.78}$$

Thus we can express $\mathbf{P}^n$ in the above equation in the form:

$$\mathbf{P}^n = \lambda_1^n\mathbf{A}_1 + \lambda_2^n\mathbf{A}_2 + \cdots + \lambda_m^n\mathbf{A}_m \tag{3.79}$$

Assume that $\lambda_1 = 1$ and all other eigenvalues have magnitudes lesser than unity (fractions) because $\mathbf{P}$ is column stochastic. In that case, the above equation becomes:

$$\mathbf{P}^n = \mathbf{A}_1 + \lambda_2^n\mathbf{A}_2 + \cdots + \lambda_m^n\mathbf{A}_m \tag{3.80}$$

This shows that as time progresses, $n$ becomes large and the powers of $\lambda_i^n$ will quickly decrease. The main contribution to $\mathbf{P}^n$ will be due to $\mathbf{A}_1$ only.

The matrices $\mathbf{A}_i$ can be determined from the product:

$$\mathbf{A}_i = \mathbf{X}\,\mathbf{Y}_i\,\mathbf{X}^{-1} \tag{3.81}$$

where $\mathbf{Y}_i$ is the *selection matrix* which has zeros everywhere except for element $y_{ii} = 1$.

For a $3 \times 3$ matrix, we can write:

$$\mathbf{P}^n = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3\end{bmatrix}\begin{bmatrix}\lambda_1^n & 0 & 0\\ 0 & \lambda_2^n & 0\\ 0 & 0 & \lambda_3^n\end{bmatrix}\begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3\end{bmatrix}^{-1} \tag{3.82}$$

$$\mathbf{A}_1 = \mathbf{X}\begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\mathbf{X}^{-1} \tag{3.83}$$

$$\mathbf{A}_2 = \mathbf{X} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{X}^{-1} \tag{3.84}$$

$$\mathbf{A}_3 = \mathbf{X} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{X}^{-1} \tag{3.85}$$

The selection matrix effectively ensures that $\mathbf{A}_i$ is given by the product of the $i$th column of $\mathbf{X}$ and the $i$th row of $\mathbf{X}^{-1}$

**WWW** We developed the function `SELECT(P)` that can be used to obtain the different matrices $\mathbf{A}_i$ for a diagonalizable matrix.

It will prove useful to write $\mathbf{A}_i$ explicitly in terms of the corresponding eigenvector. $\mathbf{A}_i$ can be written in the alternative form:

$$\mathbf{A}_i = \mathbf{x}_i \, \mathbf{z}_i \tag{3.86}$$

where $\mathbf{x}_i$ is the $i$th eigenvector of $\mathbf{P}$, which is also the $i$th column of $\mathbf{X}$, and $\mathbf{z}_i$ is the $m$-row vector corresponding to the $i$th row of $\mathbf{X}^{-1}$. Thus we have:

$$\mathbf{A}_i = \begin{bmatrix} z_{i1}\mathbf{x}_i & z_{i2}\mathbf{x}_i & z_{i3}\mathbf{x}_i & \cdots & z_{im}\mathbf{x}_i \end{bmatrix} \tag{3.87}$$

The following two theorems discuss some interesting properties of the expansion matrices $\mathbf{A}_i$.

**Theorem 3.5.** *Matrix* $\mathbf{A}_1$ *is column stochastic and all its columns are identical.*

*Proof.* $\mathbf{A}_1$ is found from the Eq. (3.87) as

$$\mathbf{A}_1 = \begin{bmatrix} z_{11}\mathbf{x}_1 & z_{12}\mathbf{x}_1 & z_{13}\mathbf{x}_1 & \cdots & z_{1m}\mathbf{x}_1 \end{bmatrix} \tag{3.88}$$

But Lemma 3.3 on page 95 proved that $z_{11}, z_{12}, \cdots, z_{1m}$ are all equal to unity. Thus the above equation becomes:

$$\mathbf{A}_1 = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_1 & \cdots & \mathbf{x}_1 \end{bmatrix} \tag{3.89}$$

This proves the theorem. □

Theorem 3.5 results in the following very useful lemma.

**Lemma 3.4.** *The steady state distribution vector* $\mathbf{s}(\infty) = \mathbf{s}$ *must be independent of the initial value* $\mathbf{s}(0)$ *and equals any column of* $\mathbf{A}_1$.

The proof of this lemma is found in Problem 3.53. □

From Theorem 3.5 and Lemma 3.4 we can state the following very useful lemma.

**Lemma 3.5.** *Each column of* $\mathbf{A}_1$ *represents the steady-state distribution vector for the Markov chain.* □

We define a *differential matrix* as a matrix in which the sum of each column is zero. The following theorem states that matrices $\mathbf{A}_i$ corresponding to eigenvalues $\lambda_i \neq 1$ are all differential matrices.

**Theorem 3.6.** *The expansion matrices* $\mathbf{A}_i$ *corresponding to eigenvalues* $\lambda_i \neq 1$ *are differential; i.e. have* $\sigma(A_i) = 0$.

*Proof.* The sum of column $j$ in (3.80) is given by:

$$\sigma_j(\mathbf{P}^n) = \sigma_j(\mathbf{A}_1) + \lambda_2^n \, \sigma_j(\mathbf{A}_2) + \lambda_3^n \, \sigma_j(\mathbf{A}_3) + \cdots \tag{3.90}$$

Lemma 3.1 on page 87 assures us that $\mathbf{P}^n$ is column stochastic and Theorem 3.5 on page 97 assures us that $\mathbf{A}_1$ is also column stochastic. Therefore, we can write the above equation as:

$$1 = 1 + \lambda_2^n \, \sigma_j(\mathbf{A}_2) + \lambda_3^n \, \sigma_j(\mathbf{A}_3) + \cdots \tag{3.91}$$

Since this equation is valid for all values of $\lambda_i$ and $n$, we must have:

$$\sigma_j(\mathbf{A}_2) = \sigma_j(\mathbf{A}_3) = \cdots = \sigma_j(\mathbf{A}_m) = 0 \tag{3.92}$$

The above equations are valid for all values of $1 \leq j \leq m$. Thus all the matrices $\mathbf{A}_i$ which correspond to $\lambda_i \neq 1$ are all differential matrices. And we can write:

$$\sigma(\mathbf{A}_2) = \sigma(\mathbf{A}_3) = \cdots = \sigma(\mathbf{A}_m) = 0 \tag{3.93}$$

This proves the theorem. □

The following theorem is related to Theorem 3.2 on page 79. The theorem essentially explains the effect of premultiplying any matrix by a differential matrix.

**Theorem 3.7.** *Given any matrix* $\mathbf{A}$ *and a differential matrix* $\mathbf{V}$*, then matrix* $\mathbf{B} = \mathbf{VA}$ *will be a differential matrix.*

*Proof.* When $\mathbf{A}$ is premultiplied by $\mathbf{V}$ matrix $\mathbf{B}$ results

$$\mathbf{B} = \mathbf{VA} \tag{3.94}$$

Element $b_{ij}$ is given by the usual matrix product formula:

$$b_{ij} = \sum_{k=1}^{m} v_{ik} \, a_{kj} \tag{3.95}$$

The sum of the $j$th column of matrix $\mathbf{B}$ is denoted by $\sigma_j(\mathbf{B})$ and is given by

$$\sigma_j(\mathbf{B}) = \sum_{i=1}^{m} b_{ij} = \sum_{i=1}^{m} \sum_{k=1}^{m} v_{ik}\, a_{kj} \tag{3.96}$$

Now reverse the order of summation on the right-hand side of the above equation:

$$\sigma_j(\mathbf{B}) = \sum_{i=1}^{m} b_{ij} = \sum_{k=1}^{m} a_{kj} \sum_{i=1}^{m} v_{ik} \tag{3.97}$$

Because $\mathbf{V}$ is differential we have:

$$\sigma_j(\mathbf{V}) = \sum_{i=1}^{m} v_{ik} = 0 \tag{3.98}$$

Therefore, (3.97) becomes:

$$\begin{aligned} \sigma_j(\mathbf{B}) &= \sum_{i=1}^{m} b_{ij} \\ &= \sum_{k=1}^{m} a_{kj} \times 0 \\ &= 0 \end{aligned} \tag{3.99}$$

Thus we proved that sum of columns of a matrix becomes zero when the matrix is premultiplied by a differential matrix. □

*Example 3.17.* The following is a diagonalizable state matrix.

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.3 & 1 \\ 0.2 & 0.3 & 0 \\ 0.7 & 0.4 & 0 \end{bmatrix}$$

We would like to express $\mathbf{P}^n$ in the form in (3.80).

First thing is to check that $\mathbf{P}$ is diagonalizable by checking that it has three distinct eigenvalues. Having assured ourselves that this is the case, we use the MATLAB function `select` that we developed, we find that:

$$\mathbf{P}^n = \mathbf{A}_1 + \lambda_2^n \mathbf{A}_2 + \lambda_3^n \mathbf{A}_3$$

where $\lambda_1 = 1$ and

$$\mathbf{A}_1 = \begin{bmatrix} 0.476 & 0.476 & 0.476 \\ 0.136 & 0.136 & 0.136 \\ 0.388 & 0.388 & 0.388 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} 0.495 & 0.102 & -0.644 \\ -0.093 & -0.019 & 0.121 \\ -0.403 & -0.083 & 0.524 \end{bmatrix}$$

$$\mathbf{A}_3 = \begin{bmatrix} 0.028 & -0.578 & 0.168 \\ -0.043 & 0.883 & -0.257 \\ 0.015 & -0.305 & 0.089 \end{bmatrix}$$

Notice that matrix $\mathbf{A}_1$ is column stochastic and all its columns are equal. Notice also that the sum of columns for matrices $\mathbf{A}_2$ and $\mathbf{A}_3$ is zero. ■

Sometimes two eigenvalues are the complex conjugate of each other. In that case the corresponding eigenvectors will be complex conjugate and so are the corresponding matrices $\mathbf{A}_i$ such that the end result $\mathbf{P}^{(n)}$ is purely real.

*Example 3.18.* A Markov chain has the transition matrix:

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0.4 & 0.2 \\ 0.1 & 0.4 & 0.6 \\ 0.7 & 0.2 & 0.2 \end{bmatrix}$$

Check to see if this matrix has distinct eigenvalues. If so:

(a) Expand the transition matrix $\mathbf{P}^n$ in terms of its eigenvalues.
(b) Find the value of $\mathbf{s}(3)$ using the expression in (3.80).

The eigenvalues of $\mathbf{P}$ are $\lambda_1 = 1$, $\lambda_2 = -0.1 + j0.3$, and $\lambda_3 = -0.1 - j0.3$. We see that complex eigenvalues appear as complex conjugate pairs.

The eigenvectors corresponding to these eigenvalues are:

$$\mathbf{X} = \begin{bmatrix} 0.476 & -0.39 - j0.122 & -0.39 + j0.122 \\ 0.660 & 0.378 - j0.523 & 0.378 + j0.523 \\ 0.581 & 0.011 + j0.645 & 0.011 - j0.645 \end{bmatrix}$$

We see that the eigenvectors corresponding to the complex eigenvalues appear also as complex conjugate pairs.

Using the function `select(P)`, we express $\mathbf{P}^n$ according to: (3.80)

$$\mathbf{P}^n = \mathbf{A}_1 + \lambda_2^n \mathbf{A}_2 + \lambda_3^n \mathbf{A}_3$$

where

$$\mathbf{A}_1 = \begin{bmatrix} 0.277 & 0.277 & 0.277 \\ 0.385 & 0.385 & 0.385 \\ 0.339 & 0.339 & 0.339 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} 0.362 + j0.008 & -0.139 - j0.159 & -0.139 + j0.174 \\ -0.192 + j0.539 & 0.308 - j0.128 & -0.192 - j0.295 \\ -0.169 - j0.546 & -0.169 + j0.287 & 0.331 + j0.121 \end{bmatrix}$$

$$\mathbf{A}_3 = \begin{bmatrix} 0.362 - j0.008 & -0.139 + j0.159 & -0.139 - j0.174 \\ -0.192 - j0.539 & 0.308 + j0.128 & -0.192 + j0.295 \\ -0.169 + j0.546 & -0.169 - j0.287 & 0.331 - j0.121 \end{bmatrix}$$

We see that the expansion coefficients corresponding to the complex eigenvectors appear as complex conjugate pairs. This ensures that all the components of the distribution vector will always be real numbers. The distribution vector at time step $n = 3$ is:

$$\begin{aligned} \mathbf{s}(3) &= \mathbf{P}^3\, \mathbf{s}(0) \\ &= \left[\mathbf{A}_1 + \lambda_2^3 \mathbf{A}_2 + \lambda_3^3 \mathbf{A}_3\right] \mathbf{s}(0) \\ &= \left[\, 0.285 \;\; 0.389 \;\; 0.326 \,\right]^t \end{aligned}$$ ■

*Example 3.19.* Consider the on–off source example whose transition matrix was given by:

$$\mathbf{P} = \begin{bmatrix} s & 1-a \\ 1-s & a \end{bmatrix} \tag{3.100}$$

express this matrix in diagonal form and find an expression for the $n$th power of $\mathbf{P}$.

Using MAPLE or MATLAB's SYMBOLIC packages, the eigenvectors for this matrix are:

$$x_1 = \left[\, (1-a) \,/\, (1-s) \;\; 1 \,\right]^t; \qquad \text{with } \lambda_0 = 1 \tag{3.101}$$

$$x_2 = \left[\, -1 \;\; 1 \,\right]^t; \qquad \text{with } \lambda_1 = s + a - 1 \tag{3.102}$$

Thus we have:

$$\mathbf{X} = \begin{bmatrix} (1-a)/(1-s) & -1 \\ 1 & 1 \end{bmatrix} \tag{3.103}$$

$$\mathbf{X}^{-1} = \frac{1}{\alpha} \begin{bmatrix} 1-s & 1-s \\ s-1 & 1-a \end{bmatrix} \tag{3.104}$$

where $\alpha = (s + a - 1)$. And $\mathbf{P}^n$ is given by:

$$\mathbf{P}^n = \mathbf{X}\begin{bmatrix} 1 & 0 \\ 0 & \alpha^n \end{bmatrix}\mathbf{X}^{-1} \tag{3.105}$$

$$= \frac{1}{\alpha}\begin{bmatrix} 1-a-\alpha^n(s-1) & (1-\alpha^n)(1-a) \\ (1-\alpha^n)(1-s) & 1-s+\alpha^n(1-a) \end{bmatrix} \tag{3.106}$$

We can write $\mathbf{P}^n$ in the summation form:

$$\mathbf{P}^n = \mathbf{A}_1 + \alpha^n \mathbf{A}_2 \tag{3.107}$$

where

$$\mathbf{A}_1 = \frac{1}{2-a-s}\begin{bmatrix} 1-a & 1-a \\ 1-s & 1-s \end{bmatrix} \tag{3.108}$$

$$\mathbf{A}_2 = \frac{1}{2-a-s}\begin{bmatrix} 1-s & a-1 \\ s-1 & 1-a \end{bmatrix} \tag{3.109}$$

The observations on the properties of the matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ can be easily verified.

For large values of $n$, we get the simpler form:

$$\lim_{n\to\infty} \mathbf{P}^n = \mathbf{A}_1 = \frac{1}{2-a-s}\begin{bmatrix} 1-a & 1-a \\ 1-s & 1-s \end{bmatrix} \tag{3.110}$$

and in the steady state our distribution vector becomes:

$$\mathbf{s} = \frac{1}{2-a-s}\begin{bmatrix}(1-a) & (1-s)\end{bmatrix}^t$$

and this is independent of the initial state of our source.

*Example 3.20.* Consider the column stochastic matrix:

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.8 & 0.4 \\ 0.5 & 0 & 0.3 \\ 0 & 0.2 & 0.3 \end{bmatrix}.$$

This matrix is diagonalizable. Express it in the form given in (3.63) and obtain a general expression for the distribution vector at step $n$. What would be the distribution vector after 100 steps assuming an initial distribution vector $\mathbf{s}(0) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^t$

The eigenvalues for this matrix are: $\lambda_1 = 1$, $\lambda_2 = -0.45$, $\lambda_3 = 0.25$. Since they are distinct, we know that the matrix is diagonalizable. The eigenvectors

corresponding to these eigenvalues are:

$$\mathbf{x}_1 = \begin{bmatrix} 0.869 & 0.475 & 0.136 \end{bmatrix}^t$$
$$\mathbf{x}_2 = \begin{bmatrix} -0.577 & 0.789 & -0.211 \end{bmatrix}^t$$
$$\mathbf{x}_3 = \begin{bmatrix} -0.577 & -0.211 & 0.789 \end{bmatrix}^t$$

and matrix $\mathbf{X}$ in (3.63) is simply $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 \end{bmatrix}$ which is given by

$$\mathbf{X} = \begin{bmatrix} 0.869 & -0.577 & -0.577 \\ 0.475 & 0.789 & -0.212 \\ 0.136 & -0.212 & 0.789 \end{bmatrix}$$

Notice that the sum of the elements in the second and third columns is exactly zero. We also need the inverse of this matrix which is:

$$\mathbf{X}^{-1} = \begin{bmatrix} 0.676 & 0.676 & 0.676 \\ -0.473 & 0.894 & -0.106 \\ -0.243 & 0.123 & 1.123 \end{bmatrix}$$

Thus we can express $\mathbf{P}^n$ as:

$$\begin{aligned} \mathbf{P}^n &= \mathbf{X} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \lambda_2^n & 0 \\ 0 & 0 & \lambda_3^n \end{bmatrix} \mathbf{X}^{-1} \\ &= \mathbf{A}_1 + \lambda_2^n \mathbf{A}_2 + \lambda_3^n \mathbf{A}_3 \end{aligned}$$

where the three matrices $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$ are given using the following expressions:

$$\mathbf{A}_1 = \mathbf{X} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{X}^{-1} \tag{3.111}$$

$$\mathbf{A}_2 = \mathbf{X} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{X}^{-1} \tag{3.112}$$

$$\mathbf{A}_3 = \mathbf{X} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{X}^{-1} \tag{3.113}$$

The above formulas produce:

$$\mathbf{A}_1 = \begin{bmatrix} 0.587 & 0.587 & 0.587 \\ 0.321 & 0.321 & 0.321 \\ 0.092 & 0.092 & 0.092 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} 0.273 & -0.516 & 0.062 \\ -0.373 & 0.705 & -0.084 \\ 0.1 & -0.189 & 0.022 \end{bmatrix}$$

$$\mathbf{A}_3 = \begin{bmatrix} 0.141 & -0.071 & -0.649 \\ 0.051 & -0.026 & -0.237 \\ -0.192 & 0.097 & 0.886 \end{bmatrix}$$

We notice that all the columns in $\mathbf{A}_1$ are identical while the sum of the columns in $\mathbf{A}_2$ and $\mathbf{A}_3$ is zero. These properties of matrices $\mathbf{A}_i$ guarantee that $\mathbf{P}^n$ remains a column stochastic matrix for all values of $n$ between 0 and $\infty$.

Since each $\lambda$ is a fraction, we see that the probabilities of each state consist of a steady-state solution (independent of $\lambda$) and a transient solution that contains the different $\lambda's$. In the steady-state (i.e., when $n \to \infty$) we have the distribution vector that equals any column of $\mathbf{A}_1$:

$$\mathbf{s}(\infty) = \begin{bmatrix} 0.587 & 0.321 & 0.092 \end{bmatrix}^t \tag{3.114}$$

Thus we get after 100 steps

$$\begin{aligned} \mathbf{s}(100) &= \mathbf{P}^{100}\,\mathbf{s}(0) \\ &= \begin{bmatrix} 0.6 + 0.3 \times \lambda_2^{100} + 0.1 \times \lambda_3^{100} \\ 0.3 - 0.4\lambda_2^{100} \\ 0.1 + 0.1\lambda_2^{100} - 0.2\lambda_3^{100} \end{bmatrix} \\ &= \begin{bmatrix} 0.587 & 0.321 & 0.092 \end{bmatrix}^t \end{aligned}$$

Thus the distribution vector settles down to its steady-state value irrespective of the initial state. ■

### *3.13.1 Test for Matrix Diagonalizability*

The above two techniques: expanding $\mathbf{s}(0)$ and diagonalizing $\mathbf{P}$ both relied on the fact that $\mathbf{P}$ could be diagonalized. We mentioned earlier that a matrix could be diagonalized when its eigenvalues are all distinct. While this is certainly true, there is a more general test for the diagonalizability of a matrix.

**A matrix is diagonalizable only when its Jordan canonic form (JCF) is diagonal [2].**

We will explore this topic more in the next section.

*Example 3.21.* Indicate whether the matrix in Example 3.6 on page 73 is diagonalizable or not.

The eigenvalues of this matrix are: $\lambda_1 = 1$ and $\lambda_2 = \lambda_3 = -1/4$. Since some eigenvalues are repeated, **P** might or might not be diagonalizable. Using Maple or the MATLAB function `JORDAN(P)`, the Jordan canonic form for this matrix is:

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1/4 & 1 \\ 0 & 0 & -1/4 \end{bmatrix}$$

Since **J** is not diagonal, **P** is not diagonalizable. It is as simple as that! ■

## 3.14 Finding s($n$) Using the Jordan Canonic Form

We saw in Sect. 3.12 how easy it was to find **s**($n$) by diagonalizing **P**. We would like to follow the same lines of reasoning even when **P** cannot be diagonalized because one or more of its eigenvalues are repeated.

### *3.14.1 Jordan Canonic Form (JCF)*

Any $m \times m$ matrix **P** can be transformed through a similarity transformation into its Jordan canonic form (JCF) [2]:

$$\mathbf{P} = \mathbf{U}\mathbf{J}\mathbf{U}^{-1} \tag{3.115}$$

Matrix **U** is a nonsingular matrix and $J$ is a block-diagonal matrix:

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 & 0 & 0 & \cdots & 0 \\ 0 & \mathbf{J}_2 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{J}_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{J}_t \end{bmatrix} \tag{3.116}$$

where the matrix $\mathbf{J}_i$ is an $m_i \times m_i$ *Jordan block* or *Jordan box* matrix of the form

$$\mathbf{J}_i = \begin{bmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & \cdots & 0 \\ 0 & 0 & \lambda_i & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & \lambda_i \end{bmatrix} \tag{3.117}$$

such that the following equation holds

$$m_1 + m_2 + \cdots + m_t = m \tag{3.118}$$

The similarity transformation performed in (3.115) above ensures that the eigenvalues of the two matrices **P** and **J** are identical. The following example proves this statement.

*Example 3.22.* Prove that if the Markov matrix **P** has Jordan canonic form matrix **J**, then the eigenvalues of **P** are identical to those of **J**. Having proved that, find the relation between the corresponding eigenvectors for both matrices.

We start first by proving that the characteristic equations for both matrices are identical. The characteristic equation or characteristic polynomial of **P** is given by:

$$x(\lambda) = \det(\mathbf{P} - \lambda\mathbf{I}) \tag{3.119}$$

where det(**A**) is the determinant of the matrix **A**. The characteristic polynomial of **J** is given by:

$$y(\alpha) = \det(\mathbf{J} - \alpha\mathbf{I}) \tag{3.120}$$

where $\alpha$ is the assumed root or eigenvalue of **J**. But (3.115) indicates that **J** can be expressed in terms of **P** and **U** as:

$$\mathbf{J} = \mathbf{U}^{-1}\mathbf{P}\mathbf{U}$$

Using this, we can express (3.120) as

$$y(\alpha) = \det\left(\mathbf{U}^{-1}\mathbf{P}\mathbf{U} - \alpha\mathbf{U}^{-1}\mathbf{U}\right)$$

We can factor out the matrices $\mathbf{U}^{-1}$ and **U** to get:

$$y(\alpha) = \det\left[\mathbf{U}^{-1}(\mathbf{P} - \alpha\mathbf{I})\,\mathbf{U}\right]$$

But the rules of determinants indicate that $\det(\mathbf{AB}) = \det(\mathbf{A}) \times \det(\mathbf{B})$. Thus we have

$$y(\alpha) = \det\left(\mathbf{U}^{-1}\right)\ \det(\mathbf{U})\ \det(\mathbf{P} - \alpha\mathbf{I})$$

But the rules of determinants also indicate that $\det(\mathbf{A}^{-1}) \times \det(\mathbf{A}) = 1$. Thus we have:

$$y(\alpha) = \det(\mathbf{P} - \alpha\mathbf{I}) \tag{3.121}$$

This proves that the polynomial $x(\lambda)$ in (3.119) and the polynomial $y(\alpha)$ in (3.121) are identical and the matrices **P** and **J** have the same roots or eigenvalues.

So far, we proved that the two matrices **P** and **J** have the same roots or eigenvalues. Now we want to prove that they have related eigenvectors. Assume **x** is an eigenvector of **P** associated with eigenvalue $\lambda$

$$\mathbf{P}\mathbf{x} = \lambda\mathbf{x}$$

but $\mathbf{P} = \mathbf{U}\mathbf{J}\mathbf{U}^{-1}$ and we can write the above equation as

$$\mathbf{U}\mathbf{J}\mathbf{U}^{-1}\,\mathbf{x} = \lambda\mathbf{x}$$

Now we premultiply both sides of the equation by $\mathbf{U}^{-1}$ to get

$$\mathbf{J}\mathbf{U}^{-1}\,\mathbf{x} = \lambda\mathbf{U}^{-1}\mathbf{x}$$

We write $\mathbf{u} = \mathbf{U}^{-1}\mathbf{x}$ to express the above equation as

$$\mathbf{J}\,\mathbf{u} = \lambda\mathbf{u}$$

Thus the relation between the eigenvectors of **P** and **J** is

$$\mathbf{u} = \mathbf{U}^{-1}\mathbf{x}$$

This is why the transformation

$$\mathbf{P} = \mathbf{U}\mathbf{J}\mathbf{U}^{-1}$$

is called a *similarity transformation*. In other words, a similarity transformation does not change the eigenvalues and scales the eigenvectors. ■

MATLAB offers the command `[U,J] = JORDAN(P)` to obtain the Jordan canonic form as the following example shows.

*Example 3.23.* Obtain the Jordan canonic form for the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 0.25 & 0.25 \\ 0.75 & 0 & 0.25 \\ 0.25 & 0.75 & 0.5 \end{bmatrix}$$

Using MATLAB we are able to obtain the Jordan canonic form:

```
[U,J] = jordan(P)
U =
    0.2000    0.0000    0.8000
    0.2800    0.4000   -0.2800
    0.5200   -0.4000   -0.5200
J =
    1.0000         0         0
         0   -0.2500    1.0000
         0         0   -0.2500
```

■

### 3.14.2 Properties of Jordan Canonic Form

We make the following observations on the properties of the Jordan canonic from:

1. The number of Jordan blocks equals the number of linearly independent eigenvectors of **P**.
2. The elements in **P** are real but matrix **U** might have complex elements.
3. Matrix **U** can be written as

$$\mathbf{U} = \left[ \mathbf{U}_1 \; \mathbf{U}_2 \cdots \mathbf{U}_t \right]$$

   where each matrix $\mathbf{U}_i$ is a rectangular matrix of dimension $m \times m_i$ such that

$$m_1 + m_2 + \cdots + m_t = m$$

4. Rectangular matrix $\mathbf{U}_i$ corresponds to the Jordan block $\mathbf{J}_i$ and eigenvalue $\lambda_i$.
5. Each rectangular matrix $\mathbf{U}_i$ can be decomposed into $m_i$ column vectors:

$$\mathbf{U}_i = \left[ \mathbf{u}_{i,1} \; \mathbf{u}_{i,2} \cdots \mathbf{u}_{i,m_i} \right] \tag{3.122}$$

   where each column vector has $m$ components.
6. The first column $\mathbf{u}_{i,1}$ is the eigenvector of matrix **P** and corresponds to the eigenvalue $\lambda_i$:

$$\mathbf{P}\,\mathbf{u}_{i,1} = \lambda_i\,\mathbf{u}_{i,1} \tag{3.123}$$

7. The other column vectors of $\mathbf{U}_i$ satisfy the recursive formula:

$$\mathbf{P}\,\mathbf{u}_{i,j} = \lambda_i\,\mathbf{u}_{i,j} + \mathbf{u}_{i,j-1} \tag{3.124}$$

   where $2 \leq j \leq m_i$.
8. There could be one or more blocks having the same eigenvalue. In other words, we could have two Jordan blocks $\mathbf{J}_1$ and $\mathbf{J}_2$ such that both have the same eigenvalue on their main diagonals.
9. The eigenvalue $\lambda_i$ is said to have algebraic multiplicity of $m_i$.
10. If all Jordan blocks are one-dimensional (i.e., all $m_i = 1$), then the Jordan matrix $\mathbf{J}$ becomes diagonal. In that case, matrix $\mathbf{P}$ is diagonalizable.

*Example 3.24.* Given the following Jordan matrix, identify the Jordan blocks and find the eigenvalues and the number of linearly independent eigenvectors.

$$\mathbf{J} = \begin{bmatrix} 0.5 & 1 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & 1 & 0 \\ 0 & 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

We have three Jordan blocks as follows:

$$\mathbf{J}_1 = \begin{bmatrix} 0.5 & 1 \\ 0 & 0.5 \end{bmatrix}$$

$$\mathbf{J}_2 = \begin{bmatrix} 0.2 & 1 \\ 0 & 0.2 \end{bmatrix}$$

$$\mathbf{J}_3 = [1]$$

From the three Jordan blocks, we determine the eigenvalues as $\lambda_1 = 0.5$, $\lambda_2 = 0.2$, and $\lambda_3 = 1$. We also know that we must have three linearly independent eigenvectors.

Using MATLAB, the eigenvectors of $\mathbf{J}$ are:

$$\mathbf{x}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}^t$$

$$\mathbf{x}_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \end{bmatrix}^t$$

$$\mathbf{x}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}^t$$

we notice that these eigenvectors are linearly independent because they are orthogonal to each other. ■

## 3.15 Properties of Matrix U

According to Theorem 3.3 on page 80, Theorem 3.4 on page 81, and Lemma 3.3 on page 95, the columns of matrix $\mathbf{U}$ must satisfy the following properties:

1. The sum of the elements of the vector $\mathbf{u}$ corresponding to the eigenvalue $\lambda = 1$ is arbitrary and could be taken as unity, i.e. $\sigma(\mathbf{u}) = 1$.
2. The sum of the elements of the vectors $\mathbf{u}$ belonging to Jordan blocks with eigenvalue $\lambda \neq 1$ must be zero, i.e. $\sigma(u) = 0$.
3. Matrix $\mathbf{U}^{-1}$ has ones in its first row.

## 3.16 $\mathbf{P}^n$ Expressed in Jordan Canonic Form

We explained in the previous subsection that the transition matrix could be expressed in terms of its Jordan canonic form which we repeat here for convenience:

$$\mathbf{P} = \mathbf{U}\mathbf{J}\mathbf{U}^{-1} \tag{3.125}$$

where $\mathbf{U}$ is a unitary matrix. Equation 3.125 results in a very simple expression for $\mathbf{P^n}$ in (3.38). To start, we can calculate $\mathbf{P}^2$ as:

$$\mathbf{P}^2 = \left(\mathbf{U}\mathbf{J}\mathbf{U}^{-1}\right) \times \left(\mathbf{U}\mathbf{J}\mathbf{U}^{-1}\right) \tag{3.126}$$

$$= \mathbf{U}\mathbf{J}^2\mathbf{U}^{-1} \tag{3.127}$$

In general we have:

$$\mathbf{P}^n = \mathbf{U}\mathbf{J}^n\mathbf{U}^{-1} \tag{3.128}$$

where $\mathbf{J}^n$ has the same block structure as $\mathbf{J}$:

$$\mathbf{J}^n = \begin{bmatrix} \mathbf{J}_1^n & 0 & 0 & \cdots & 0 \\ 0 & \mathbf{J}_2^n & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{J}_3^n & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{J}_t^n \end{bmatrix} \tag{3.129}$$

In the above equation, the Jordan block $\mathbf{J}_i^n$ of dimension $m_i \times m_i$ is an upper triangular Toeplitz matrix in the form:

$$\mathbf{J}_i^n = \begin{bmatrix} f_{i0}^n & f_{i1}^n & f_{i2}^n & f_{i3}^n & \cdots & f_{i,m_i-1}^n \\ 0 & f_{i0}^n & f_{i1}^n & f_{i2}^n & \cdots & f_{i,m_i-2}^n \\ 0 & 0 & f_{i0}^n & f_{i1}^n & \cdots & f_{i,m_i-3}^n \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & f_{i0}^n \end{bmatrix} \tag{3.130}$$

where $f_{ij}^n$ is given by:

$$f_{ij}^n = \binom{n}{j} \lambda_i^{n-j} \qquad 0 \le j < m_i \tag{3.131}$$

We assumed the binomial coefficient vanishes whenever $j > n$.

In fact, the term $f_{ij}^n$ equals the $j$th term in the binomial expansion:

$$(\lambda_i + 1)^n = \sum_{j=0}^{m_i-1} f_{ij}^n \tag{3.132}$$

## 3.17 Expressing P$^n$ in terms of its Eigenvalues

Equation (3.128) related $\mathbf{P}^n$ to the $n$th power of its Jordan canonic form as:

$$\mathbf{P}^n = \mathbf{U}\mathbf{J}^n\mathbf{U}^{-1} \tag{3.133}$$

Thus we can express $\mathbf{P}^n$ in the above equation in the form:

$$\mathbf{P}^n = \sum_{j=0}^{m_1-1} f_{1j}^n \mathbf{A}_{1j} + \sum_{j=0}^{m_1-1} f_{2j}^n \mathbf{A}_{2j} + \sum_{j=0}^{m_2-1} f_{3j}^n \mathbf{A}_{3j} + \cdots \tag{3.134}$$

The above equation can be represented as the double summation:

$$\mathbf{P}^n = \sum_{i=1}^{t} \sum_{j=0}^{m_i-1} f_{ij}^n \, \mathbf{A}_{ij} \tag{3.135}$$

where $t$ is the number of Jordan blocks and it was assumed that $f_{ij}^n$ is zero whenever $j > n$.

The matrices $\mathbf{A}_{ij}$ can be determined from the product:

$$\mathbf{A}_{ij} = \mathbf{U}\,\mathbf{Y}_{ij}\,\mathbf{U}^{-1} \tag{3.136}$$

where $\mathbf{Y}_{ij}$ is the selection matrix which has zeros everywhere except for the elements corresponding to the superdiagonal $j$ in Jordan block $i$ which contain the values 1. For example, assume a $6 \times 6$ Markov matrix whose Jordan canonic form is:

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix}$$

This Jordan canonic form has three Jordan blocks ($t = 3$) and the selection matrix $\mathbf{Y}_{21}$ indicates that we need to access the first superdiagonal of the second Jordan block:

$$\mathbf{Y}_{21} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

**WWW** We have prepared the MATLAB function `J_POWERS(n,P)` that accepts a Markov matrix (or any square matrix) and expresses the matrix $\mathbf{P}^n$ in the form given by (3.134).

*Example 3.25.* Consider the Markov matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 0.25 & 0.4 \\ 0.75 & 0 & 0.3 \\ 0.25 & 0.75 & 0.3 \end{bmatrix}$$

Use the Jordan canonic form technique to find the decomposition of $\mathbf{P}^3$ according to (3.134).

We start by finding the Jordan canonic form for the given matrix to determine whether $\mathbf{P}$ can be diagonalized or not. The Jordan decomposition produces

$$\mathbf{U} = \begin{bmatrix} 0.20 & 0.00 & 0.80 \\ 0.44 & 0.20 & -0.44 \\ 0.36 & -0.20 & -0.36 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.25 & 1 \\ 0 & 0 & -0.25 \end{bmatrix}$$

The given Markov matrix has two eigenvalues $\lambda_1 = 1$ and $\lambda_2 = -0.25$, which is repeated twice. Since the Jordan matrix is not diagonal, we cannot diagonalize the transition matrix and we need to use the Jordan decomposition techniques to find $\mathbf{P}^n$. Using the function `J_POWERS` we get

$$\mathbf{P}^3 = \mathbf{A}_1 + f_{20}^3 \mathbf{A}_{20} + f_{21}^3 \mathbf{A}_{21}$$

where

$$f_{20}^3 = 0.0156$$
$$f_{21}^3 = 0.187$$

and the corresponding matrices are given by

$$\mathbf{A}_1 = \begin{bmatrix} 0.20 & 0.20 & 0.20 \\ 0.44 & 0.44 & 0.44 \\ 0.36 & 0.36 & 0.36 \end{bmatrix}$$

$$\mathbf{A}_{20} = \begin{bmatrix} 0.80 & -0.20 & -0.20 \\ -0.44 & 0.56 & -0.44 \\ -0.36 & -0.36 & 0.64 \end{bmatrix}$$

$$\mathbf{A}_{21} = \begin{bmatrix} 0 & 0 & 0 \\ 0.2 & -0.05 & -0.05 \\ -0.2 & 0.05 & 0.05 \end{bmatrix}$$

■

## 3.18 Finding P$^n$ Using the Z-Transform

The z-transform technique has been proposed for finding expressions for $\mathbf{P}^n$. In our opinion, this technique is not useful for the following reasons. Obtaining the z-transform is very tedious since it involves finding the inverse of a matrix using symbolic, not numerical, techniques. This is really tough for any matrix whose dimension is above $2 \times 2$ even when symbolic arithmetic packages are used. Furthermore, the technique will not offer any new insights that have not been already covered in this chapter. For that reason, we delegate discussion of this topic to Appendix C on page 551. The interested reader can gloss over the appendix and compare it to the techniques developed in this chapter.

## 3.19 Renaming the States

Sometimes we will need to rename or relabel the states of a Markov chain. When we do that, the transition matrix will assume a simple form that helps in understanding the behavior of the system.

Renaming or relabeling the states amounts to exchanging the rows and columns of the transition matrix. For example, if we exchange states $s_2$ and $s_5$, then rows 2 and 5 as well as columns 2 and 5 will be exchanged. We perform this rearranging through the elementary exchange matrix $\mathbf{E}(2,5)$ which exchanges states 2 and 5:

$$\mathbf{E}(2,5) = \begin{bmatrix} 1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1 \\ 0\,0\,1\,0\,0 \\ 0\,0\,0\,1\,0 \\ 0\,1\,0\,0\,0 \end{bmatrix}$$

In general, the exchange matrix $\mathbf{E}(i, j)$ is similar to the identity matrix except that rows $i$ and $j$ of the identity matrix are exchanged.

The exchange of states is achieved by pre and post multiplying the transition matrix:

$$\mathbf{P}' = \mathbf{E}(2,5)\ \mathbf{P}\ \mathbf{E}(2,5)$$

Assume, for example, that $\mathbf{P}$ is given by

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0.1 & 0 & 0.1 & 1 \\ 0 & 0.3 & 0 & 0.2 & 0 \\ 1 & 0.2 & 0 & 0.2 & 0 \\ 0 & 0.3 & 0 & 0.4 & 0 \\ 0 & 0.1 & 1 & 0.1 & 0 \end{bmatrix} \end{array}$$

where the state indices are indicated around $\mathbf{P}$ for illustration.

Exchanging states 2 and 5 results in

$$\mathbf{P}' = \begin{array}{c} \\ 1 \\ 5 \\ 3 \\ 4 \\ 2 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0.1 & 0.1 \\ 0 & 0 & 1 & 0.1 & 0.1 \\ 1 & 0 & 0 & 0.2 & 0.2 \\ 0 & 0 & 0 & 0.4 & 0.1 \\ 0 & 0 & 0 & 0.2 & 0.3 \end{bmatrix} \end{array}$$

## 3.20 Problems

### *Markov Chains*

**3.1.** Consider Example 3.2 where the time step value is chosen as $T = 8/\lambda_a$. Estimate the packet arrival probability $a$.

**3.2.** Three workstation clusters are connected to each other using switching hubs. At steady state the following daily traffic share of each hub was observed. For hub A, 80 % of its traffic is switched to its local cluster, 5 % of its traffic is switched to hub B, and 15 % of its traffic is switched to hub C. For hub B, 90 % of its traffic is switched to its local cluster, 5 % of its traffic is switched to hub A, and 5 % of its traffic is switched to hub C. For hub C, 75 % of its traffic is switched to its local cluster, 10 % of its traffic is switched to hub A, and 15 % of its traffic is switched to hub B. Assume initially the total traffic is distributed among the three hubs as follows 60 % in hub A, 30 % in hub B, and 10 % in hub C.

(a) Write the initial distribution vector for the total traffic.
(b) Construct the transition matrix for the Markov chain that describes the traffic share of the three hubs.

**3.3.** In order to plan the volume of LAN traffic flow in a building, the system administrator divided the building into three floors. Traffic volume trend indicated the following hourly pattern. In the first floor, 60 % of traffic is local, 30 % of traffic goes to second floor, 10 % of traffic goes to third floor. In the second floor, 30 % of traffic is local, 40 % of traffic goes to first floor, 30 % of traffic goes to third floor. In the third floor, 60 % of traffic is local, 30 % of traffic goes to first floor, 10 % of traffic goes to second floor. Assuming initially traffic volume is distributed as follows 10 % is in first floor, 40 % in second floor, and 50 % in third floor.

(a) Write the initial distribution vector for the total traffic.
(b) Construct the transition matrix for the Markov chain that describes the traffic share of the three floors.

**3.4.** The transition matrix for a Markov chain is given by

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.6 \\ 0.7 & 0.4 \end{bmatrix}$$

What does each entry represent?

**3.5.** A traffic data generator could be either idle or is generating data at five different rates $\lambda_1 < \lambda_2 < \cdots < \lambda_5$. When idle, the source could equally likely remain idle or it could start transmitting at the lowest rate $\lambda_1$. When in the highest rate state $\lambda_5$, the source could equally likely remain in that state or it could switch to the next lower rate $\lambda_4$. When in the other states, the source is equally likely to remain in its present state or it could start transmitting at the next lower or higher rate. Identify the system states and write down the transition matrix.

**3.6.** Repeat the above problem when transitions between the different states are equally likely.

**3.7.** The market over reaction theory proposes that stocks with low return (called "losers") subsequently outperform stocks with high return (called "winners") over some observation period. The rest of the market share is stocks with medium return (called "medium"). It was observed that winners split according to the following ratios: 70 % become losers, 25 % become medium, and 5 % stay winners. Medium stocks split according to the following ratios: 5 % become losers, 90 % stay medium, and 5 % become winners. Losers split according to the following ratios: 80 % stay losers, 5 % become medium, and 15 % become winners. The Markov chain representing the state of a stock is defined as follows. $s_1$ represents loser stock, $s_2$ represents medium stock, and $s_3$ represents winner stocks. Assuming an aggressive manager's portfolio is initially split among the stocks in the following percentages: 5 % losers, 70 % medium, and 25 % winners.

(a) Write the initial distribution vector for the portfolio.
(b) Construct the transition matrix for the Markov chain that describes the stock share of the portfolio.

**3.8.** Consider Example 3.8; but this time the source is transmitting packets having random lengths. Assume for simplicity that the transmitted packets can be one out of four lengths selected at random with probability $l_i$ ($i = 1, 2, 3,$ or 4).

(a) Identify the different states of the system.
(b) Draw the corresponding Markov state transition diagram.
(c) Write down the transition matrix.

## *Time Step Selection*

**3.9.** Develop the proper packet arrival statistics for the case considered in Sect. 3.3.1 when the line is sampled at a rate $r$ while the packets arrive at an average rate $\lambda_a$.

## *Markov Transition Matrices*

**3.10.** Explain what is meant by a homogeneous Markov chain.

**3.11.** Assume **P** is a transition matrix. Prove that the unit row vector

$$\mathbf{u} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

is a left eigenvector of the matrix and its associated eigenvalue is $\lambda = 1$.

**3.12.** Prove (3.11) on page 72 which states that at any time step $n$, the sum of the components of the distribution vector $\mathbf{s}(n)$ equals 1. Do your proof by proceeding as follows:

(a) Start with an initial distribution vector $\mathbf{s}(0)$ of an arbitrary dimension $m$ such that $\sum_{i=1}^{m} s_i(0) = 1$.
(b) Prove that $\mathbf{s}(1)$ satisfies (3.11).
(c) Prove that $\mathbf{s}(2)$ also satisfies (3.11) and so on.

**3.13.** Prove the properties stated in Sect. 3.9.1 on page 87.

**3.14.** Prove Lemma 3.1 on page 87.

**3.15.** Given a column stochastic matrix $\mathbf{P}$ with an eigenvector $\mathbf{x}$ that corresponds to the eigenvalue $\lambda = -1$. Prove that $\sigma(x) = 0$ in accordance with Theorem 3.4.

**3.16.** In problems 3.17–3.26 determine which of the given matrices are Markov matrices and justify your answer. For the Markov matrices, determine the eigenvalues, the corresponding eigenvectors, and the rank.

**3.17.**

$$\begin{bmatrix} 0.4 & 0.4 \\ 0.6 & 0.3 \end{bmatrix}$$

**3.18.**

$$\begin{bmatrix} 0.8 & 0.5 \\ 0.3 & 0.5 \end{bmatrix}$$

**3.19.**

$$\begin{bmatrix} -0.1 & 0.8 \\ -0.9 & 0.2 \end{bmatrix}$$

**3.20.**

$$\begin{bmatrix} 1.2 & 0.8 \\ -0.2 & 0.2 \end{bmatrix}$$

**3.21.**

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

**3.22.**

$$\begin{bmatrix} 0.4 & 0.4 \\ 0.6 & 0.3 \end{bmatrix}$$

**3.23.**

$$\begin{bmatrix} 0.1 & 0.3 \\ 0.2 & 0.5 \\ 0.7 & 0.2 \end{bmatrix}$$

**3.24.**

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

**3.25.**

$$\begin{bmatrix} 0.8 & 0.3 & 0.5 \\ 0.2 & 0.7 & 0.5 \end{bmatrix}$$

**3.26.**

$$\begin{bmatrix} 0.7 & 0.3 & 0.5 & 0.3 \\ 0.1 & 0.1 & 0.2 & 0.3 \\ 0.1 & 0.3 & 0.1 & 0.3 \\ 0.1 & 0.3 & 0.2 & 0.3 \end{bmatrix}$$

**3.27.** Choose any column stochastic matrix from the matrices in the above problems, or choose one of your own, then reduce the value of one of its nonzero elements slightly (keeping the matrix nonnegative of course). In that way, the matrix will not be a column stochastic matrix any longer. Observe the change in the maximum eigenvalue and the corresponding eigenvector.

**3.28.** Assume a source is sending packets on a wireless channel. The source could be in one of three states: (1) Idle state. (2) Successful transmission state where source is active and transmitted packet is received without errors. (3) Erroneous transmission state where source is active and transmitted packet is received with errors.

Assume the probability the source switches from idle to active is $a$ and the probability that the source successfully transmits a packet is $s$. Draw a state transition diagram indicating the transition probabilities between states and find the transition matrix.

### *Transient Behavior*

**3.29.** For problem 3.2 how much share of the traffic will be maintained by each hub after 1 day and after 2 days?

**3.30.** For problem 3.3 how much share of the traffic will be maintained by each floor after 1 h and after 2 h?

**3.31.** The transition matrix for a Markov chain is given by

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.1 \\ 0.2 & 0.9 \end{bmatrix}$$

(a) Given that the system is in state $s_1$, what is the probability the next state will be $s_2$?
(b) For the initial distribution vector $\mathbf{s}(0)$

$$\mathbf{s}(0) = \begin{bmatrix} 0.4 & 0.6 \end{bmatrix}^t$$

find $\mathbf{s}(1)$.

**3.32.** The transition matrix for a Markov chain is given by

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.3 & 0.5 \\ 0 & 0.3 & 0.25 \\ 0.5 & 0.4 & 0.25 \end{bmatrix}$$

(a) What does the entry $p_{12}$ represent?
(b) Given that the system is in state $s_1$, what is the probability the next state will be $s_2$?
(c) For the initial distribution vector $\mathbf{s}(0)$

$$\mathbf{s}(0) = \begin{bmatrix} 0.4 & 0.6 & 0 \end{bmatrix}^t$$

find $\mathbf{s}(1)$.

**3.33.** Given a transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0.7 \\ 0.8 & 0.3 \end{bmatrix}$$

what is the probability of making a transition to state $s_1$ given that we are in state $s_2$?

**3.34.** Assume in a hypothetical city where yearly computer buying trends indicate that 95 % of the persons who own a desktop computer will purchase a desktop and the rest will switch to laptops. On the other hand, 60 % of laptop owners will

continue to buy laptops and the rest will switch to desktops. At the beginning of the year 65 % of the computer owners had desktops. What will be the percentages of desktop and laptop owners after 1, 2, and 10 years?

**3.35.** Assume the state of a typical winter day in Cairo to be sunny, bright, or peaceful. Observing the weather pattern reveals the following. When today is sunny, tomorrow will be bright with probability 80 % and peaceful with probability 20 %. When today is bright, tomorrow will be sunny with probability 60 %, bright with probability 30 % and peaceful with probability 10 %. When today is peaceful, tomorrow will be sunny with probability 30 %, bright with probability 40 % and peaceful with probability 30 %. Assume state 1 represents a sunny day, state 2 represents a bright day, and state 3 represents a peaceful day.

(a) Construct a state transition matrix.
(b) What is the probability that it will be peaceful tomorrow given that it is sunny today?
(c) What is the probability that it will be bright day after tomorrow given that it is bright today?

**3.36.** Assume a gambler plays double or nothing game using a fair coin and starting with one dollar.

(a) Draw a state diagram for the amount of money with the gambler and explain how much money corresponds to each state.
(b) Derive the transition matrix.
(c) What is the probability that the gambler will have more than $500 after playing the game for ten tosses of the coin?

**3.37.** Suppose you play the following game with a friend, both of you start with $2. You flip a fair coin for $1 a flip. If the coin comes up heads, you win $1. If the coin comes up tails, you lose $1. The game ends when either of you do not have anymore money.

(a) Construct a Markov transition diagram and transition matrix for this game.
(b) Find the eigenvectors and eigenvalues for this matrix.
(c) What is the initial probability vector when you start the game?

**3.38.** Assume you are playing a truncated form of the snakes and ladder game using a fair coin instead of the dice. The number of squares is assumed to be 10, to make things simple, and each player starts at the first square (we label it square 1 and the last square is labeled 10). Tails mean the player advances one square and heads mean the player advances by two squares. To make the game interesting, some squares have special transitions according to the following rules which indicate the address on the next square upon the flip of the coin.

| Square | Heads | Tails |
|---|---|---|
| 2 | 4 | 2 |
| 3 | 6 | 1 |
| 5 | 7 | 2 |
| 8 | 9 | 4 |

Write the initial distribution vector and the transition matrix. What will be the distribution vector be after five flips of the coin? What are the chances of a player winning the game after ten flips?

**3.39.** Assume a particle is allowed to move on a one-dimensional grid and starts at the middle. The probability of the particle moving to the right is $p$ and to the left is $q$, where $p + q = 1$. Assume the size of the gird to extend from 1 to $N$, with $N$ assumed odd. Assume that at the end points of the grid, the particle is reflected back with probability 1. Draw a Markov transition diagram and write down the corresponding transition matrix. Assume $p = 0.6$ and $q = 0.4$ and $N = 7$. Plot the most probable position for the particle versus time.

**3.40.** A parrot breeder has birds of two colors blue and green. She finds that 60 % of the males are blue if the father was blue and 80 % of the males are green if the father was green. Write down the transition matrix for the parrot males. What is the probability that a blue male has a blue male after two and three generations?

**3.41.** A virus can mutate between $N$ different states (typically $N = 20$). In each generation it either stays the same or mutates to another sate. Assume that the virus is equally likely to change state to any of the $N$ states. You can reduce the size of the transition matrix $\mathbf{P}$ from $N \times N$ to 2×2 only by studying two sates: original state and "others" state which contains all other mutations. Construct the transition matrix describing the two-state Markov chain. What is the probability that in the $n$th generation it will return to its original state?

**3.42.** Consider the stock portfolio problem 3.7.

(a) What will be the performance of the portfolio after 1, 2, and 10 years?
(b) Investigate the performance of the conservative portfolio that starts with the following percentage distribution of stocks $\mathbf{s}(0) = \begin{bmatrix} 0.5 & 0.5 & 0 \end{bmatrix}^t$ over the same period of time as in (a) above.
(c) Investigate the performance of a "very aggressive" portfolio that starts with the following percentage distribution of stocks $\mathbf{s}(0) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^t$ over the same period of time as in (a) above.
(d) Compare the long-term performance of the conservative and aggressive portfolios.

**3.43.** A hidden Markov chain model can be used as a waveform generator. Your task is to generate random waveforms using the following procedure.

1. Define a set of quantization levels $Q_1, Q_2, \cdots, Q_m$ for the signal values.

2. Define an $m \times m$ Markov transition matrix for the system. Choose your own transition probabilities for the matrix.
3. Choose an initial state vector from the set with only one nonzero entry chosen at random from the set of quantization levels

$$\mathbf{s}(0) = \begin{bmatrix} 0 \cdots 0 \; 1 \; 0 \cdots 0 \end{bmatrix}^t$$

If that one element is in position $k$, then the corresponding initial output value is $Q_k$.
4. Evaluate the next state vector using the iteration

$$\mathbf{s}(i) = \mathbf{P}\,\mathbf{s}(i-1)$$

5. Generate the cumulative function

$$F_j(i) = \sum_{k=1}^{j} s_k(i)$$

6. Generate a random variable $x$ using the uniform distribution and estimate the index $j$ for the output $Q_j$ that satisfies the inequality

$$F_{j-1}(i) < x \leq F_j(i)$$

7. Repeat 4.

## *Finding* $\mathbf{P}^n$ *by Expanding* $\mathbf{s}(0)$

**3.44.** In Sect. 3.11 we expressed $\mathbf{s}(0)$ in terms of the eigenvectors of the transition matrix according to (3.41). Prove that if a pair of the eigenvectors is a complex conjugate pair, then the corresponding coefficients of $\mathbf{c}$ are also a complex conjugate pair.

**3.45.** The given transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.4 & 0.6 \\ 0.1 & 0.4 & 0.2 \\ 0.8 & 0.2 & 0.2 \end{bmatrix}$$

has distinct eigenvalues. Express the initial state vector

$$\mathbf{s}(0) = \begin{bmatrix} 0 \; 1 \; 0 \end{bmatrix}^t$$

in terms of its eigenvectors then find the distribution vector for $n = 4$.

**3.46.** The given transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.4 & 0.4 & 0.1 \\ 0.1 & 0.4 & 0.2 & 0.1 \\ 0.1 & 0.1 & 0.2 & 0.7 \\ 0.7 & 0.1 & 0.2 & 0.1 \end{bmatrix}$$

has distinct, but complex, eigenvalues. Express the initial state vector

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}^t$$

in terms of its eigenvectors then find the distribution vector for $n = 5$.

**3.47.** The given transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.4 & 0.1 & 0.8 \\ 0.1 & 0.4 & 0.5 & 0 \\ 0.3 & 0.1 & 0.4 & 0.1 \\ 0.5 & 0.1 & 0.0 & 0.1 \end{bmatrix}$$

has distinct, but complex, eigenvalues. Express the initial state vector

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}^t$$

in terms of its eigenvectors then find the distribution vector for $n = 5$.

**3.48.** The given transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.3 & 0.5 \\ 0.5 & 0.3 & 0 \\ 0 & 0.4 & 0.5 \end{bmatrix}$$

has distinct, but complex, eigenvalues. Express the initial state vector

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^t$$

in terms of its eigenvectors then find the distribution vector for $n = 7$.

**3.49.** The given transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.75 \\ 0.5 & 0.25 \end{bmatrix}$$

has distinct, and real, eigenvalues. Express the initial state vector

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 1 \end{bmatrix}^t$$

in terms of its eigenvectors then find the distribution vector for $n = 7$.

**3.50.** The given transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.9 & 0.75 \\ 0.1 & 0.25 \end{bmatrix}$$

has distinct, and real, eigenvalues. Express the initial state vector

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 1 \end{bmatrix}^t$$

in terms of its eigenvectors then find the distribution vector for $n = 7$.

## *Finding* $\mathbf{P}^n$ *by Diagonalizing* $\mathbf{P}$

**3.51.** Prove Lemma 3.3 on page 95.

**3.52.** Prove Theorem 3.7 on page 98.

**3.53.** Prove Lemma 3.5 on page 97. Start with an initial distribution vector $\mathbf{s}(0)$ and show that $\mathbf{s}(\infty) = \mathbf{A}_1\mathbf{s}(0)$, then find the components of the equilibrium distribution vector.

**3.54.** Prove Lemma 3.4 on page 97.

**3.55.** Repeat problem 3.45 by decomposing $\mathbf{P}^n$ in terms of the matrices $\mathbf{A}_i$.

**3.56.** Repeat problem 3.46 by decomposing $\mathbf{P}^n$ in terms of the matrices $\mathbf{A}_i$.

**3.57.** Repeat problem 3.47 by decomposing $\mathbf{P}^n$ in terms of the matrices $\mathbf{A}_i$.

**3.58.** Diagonalize the given transition matrix and write an expression for $\mathbf{P}^n$ similar to the one in (3.80)

$$\mathbf{P} = \begin{bmatrix} 1-a & d \\ a & 1-d \end{bmatrix}$$

**3.59.** Diagonalize the given transition matrix and write an expression for $\mathbf{P}^n$ similar to the one in (3.80)

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.75 \\ 0.5 & 0.25 \end{bmatrix}$$

**3.60.** Diagonalize the given transition matrix and write an expression for $\mathbf{P}^n$ similar to the one in (3.80)

$$\mathbf{P} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Will this matrix ever settle down to a steady-state value?

**3.61.** Express the probability of requiring a block from the cache memory at any time step $n$ in Example 3.13 as a function of the eigenvalue powers.

**3.62.** Prove Lemma 3.4 on page 97 starting with (3.80) and using Theorem 3.5.

## *Jordan Canonic Form*

**3.63.** Prove the observations in Sect. 3.15.

## *Finding* $\mathbf{P}^n$ *Using Jordan Canonic Form*

**3.64.** Assume an $m \times m$ Markov matrix $\mathbf{P}$ whose Jordan canonic form equivalent matrix is $\mathbf{J}$. Prove that if $\mathbf{J}$ is purely diagonal then:

(a) $\mathbf{J}$ has $m$ Jordan blocks each of dimension $1 \times 1$.
(b) The number of linearly independent eigenvectors is $m$.
(c) The unitary matrix $\mathbf{Y}$ becomes the eigenvector matrix $\mathbf{X}$.

**3.65.** Prove for the $m \times m$ Markov matrix $\mathbf{P}$ the following statements are equivalent:

(a) Rank of $\mathbf{P}$ is $m$.
(b) Dimension of nullspace of $\mathbf{P}$ is zero.
(c) Number of Jordan blocks is $m$.
(d) Each Jordan block has dimension $1 \times 1$.
(e) Number of linearly independent eigenvectors is $m$.
(f) $\mathbf{J}$ is a diagonal matrix $\mathbf{D}$.
(g) $\mathbf{Y}$ is itself the eigenvector matrix $\mathbf{X}$.

**3.66.** Prove the statement in observation 6 on page 108 about the column vectors of the similarity matrix $\mathbf{V}$.

**3.67.** If a transition matrix has the eigenvalues 1, 0.4, and 0.3, what is the expected structure of its Jordan canonic form?

**3.68.** If a transition matrix has the eigenvalues 1, 0.4 (repeated 3 times), and 0.3, what is the expected structure of its Jordan canonic form?

**3.69.** Consider the Jordan matrix

$$\mathbf{J} = \begin{bmatrix} 0.2 & 1 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0.3 \end{bmatrix}$$

Identify the Jordan blocks and find the eigenvalues and the number of linearly independent eigenvectors.

**3.70.** Consider the Jordan matrix

$$\mathbf{J} = \begin{bmatrix} 0.2 & 1 & 0 & 0 & 0 \\ 0 & 0.2 & 1 & 0 & 0 \\ 0 & 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0.3 & 1 \\ 0 & 0 & 0 & 0 & 0.3 \end{bmatrix}$$

Identify the Jordan blocks and find the eigenvalues and the number of linearly independent eigenvectors.

**3.71.** Consider the Jordan matrix

$$\mathbf{J} = \begin{bmatrix} 0.2 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 1 & 0 & 0 \\ 0 & 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0.3 & 1 \\ 0 & 0 & 0 & 0 & 0.3 \end{bmatrix}$$

Identify the Jordan blocks and find the eigenvalues and the number of linearly independent eigenvectors.

**3.72.** Consider one Jordan block matrix of dimension $10 \times 10$

$$\mathbf{J} = \begin{bmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Use MATLAB, or any other mathematical package to see how the structure of $\mathbf{J}^i$ develops for all values of $i$ in the range 2 to 15. What are your observations? Compare your results to (3.130).

**3.73.** Obtain the Jordan canonic form for the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.25 \\ 0.1 & 0.4 & 0.55 \end{bmatrix}.$$

Compare the columns of the similarity matrix $\mathbf{V}$ with the eigenvectors of $\mathbf{P}$.

## References

1. M. Gardner, *Aha! Insight* (Scientific American/W.H.Freeman and Company, New York, 1978)
2. R. Horn, C.R. Johnson, *Matrix Analysis* (Cambridge University Press, Cambridge, 1985)
3. W.J. Stewart, *Introduction to Numerical Solutions of Markov Chains* (Princeton University Press, Princeton, 1994)
4. J. Warland, R. Varaiya, *High-Performance Communication Networks* (Moragan Kaufmann, San Francisco, 2000)
5. M.E. Woodward, *Communication and Computer Networks* (IEEE Computer Society Press, Los Alamitos, 1994)

# Chapter 4
# Markov Chains at Equilibrium

## 4.1 Introduction

In this chapter we will study the long-term behavior of Markov chains. In other words, we would like to know the distribution vector $\mathbf{s}(n)$ when $n \to \infty$. The state of the system at equilibrium or steady state can then be used to obtain performance parameters such as throughput, delay, loss probability, etc.

## 4.2 Markov Chains at Equilibrium

Assume a Markov chain in which the transition probabilities are not a function of time $t$ or $n$, for the continuous-time or discrete-time cases, respectively. This defines a *homogeneous* Markov chain. At steady state as $n \to \infty$ the distribution vector $\mathbf{s}$ settles down to a unique value and satisfies the equation

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{4.1}$$

This is because the distribution vector value does not vary from one time instant to another at steady state. We immediately recognize that $\mathbf{s}$ in that case is an eigenvector for $\mathbf{P}$ with corresponding eigenvalue $\lambda = 1$. We say that the Markov chain has reached its steady state when the above equation is satisfied.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_4

## 4.3 Significance of s at "Steady State"

Equation (4.1) indicates that if **s** is the present value of the distribution vector, then after one time step the distribution vector will be **s** still. The system is now in equilibrium or steady state. The reader should realize that we are talking about probabilities here.

At steady state the system will not settle down to one particular state, as one might suspect. Steady state means that the probability of being in any state will not change with time. The probabilities, or components, of the vector **s** are the ones that are in steady state. The components of the transition matrix $\mathbf{P}^n$ will also reach their steady state. The system is then said to be in steady state.

Assume a five-state system whose equilibrium or steady state distribution vector is

$$\mathbf{s} = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 & s_5 \end{bmatrix}^t \tag{4.2}$$

$$= \begin{bmatrix} 0.2 & 0.1 & 0.4 & 0.1 & 0.2 \end{bmatrix}^t \tag{4.3}$$

Which state would you think the system will be in at equilibrium? The answer is: The system is in state $s_1$ *with probability* 20 %. Or the system is in state $s_2$ with probability 10 %, and so on. However, we can say that at steady state the system is most probably in state $s_3$ since it has the highest probability value.

## 4.4 Finding Steady State Distribution Vector s

The main goal of this chapter is to find **s** for a given **P**. Knowledge of this vector helps us find many performance measures for our system such as packet loss probability, throughput, delay, etc. The technique we choose for finding **s** depends on the size and structure of **P**.

Since the steady state distribution is independent of the initial distribution vector $\mathbf{s}(0)$, we conclude therefore that $\mathbf{P}^n$ approaches a special structure for large values of $n$. In this case we find that the columns of $\mathbf{P}^n$, for large values of $n$, will all be identical and equal to the steady state distribution vector **s**. We could see that in Examples 3.11 on p. 84 and 3.20 on p. 102.

*Example 4.1.* Find the steady state distribution vector for the given transition matrix by:

(a) Calculating higher powers for the matrix $\mathbf{P}^n$.
(b) Calculating the eigenvectors for the matrix.

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0.4 & 0.5 \\ 0.8 & 0 & 0.5 \\ 0 & 0.6 & 0 \end{bmatrix}$$

The given matrix is column stochastic and hence could describe a Markov chain.

Repeated multiplication shows that the entries for $\mathbf{P}^n$ settle down to their stable values.

$$\mathbf{P}^2 = \begin{bmatrix} 0.36 & 0.38 & 0.30 \\ 0.16 & 0.62 & 0.40 \\ 0.48 & 0 & 0.30 \end{bmatrix}$$

$$\mathbf{P}^5 = \begin{bmatrix} 0.3648 & 0.3438 & 0.3534 \\ 0.4259 & 0.3891 & 0.3970 \\ 0.2093 & 0.2671 & 0.2496 \end{bmatrix}$$

$$\mathbf{P}^{10} = \begin{bmatrix} 0.3535 & 0.3536 & 0.3536 \\ 0.4042 & 0.4039 & 0.4041 \\ 0.2424 & 0.2426 & 0.2423 \end{bmatrix}$$

$$\mathbf{P}^{20} = \begin{bmatrix} 0.3535 & 0.3535 & 0.3535 \\ 0.4040 & 0.4040 & 0.4040 \\ 0.2424 & 0.2424 & 0.2424 \end{bmatrix}$$

The entries for $\mathbf{P}^{20}$ all reached their stable values. Since all the columns of $\mathbf{P}^{20}$ are identical, the stable distribution vector is independent of the initial distribution vector (could you prove that? It is rather simple). Furthermore, any column of $\mathbf{P}^{20}$ gives us the value of the equilibrium distribution vector.

The eigenvector corresponding to unity eigenvalue is found to be

$$\mathbf{s} = \begin{bmatrix} 0.3535 & 0.4040 & 0.2424 \end{bmatrix}^t$$

Notice that the equilibrium distribution vector is identical to the columns of the transition matrix $\mathbf{P}^{20}$. ■

## 4.5 Techniques for Finding s

We can use one of the following approaches for finding the steady state distribution vector **s**. Some approaches are algebraic while the others rely on numerical techniques.

1. Repeated multiplication of **P** to obtain $\mathbf{P}^n$ for high values of $n$.
2. Eigenvector corresponding to eigenvalue $\lambda = 1$ for **P**.
3. Difference equations.
4. Z-transform (generating functions).
5. Direct numerical techniques for solving a system of linear equations.
6. Iterative numerical techniques for solving a system of linear equations.
7. Iterative techniques for expressing the states of **P** in terms of other states.

Which technique is easier depends on the structure of **P**. Some rough guidelines follow.

The repeated multiplication technique in 1 is prone to numerical roundoff errors and one has to use repeated trials until the matrix entries stop changing for increasing values of $n$.

The eigenvector technique (2) is used when **P** is expressed numerically and its size is reasonable so that any mathematical package could easily find the eigenvector. Some communication systems are described by a small $2 \times 2$ transition matrix and it instructive to get a closed-form expression for **s**. We shall see this for the case of packet generators.

The difference equations technique (3) is used when **P** is banded with few sub-diagonals. Again, many communication systems have banded transition matrices. We shall see many examples throughout this book about such systems.

The z-transform technique (4) is used when **P** is lower triangular or lower Hessenberg such that each diagonal has identical elements. Again, some communication systems have this structure and we will discuss many of them throughout this book.

The direct technique (5) is used when **P** is expressed numerically and **P** has no particular structure. Furthermore the size of **P** is not too large such that rounding or truncation noise is insignificant. Direct techniques produce results with accuracies dependent on the machine precision and the number of calculations involved.

The iterative numerical technique (6) is used when **P** is expressed numerically and **P** has no particular structure. The size of **P** has little effect on truncation noise because iterative techniques produce results with accuracies that depend only on the machine precision and independent of the number of calculations involved.

The iterative technique (7) for expressing the states of **P** in terms of other states is illustrated in Sect. 9.3.2 on p. 322.

We illustrate these approaches in the following sections.

## 4.6 Finding s Using Eigenvector Approach

In this case we are interested in finding the eigenvector **s** which satisfies the condition

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{4.4}$$

MATLAB and other mathematical packages such as Maple and Mathematica have commands for finding that eigenvector. This technique is useful only if **P** is expressed numerically. Nowadays, those mathematical packages can also do symbolic computations and can produce an answer for **s** when **P** is expressed in symbols. However symbolic computations demand that the size of **P** must be small like 2 to 5 at the most to get any useful data.

Having found a numeric or symbolic answer, we must normalize **s** to ensure that

$$\sum_i s_i = 1 \tag{4.5}$$

*Example 4.2.* Find the steady state distribution vector for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.7 & 0.5 \\ 0.15 & 0.2 & 0.3 \\ 0.05 & 0.1 & 0.2 \end{bmatrix}$$

We use MATLAB to find the eigenvectors and eigenvalues for **P** :

$$\mathbf{s}_1 = \begin{bmatrix} 0.9726 & 0.2153 & .0877 \end{bmatrix}^t \qquad \leftrightarrow \lambda_1 = 1$$

$$\mathbf{s}_2 = \begin{bmatrix} 0.8165 & -0.4882 & -0.4082 \end{bmatrix}^t \qquad \leftrightarrow \lambda_2 = 0.2$$

$$\mathbf{s}_3 = \begin{bmatrix} 0.5345 & -0.8018 & 0.2673 \end{bmatrix}^t \qquad \leftrightarrow \lambda_3 = 0$$

The steady state distribution vector **s** corresponds to $\mathbf{s}_1$ and we have to normalize it. We have

$$\sum_i s_i = 1.2756$$

Dividing $\mathbf{s}_1$ by this value we get the steady state distribution vector as

$$\mathbf{s}_1 = \begin{bmatrix} 0.7625 & 0.1688 & 0.0687 \end{bmatrix}^t$$

■

## 4.7 Finding s Using Difference Equations

This technique for finding **s** is useful only when the state transition matrix **P** is banded. Consider the Markov chain representing a simple discrete-time birth–death process whose state transition diagram is shown in Fig. 4.1. For example, each state might correspond to the number of packets in a buffer whose size grows by one or decreases by one at each time step. The resulting state transition matrix **P** is tridiagonal with each subdiagonal composed of identical elements.

We make the following assumptions for the Markov chain.

1. The state of the Markov chain corresponds to the number of packets in the buffer or queue. $s_i$ is the probability that $i$ packets are in the buffer.
2. The size of the buffer or queue is assumed unrestricted.

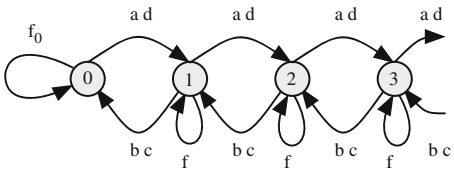



**Fig. 4.1** State transition diagram for a discrete-time birth–death Markov chain

3. The probability of a packet arriving to the system is $a$ at a particular time; and the probability that a packet does not arrive is $b = 1 - a$.
4. The probability of a packet departing the system is $c$ at a particular time; and the probability that a packet does not depart is $d = 1 - c$.
5. When a packet arrives, it could be serviced at the same time step and it could leave the queue, at that time step, with probability $c$.

From the transition diagram, we write the state transition matrix as

$$\mathbf{P} = \begin{bmatrix} f_0 & bc & 0 & 0 & \cdots \\ ad & f & bc & 0 & \cdots \\ 0 & ad & f & bc & \cdots \\ 0 & 0 & ad & f & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.6}$$

where $f_0 = ac + b$ and $f = ac + bd$. For example, starting with state 1, the system goes to state 0 when a packet does not arrive and the packet that was in the buffer departs. This is represented by the term $bc$ at location (1, 2) of the matrix.

Using this matrix, or the transition diagram, we can arrive at difference equations relating the equilibrium distribution vector components as follows. We start by writing the equilibrium equation

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{4.7}$$

Equating corresponding elements on both sides of the equation, we get the following equations.

$$ad\ s_0 - bc\ s_1 = 0 \tag{4.8}$$

$$ad\ s_0 - g\ s_1 + bc\ s_2 = 0 \tag{4.9}$$

$$ad\ s_{i-1} - g\ s_i + bc\ s_{i+1} = 0 \qquad i > 0 \tag{4.10}$$

where $g = 1 - f$ and $s_i$ is the $i$th component of the state vector $\mathbf{s}$ which is equal to the probability that the system is in state $i$. Appendix B gives techniques for solving such difference equations. However, we show here a simple method based on iterations. From Eqs. (4.8)–(4.10) we can write the expressions

$$s_1 = \left(\frac{a\ d}{b\ c}\right) s_0$$

$$s_2 = \left(\frac{a\ d}{b\ c}\right)^2 s_0$$

$$s_3 = \left(\frac{a\ d}{b\ c}\right)^3 s_0$$

and in general

$$s_i = \left(\frac{ad}{bc}\right)^i s_0 \qquad i \geq 0 \tag{4.11}$$

It is more convenient to write $s_i$ in the form

$$s_i = \rho^i\ s_0 \qquad i \geq 0 \tag{4.12}$$

where

$$\rho = \frac{a\ d}{b\ c} < 1 \tag{4.13}$$

In that sense $\rho$ can be thought of as *distribution index* that dictates the magnitude of the distribution vector components.

The complete solution is obtained from the above equations, plus the condition

$$\sum_{i=0}^{\infty} s_i = 1 \tag{4.14}$$

Substituting the expressions for each $s_i$ in the above equation, we get

$$s_0 \sum_{i=0}^{\infty} \rho^i = 1 \tag{4.15}$$

Thus we obtain

$$\frac{s_0}{1-\rho} = 1 \tag{4.16}$$

from which we obtain the probability that the system is in state 0 as

$$s_0 = 1 - \rho \tag{4.17}$$

and the components of the equilibrium distribution vector are given from (4.12) by

$$s_i = (1-\rho)\rho^i \qquad i \geq 0 \tag{4.18}$$

For the system to be stable we must have $\rho < 1$. If we are interested in situations where $\rho \geq 1$, then we must deal with finite-sized systems where the highest state could be $s_B$ and the system could exist in one of $B + 1$ states only. This situation will be treated more fully in Sect. 7.6 on p. 241.

*Example 4.3.* Consider the transition matrix for the discrete-time birth–death process that describes single arrival, single departure queue with the following parameters $a = 0.4$ and $c = 0.6$. Construct the transition matrix and find the equilibrium distribution vector.

The transition matrix becomes

$$\mathbf{P} = \begin{bmatrix} 0.84 & 0.36 & 0 & 0 & \cdots \\ 0.16 & 0.48 & 0.36 & 0 & \cdots \\ 0 & 0.16 & 0.48 & 0.36 & \cdots \\ 0 & 0 & 0.16 & 0.48 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Using (4.13), the distribution index is equal to

$$\rho = 0.4444$$

From (4.17) we have

$$s_0 = 0.5556$$

and from (4.18) we have

$$s_i = (1 - \rho)\rho^i = (0.5556) \times 0.4444^i$$

The distribution vector at steady-state is

$$\mathbf{s} = \begin{bmatrix} 0.5556 & 0.2469 & 0.1097 & 0.0488 & 0.0217 & \cdots \end{bmatrix}^t$$

■

## 4.8 Finding s Using Z-Transform

This technique is useful to find the steady state distribution vector **s** only if the state transition matrix **P** is lower Hessenberg such that elements on one diagonal are mostly identical. This last restriction will result in difference equations with constant coefficients for most of the elements of **s**. This type of transition matrix occurs naturally in multiple arrival, single departure ($M^m/M/1$) queues. This queue will be discussed in complete detail in Sect. 7.7 on p. 249.

Let us consider a Markov chain where we can move from state $j$ to any state $i$ where $i \geq j - 1$. This system corresponds to a buffer, or queue, whose size can increase by more than one due to multiple packet arrivals at any time, but the size of the queue can only decrease by one due to the presence of a single server. Assume the probability of $K$ arrivals at instant $n$ is given by

$$p(K \text{ arrivals}) = a_K \qquad K = 0, 1, 2, \ldots \tag{4.19}$$

for all time instants $n = 0, 1, 2, \cdots$. Assume that the probability that a packet is able to leave the queue is $c$ and the probability that it is not able to leave the queue is $d = 1 - c$.

The condition for the stability of the system is

$$\sum_{K=0}^{\infty} K\, a_K < c \tag{4.20}$$

which indicates that the average number of arrivals at a given time is less than the average number of departures from the system. The state transition matrix will be lower Hessenberg:

$$\mathbf{P} = \begin{bmatrix} a_0 & b_0 & 0 & 0 & \cdots \\ a_1 & b_1 & b_0 & 0 & \cdots \\ a_2 & b_2 & b_1 & b_0 & \cdots \\ a_3 & b_3 & b_2 & b_1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.21}$$

where $b_i = a_i c + a_{i-1} d$ and we assumed

$$a_i = 0 \qquad \text{when } i < 0$$

Note that the sum of each column is unity, as required from the definition of a Markov chain transition matrix.

At equilibrium we have

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{4.22}$$

The general expression for the equilibrium equations for the states is given by the $i$-th term in the above equation

$$s_i = a_i s_0 + \sum_{j=1}^{i+1} b_{i-j+1}\, s_j \tag{4.23}$$

This analysis is a modified and simpler version of the one given in [1]. Define the z-transform for the state transition probabilities $a_i$ and $b_i$ as

$$A(z) = \sum_{i=0}^{\infty} a_i \; z^{-i} \tag{4.24}$$

$$B(z) = \sum_{i=0}^{\infty} b_i \; z^{-i} \tag{4.25}$$

One observation worth mentioning here for later use is that since all entries in the transition matrix are positive, all the coefficients of $A(z)$ and $B(z)$ are positive.

We define the z-transform for the equilibrium distribution vector $\mathbf{s}$ as

$$S(z) = \sum_{i=0}^{\infty} s_i \; z^{-i} \tag{4.26}$$

where $s_i$ are the components of the distribution vector. From (4.23) we can write

$$S(z) = s_0 \sum_{i=0}^{\infty} a_i \; z^{-i} + \sum_{i=0}^{\infty} \sum_{j=1}^{i+1} b_{i-j+1} \; s_j \; z^{-i} \tag{4.27}$$

$$= s_0 A(z) + \sum_{i=0}^{\infty} \sum_{j=1}^{\infty} b_{i-j+1} \; s_j \; z^{-i} \tag{4.28}$$

we were able to change the upper limit for $j$, in the above equation, by assuming

$$b_i = 0 \qquad \text{when } i < 0 \tag{4.29}$$

Now we change the order of the summation

$$S(z) = s_0 A(z) + \sum_{j=1}^{\infty} \sum_{i=0}^{\infty} b_{i-j+1} \; s_j \; z^{-i} \tag{4.30}$$

Making use of the assumption in (4.29), we can change the limits of the summation for $i$

$$S(z) = s_0 A(z) + \left[ \sum_{j=1}^{\infty} s_j \; z^{-j} \right] \times \sum_{i=j-1}^{\infty} b_{i-j+1} \; z^{-(i-j)} \tag{4.31}$$

We make use of the definition of $S(z)$ to change the term in the square brackets

$$S(z) = s_0 A(z) + [S(z) - s_0] \times \sum_{i=j-1}^{\infty} b_{i-j+1}\, z^{-(i-j)} \tag{4.32}$$

We now change the summation symbol $i$ by using the new variable $m = i - j + 1$

$$\begin{aligned} S(z) &= s_0 A(z) + [S(z) - s_0] \times z \sum_{m=0}^{\infty} b_m\, z^{-m} \\ &= s_0 A(z) + z\,[S(z) - s_0]\, B(z) \end{aligned} \tag{4.33}$$

We finally get

$$S(z) = s_0 \times \frac{z^{-1} A(z) - B(z)}{z^{-1} - B(z)} \tag{4.34}$$

Below we show how we can obtain a numerical value for $s_0$. Assuming for the moment that $s_0$ is found, MATLAB allows us to find the inverse z-transform of $S(z)$ using the command `RESIDUE(a,b)` where $a$ and $b$ are the coefficients of $A(z)$ and $B(z)$, respectively, in descending powers of $z^{-1}$. The function `RESIDUE` returns the column vectors $r$, $p$, and $c$ which give the residues, poles, and direct terms, respectively.

The solution for $s_i$ is given by the expression

$$s_i = c_i + \sum_{j=1}^{m} r_j (p_j)^{(i-1)} \qquad i > 0 \tag{4.35}$$

where $m$ is the number of elements in $r$ or $p$ vectors. The examples below show how this procedure is done.

When $z = 1$, the z-transforms for $s$, $a_i$, and $b_i$ are

$$S(1) = \sum_{j=1}^{\infty} s_j = 1 \tag{4.36}$$

$$A(1) = B(1) = 1 \tag{4.37}$$

Thus we can put $z = 1$ in (4.34) and use L'Hospital's rule to get

$$s_0 = \frac{1 + B'(1)}{1 + B'(1) - A'(1)} \tag{4.38}$$

where

$$A'(1) = \left.\frac{dA(z)}{dz}\right|_{z=1} < 0 \tag{4.39}$$

$$B'(1) = \left.\frac{dB(z)}{dz}\right|_{z=1} < 0 \tag{4.40}$$

Since all the coefficients of $A(z)$ and $B(z)$ are positive, all the coefficients of $A'$ and $B'$ are negative and the two numbers $A'(1)$ and $B'(1)$ are smaller than zero. As a result of this observation, it is guaranteed that $s_0 < 1$ as expected for the probability that the queue is empty.

We should note that the term $-A'(1)$ represents the average number of packet arrivals per time step when the queue is empty. Similarly, $-B'(1)$ represents the average number of packet arrivals per time step when the queue is not empty.

Having found a numerical value for $s_0$, we use (4.34) to obtain the inverse z-transform of $S(z)$ and get expressions for the steady state distribution vector

$$\mathbf{s} = \begin{bmatrix} s_0 & s_1 & s_2 & \cdots \end{bmatrix} \tag{4.41}$$

*Example 4.4.* Use the z-transform technique to find the equilibrium distribution vector for the Markov chain whose transition matrix is

$$\mathbf{P} = \begin{bmatrix} 0.84 & 0.36 & 0 & 0 & \cdots \\ 0.16 & 0.48 & 0.36 & 0 & \cdots \\ 0 & 0.16 & 0.48 & 0.36 & \cdots \\ 0 & 0 & 0.16 & 0.48 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

The transition probabilities are given by

$$\begin{aligned} a_0 &= 0.84 \\ a_1 &= 0.16 \\ a_i &= 0 \qquad \text{when } i > 1 \\ b_0 &= 0.36 \\ b_1 &= 0.48 \\ b_2 &= 0.16 \\ b_i &= 0 \qquad \text{when } i > 2 \end{aligned}$$

We have the z-transforms of $A(z)$ and $B(z)$ as

$$\begin{aligned} A(z) &= 0.84 + 0.16z^{-1} \\ B(z) &= 0.36 + 0.48z^{-1} + 0.16z^{-2} \end{aligned}$$

Differentiation of the above two expressions gives

$$A(z)' = -0.16z^{-2}$$
$$B(z)' = -0.48z^{-2} - 0.32z^{-3}$$

Substituting $z^{-1} = 1$ in the above expressions, we get

$$A(1)' = -0.16$$
$$B(1)' = -0.8$$

Using (4.38) we get the probability that the queue is empty

$$s_0 = \frac{1 + B'(1)}{1 + B'(1) - A'(1)} = 0.5556$$

From (4.34) we can write

$$S(z) = \frac{0.2 - 0.2z^{-1}}{0.36 - 0.52z^{-1} + 0.16z^{-2}}$$

We can convert the polynomial expression for $S(z)$ into a partial-fraction expansion (residues) using the MATLAB command `RESIDUE`:

```
b = [0.2, -0.2]; a = [0.36, -0.52, 0.16];
[r,p,c] = residue(b,a)
r =
    0.0000
    0.2469
p =
    1.0000
    0.4444
c =
    0.5556
```

where the column vectors $r$, $p$, and $c$ give the residues, poles, and direct terms, respectively. Thus we have

$$s_0 = c = 0.5555$$

which confirms the value obtained earlier. For $i \geq 1$ we have

$$s_i = \sum_j r_j \ (p_j)^{i-1}$$

Thus the distribution vector at steady-state is given by

$$\mathbf{s} = \begin{bmatrix} 0.5556 & 0.2469 & 0.1097 & 0.0488 & 0.0217 & \cdots \end{bmatrix}^t$$

Note that this is the same distribution vector that was obtained for the same matrix using the difference equations approach in Example 4.3.

As a check, we generated the first 50 components of **s** and ensured that their sum equals unity. ■

*Example 4.5.* Use the z-transform technique to find the equilibrium distribution vector for the Markov chain whose transition matrix is

$$\mathbf{P} = \begin{bmatrix} 0.5714 & 0.4082 & 0 & 0 & 0 & \cdots \\ 0.2857 & 0.3673 & 0.4082 & 0 & 0 & \cdots \\ 0.1429 & 0.1837 & 0.3673 & 0.4082 & 0 & \cdots \\ 0 & 0.0408 & 0.1837 & 0.3673 & 0.4082 & \cdots \\ 0 & 0 & 0.0408 & 0.1837 & 0.3673 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

We have the z-transforms of $A(z)$ and $B(z)$ as

$$A(z) = 0.5714 + 0.2857z^{-1} + 0.1429z^{-2}$$
$$B(z) = 0.4082 + 0.3673z^{-1} + 0.1837z^{-2} + 0.0408z^{-3}$$

Differentiation of the above two expressions gives

$$A(z)' = -0.2857z^{-2} - 0.2857z^{-3}$$
$$B(z)' = -0.3673z^{-2} - 0.3673z^{-3} - 0.1224z^{-4}$$

Substituting $z^{-1} = 1$ in the above expressions, we get

$$A(1)' = -0.5714$$
$$B(1)' = -0.8571$$

Using (4.38) we get the probability that the queue is empty

$$s_0 = \frac{1 + B'(1)}{1 + B'(1) - A'(1)} = 0.2$$

From (4.34) we can write

$$S(z) = \frac{0.0816 - 0.0408z^{-1} - 0.0204z^{-2} - 0.0204z^{-3}}{0.4082 - 0.6326z^{-1} + 0.1837z^{-2} + 0.0408z^{-3}}$$

We can convert the polynomial expression for $S(z)$ into a partial-fraction expansion (residues) using the MATLAB command `residue`:

```
b = [0.0816, -0.0408, -0.0204, -0.0204];
a = [0.4082, -0.6327, 0.1837, 0.0408];
[r,p,c] = residue(b,a)
r =
    0.0000
    0.2574
   -0.0474
p =
    1.0000
    0.6941
   -0.0474
c =
    0.2000
```

where the column vectors $r$, $p$, and $c$ give the residues, poles, and direct terms, respectively. Thus we have

$$s_0 = c = 0.2$$

which confirms the value obtained earlier. For $i \geq 1$ we have

$$s_i = \sum_j r_j \ (p_j)^{i-1}$$

Thus the distribution vector at steady state is given by

$$\mathbf{s} = \left[0.2100\ 0.1855\ 0.1230\ 0.0860\ 0.0597 \cdots\right]^t$$

As a check, we generated the first 50 components of **s** and ensured that their sum equals unity. ■

## 4.9 Finding s Using Forward- or Back-Substitution

This technique is useful when the transition matrix **P** a is a lower Hessenberg matrix and the elements in each diagonal are not equal. In such matrices the elements $p_{ij} = 0$ for $j > i + 1$. The following example shows a lower Hessenberg matrix of order 6:

$$\mathbf{P} = \begin{bmatrix} h_{11} & h_{12} & 0 & 0 & 0 & 0 \\ h_{21} & h_{22} & h_{23} & 0 & 0 & 0 \\ h_{31} & h_{32} & h_{33} & h_{34} & 0 & 0 \\ h_{41} & h_{42} & h_{43} & h_{44} & h_{45} & 0 \\ h_{51} & h_{52} & h_{53} & h_{54} & h_{55} & h_{56} \\ h_{61} & h_{62} & h_{63} & h_{64} & h_{65} & h_{66} \end{bmatrix} \tag{4.42}$$

This matrix describes the $M^m/M/1$ queue in which a maximum of $m$ packets may arrive as will be explained in Chap. 7. At steady state the distribution vector $\mathbf{s}$ satisfies

$$\mathbf{Ps} = \mathbf{s} \tag{4.43}$$

and when $\mathbf{P}$ is lower Hessenberg we have

$$\begin{bmatrix} h_{11} & h_{12} & 0 & 0 & \cdots \\ h_{21} & h_{22} & h_{23} & 0 & \cdots \\ h_{31} & h_{32} & h_{33} & h_{34} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \end{bmatrix} \tag{4.44}$$

where we assumed our states are indexed as $s_1, s_2, \cdots$. Forward substitution starts with estimating a value for $s_1$ then proceeding to find $s_2$, $s_3$, and so on. The first row gives

$$s_1 = h_{11}\, s_1 + h_{12}\, s_2 \tag{4.45}$$

We assume an arbitrary value for $s_1 = 1$. Thus the above equation gives us a value for $s_2$

$$s_2 = (1 - h_{11})\,/\,h_{12} \tag{4.46}$$

We remind the reader again that $s_1 = 1$ by assumption. The second row gives

$$s_2 = h_{21}\, s_1 + h_{22}\, s_2 + h_{23}\, s_3 \tag{4.47}$$

Substituting the values we have so far for $s_1$ and $s_2$, we get

$$s_3 = (1 - h_{11})\,(1 - h_{22})\,/\,(h_{12}\, h_{23}) - h_{21}/\,h_{23} \tag{4.48}$$

Continuing in this fashion, we can find all the states $s_i$ where $i > 1$.

To get the true value for the distribution vector $\mathbf{s}$, we use the normalizing equation

$$\sum_{i=1}^{m} s_i = 1 \tag{4.49}$$

Let us assume that the sum of the components that we obtained for the vector **s** gives

$$\sum_{i=1}^{m} s_i = b \tag{4.50}$$

then we must divide each value of **s** by $b$ to get the true normalized vector that we desire.

Backward substitution is similar to forward substitution but starts by assuming a value for $s_m$ then we estimate $s_{m-1}$, $s_{m-2}$, and so on. Obviously, backward substitution applies only to finite matrices.

*Example 4.6.* Use forward substitution to find the equilibrium distribution vector **s** for the Markov chain with transition matrix given by

$$\mathbf{P} = \begin{bmatrix} 0.4 & 0.2 & 0 & 0 & 0 & 0 \\ 0.3 & 0.35 & 0.2 & 0 & 0 & 0 \\ 0.2 & 0.25 & 0.35 & 0.2 & 0 & 0 \\ 0.1 & 0.15 & 0.25 & 0.35 & 0.2 & 0 \\ 0 & 0.05 & 0.15 & 0.25 & 0.35 & 0.2 \\ 0 & 0 & 0.05 & 0.2 & 0.45 & 0.8 \end{bmatrix}$$

Assume $s_1 = 1$. The distribution vector must satisfy the equation

$$\mathbf{s} = \begin{bmatrix} 0.4 & 0.2 & 0 & 0 & 0 & 0 \\ 0.3 & 0.35 & 0.2 & 0 & 0 & 0 \\ 0.2 & 0.25 & 0.35 & 0.2 & 0 & 0 \\ 0.1 & 0.15 & 0.25 & 0.35 & 0.2 & 0 \\ 0 & 0.05 & 0.15 & 0.25 & 0.35 & 0.2 \\ 0 & 0 & 0.05 & 0.2 & 0.7 & 0.8 \end{bmatrix} \begin{bmatrix} 1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{bmatrix} = \begin{bmatrix} 1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{bmatrix}$$

The first row gives us a value for $s_2 = 3$. Continuing, with successive rows, we get

$$s_3 = 8.25$$
$$s_4 = 22.0625$$
$$s_5 = 58.6406$$
$$s_6 = 156.0664$$

Summing the values of all the components gives us

$$\sum_{j=1}^{6} s_j = 249.0195 \tag{4.51}$$

Thus the normalized distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0.0040 & 0.0120 & 0.0331 & 0.0886 & 0.2355 & 0.6267 \end{bmatrix}^t \tag{4.52}$$

■

## 4.10 Finding s Using Direct Techniques

Direct techniques are useful when the transition matrix **P** has no special structure but its size is small such that rounding errors[1] are below a specified maximum level. In that case we start with the equilibrium equation

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{4.53}$$

where **s** is the unknown $n$-component distribution vector. This can be written as

$$(\mathbf{P} - \mathbf{I})\,\mathbf{s} = \mathbf{0} \tag{4.54}$$

$$\mathbf{A}\,\mathbf{s} = \mathbf{0} \tag{4.55}$$

which describes a *homogeneous system of linear equations* with $\mathbf{A} = \mathbf{P} - \mathbf{I}$. The rank of **A** is $n - 1$ since the sum of the columns must be zero. Thus, there are many possible solutions to the system and we need an extra equation to get a unique solution.

The extra equation that is required is the normalizing condition

$$\sum_{i=1}^{m} s_i = 1 \tag{4.56}$$

where we assumed our states are indexed as $s_1, s_2, \cdots, s_m$. We can delete any row matrix **A** in (4.55) and replace it with (4.56). Let us replace the last row with (4.56). In that case we have the system of linear equations

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m-1,1} & a_{m-1,2} & \cdots & a_{m-1,m} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_{m-1} \\ s_m \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{4.57}$$

[1]Rounding errors occur due to finite word length in computers. In floating point arithmetic, rounding errors occur whenever addition or multiplication operations are performed. In fixed-point arithmetic, rounding errors occur whenever multiplication or shift operations are performed.

This gives us a system of linear equations whose solution is the desired steady state distribution vector.

*Example 4.7.* Find the steady state distribution vector for the state transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.4 & 0.2 & 0 \\ 0.1 & 0.5 & 0.6 \\ 0.5 & 0.3 & 0.4 \end{bmatrix}$$

First, we have to obtain matrix $\mathbf{A} = \mathbf{P} - \mathbf{I}$

$$\mathbf{A} = \begin{bmatrix} -0.6 & 0.2 & 0 \\ 0.1 & -0.5 & 0.6 \\ 0.5 & 0.3 & -0.6 \end{bmatrix}$$

Now we replace the last row in $\mathbf{A}$ with all ones to get

$$\mathbf{A} = \begin{bmatrix} -0.6 & 0.2 & 0 \\ 0.1 & -0.5 & 0.6 \\ 1 & 1 & 1 \end{bmatrix}$$

The system of linear equations we have to solve is

$$\begin{bmatrix} -0.6 & 0.2 & 0 \\ 0.1 & -0.5 & 0.6 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

The solution for $\mathbf{s}$ is

$$\mathbf{s} = \begin{bmatrix} 0.1579 & 0.4737 & 0.3684 \end{bmatrix}^t$$

■

## 4.11 Finding s Using Iterative Techniques

Iterative techniques are useful when the transition matrix $\mathbf{P}$ has no special structure and its size is large such that direct techniques will produce too much rounding errors. Iterative techniques obtain a solution to the system of linear equations without arithmetic rounding noise. The accuracy of the results is limited only by machine precision. We enumerate below three techniques for doing the iterations. These techniques are explained in more detail in Appendix D.

1. Successive overrelaxation
2. Jacobi iterations
3. Gauss–Seidel iterations

Basically the solution is obtained by first assuming a trial solution then this is improved through successive iterations. Each iteration improves the guess solution and an answer is obtained when successive iterations do not result in significant changes in the answer.

## 4.12 Balance Equations

In steady state the probability of finding ourselves in state $s_i$ is given by

$$s_i = \sum_j p_{ij}\ s_j \tag{4.58}$$

The above equation is called the *balance equation* because it provides an expression for each state of the queue at steady state.

From the definition of transition probability, we can write

$$\sum_j p_{ji} = 1 \tag{4.59}$$

which is another way of saying that the sum of all probabilities of leaving state $i$ is equal to one. From the above two equations we can write

$$s_i \sum_j p_{ji} = \sum_j p_{ij}\ s_j \tag{4.60}$$

Since $s_i$ is independent of the index of summation on the L.H.S., we can write

$$\sum_j p_{ji}\ s_i = \sum_j p_{ij}\ s_j \tag{4.61}$$

Now the L.H.S. represents all the probabilities of *flowing out* of state $i$. The R.H.S. represents all the probabilities of *flowing into* state $i$. The above equation describes the *flow balance* for state $i$.

Thus we proved that in steady state, the probability of moving out of a state equals the probability of moving into the same state. This conclusion will help us derive the steady state distributions in addition to the other techniques we have discussed above.

## 4.13 Problems

### *Finding* s *Using Eigenvectors*

**4.1.** Assume **s** is the eigenvector corresponding to unity eigenvalue for matrix **P**. Prove that this vector cannot have a zero component in it if **P** does not have any zero elements.

**4.2.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.45 & 0.2 & 0.5 \\ 0.5 & 0.2 & 0.3 \\ 0.05 & 0.6 & 0.2 \end{bmatrix}$$

**4.3.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.29 & 0.46 & 0.4 \\ 0.4 & 0.45 & 0.33 \\ 0.31 & 0.09 & 0.27 \end{bmatrix}$$

**4.4.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.33 & 0.48 & 0.41 \\ 0.3 & 0.01 & 0.48 \\ 0.37 & 0.51 & 0.11 \end{bmatrix}$$

**4.5.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.33 & 0.51 & 0.12 \\ 0.24 & 0.17 & 0.65 \\ 0.43 & 0.32 & 0.23 \end{bmatrix}$$

**4.6.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.03 & 0.19 & 0.07 \\ 0.44 & 0.17 & 0.53 \\ 0.53 & 0.64 & 0.4 \end{bmatrix}$$

**4.7.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.56 & 0.05 & 0.2 \\ 0.14 & 0.57 & 0.24 \\ 0.3 & 0.38 & 0.56 \end{bmatrix}$$

**4.8.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.08 & 0.19 & 0.07 & 0.05 \\ 0.04 & 0.17 & 0.26 & 0.03 \\ 0.18 & 0.17 & 0.27 & 0.12 \\ 0.70 & 0.47 & 0.40 & 0.8 \end{bmatrix}$$

**4.9.** Find the steady-state distribution vector corresponding to the unity eigenvalue for the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.12 & 0.06 & 0.42 & 0.1 & 0.09 \\ 0.18 & 0.14 & 0.03 & 0.14 & 0.01 \\ 0.23 & 0.33 & 0.17 & 0.14 & 0.32 \\ 0.26 & 0.32 & 0.38 & 0.43 & 0.18 \\ 0.21 & 0.15 & 0 & 0.19 & 0.4 \end{bmatrix}$$

## *Finding* s *by Difference Equations*

**4.10.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.5 & 0 & 0 & 0 \\ 0.2 & 0.3 & 0.5 & 0 & 0 \\ 0 & 0.2 & 0.3 & 0.5 & 0 \\ 0 & 0 & 0.2 & 0.3 & 0.5 \\ 0 & 0 & 0 & 0.2 & 0.5 \end{bmatrix}$$

(a) Find the steady-state distribution vector using the difference equations approach.
(b) What is the probability that the queue is full?

**4.11.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.2 & 0 & 0 & 0 \\ 0.7 & 0.1 & 0.2 & 0 & 0 \\ 0 & 0.7 & 0.1 & 0.2 & 0 \\ 0 & 0 & 0.7 & 0.1 & 0.2 \\ 0 & 0 & 0 & 0.7 & 0.8 \end{bmatrix}$$

(a) Find the steady-state distribution vector using the difference equations approach.
(b) What is the probability that the queue is full?

**4.12.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.9 & 0.1 & 0 & 0 & 0 \\ 0.1 & 0.8 & 0.1 & 0 & 0 \\ 0 & 0.1 & 0.8 & 0.1 & 0 \\ 0 & 0 & 0.1 & 0.8 & 0.1 \\ 0 & 0 & 0 & 0.1 & 0.9 \end{bmatrix}$$

(a) Find the steady-state distribution vector using the difference equations approach.
(b) What is the probability that the queue is full?

**4.13.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.75 & 0.25 & 0 & 0 & 0 \\ 0.25 & 0.5 & 0.25 & 0 & 0 \\ 0 & 0.25 & 0.5 & 0.25 & 0 \\ 0 & 0 & 0.25 & 0.5 & 0.25 \\ 0 & 0 & 0 & 0.25 & 0.75 \end{bmatrix}$$

(a) Find the steady-state distribution vector using the difference equations approach.
(b) What is the probability that the queue is full?

**4.14.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.6 & 0.2 & 0 & 0 & 0 \\ 0.2 & 0.4 & 0.2 & 0 & 0 \\ 0.2 & 0.2 & 0.4 & 0.2 & 0 \\ 0 & 0.2 & 0.2 & 0.4 & 0.2 \\ 0 & 0 & 0.2 & 0.4 & 0.8 \end{bmatrix}$$

(a) Find the steady-state distribution vector using the difference equations approach.
(b) What is the probability that the queue is full?

**4.15.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.5 & 0 & \cdots \\ 0.2 & 0.3 & 0.5 & \cdots \\ 0 & 0.2 & 0.3 & \cdots \\ 0 & 0 & 0.2 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Find the steady-state distribution vector using the difference equations approach.

**4.16.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.7 & 0 & \cdots \\ 0.2 & 0.1 & 0.7 & \cdots \\ 0 & 0.2 & 0.1 & \cdots \\ 0 & 0 & 0.2 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Find the steady-state distribution vector using the difference equations approach.

**4.17.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.9 & 0.2 & 0 & \cdots \\ 0.1 & 0.7 & 0.2 & \cdots \\ 0 & 0.1 & 0.7 & \cdots \\ 0 & 0 & 0.1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Find the steady-state distribution vector using the difference equations approach.

**4.18.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.85 & 0.35 & 0 & \cdots \\ 0.15 & 0.5 & 0.35 & \cdots \\ 0 & 0.15 & 0.5 & \cdots \\ 0 & 0 & 0.15 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Find the steady-state distribution vector using the difference equations approach.

**4.19.** A queuing system is described by the following transition matrix.

$$\mathbf{P} = \begin{bmatrix} 0.7 & 0.6 & 0 & 0 & \cdots \\ 0.2 & 0.1 & 0.6 & 0 & \cdots \\ 0.1 & 0.2 & 0.1 & 0.6 & \cdots \\ 0 & 0.1 & 0.2 & 0.1 & \cdots \\ 0 & 0 & 0.1 & 0.2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Find the steady-state distribution vector using the difference equations approach.

## *Finding* s *Using Z-Transform*

**4.20.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.3 & 0 & \cdots \\ 0.2 & 0.5 & 0.3 & \cdots \\ 0 & 0.2 & 0.5 & \cdots \\ 0 & 0 & 0.2 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.21.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.95 & 0.45 & 0 & \cdots \\ 0.05 & 0.5 & 0.45 & \cdots \\ 0 & 0.05 & 0.5 & \cdots \\ 0 & 0 & 0.05 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.22.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.95 & 0.45 & 0 & \cdots \\ 0.05 & 0.5 & 0.45 & \cdots \\ 0 & 0.05 & 0.5 & \cdots \\ 0 & 0 & 0.05 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.23.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.86 & 0.24 & 0 & \cdots \\ 0.14 & 0.62 & 0.24 & \cdots \\ 0 & 0.14 & 0.62 & \cdots \\ 0 & 0 & 0.14 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.24.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.93 & 0.27 & 0 & \cdots \\ 0.07 & 0.66 & 0.27 & \cdots \\ 0 & 0.07 & 0.66 & \cdots \\ 0 & 0 & 0.07 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.25.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.512 & 0.3584 & 0 & 0 & \cdots \\ 0.384 & 0.4224 & 0.3584 & 0 & \cdots \\ 0.096 & 0.1824 & 0.4224 & 0.3584 & \cdots \\ 0.008 & 0.0344 & 0.1824 & 0.4224 & \cdots \\ 0 & 0.0024 & 0.0344 & 0.1824 & \cdots \\ 0 & 0 & 0.0024 & 0.0344 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.26.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.720 & 0.4374 & 0 & 0 & \cdots \\ 0.243 & 0.4374 & 0.4374 & 0 & \cdots \\ 0.027 & 0.1134 & 0.4374 & 0.4374 & \cdots \\ 0.001 & 0.0114 & 0.1134 & 0.4374 & \cdots \\ 0 & 0.0004 & 0.0114 & 0.1134 & \cdots \\ 0 & 0 & 0.0004 & 0.0114 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.27.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.9127 & 0.3651 & 0 & 0 & \cdots \\ 0.0847 & 0.5815 & 0.3651 & 0 & \cdots \\ 0.0026 & 0.0519 & 0.5815 & 0.3651 & \cdots \\ 0 & 0.0016 & 0.0519 & 0.5815 & \cdots \\ 0 & 0 & 0.0016 & 0.0519 & \cdots \\ 0 & 0 & 0 & 0.0016 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.28.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.6561 & 0.5249 & 0 & 0 & \cdots \\ 0.2916 & 0.3645 & 0.5249 & 0 & \cdots \\ 0.0486 & 0.0972 & 0.3645 & 0.5249 & \cdots \\ 0.0036 & 0.0126 & 0.0972 & 0.3645 & \cdots \\ 0.0001 & 0.0008 & 0.0126 & 0.0972 & \cdots \\ 0 & 0 & 0.0008 & 0.0126 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**4.29.** Given the following state transition matrix find the first ten components of the equilibrium distribution vector using the z-transform approach.

$$\mathbf{P} = \begin{bmatrix} 0.512 & 0.4096 & 0 & 0 & \cdots \\ 0.384 & 0.4096 & 0.4096 & 0 & \cdots \\ 0.096 & 0.1536 & 0.4096 & 0.4096 & \cdots \\ 0.008 & 0.0256 & 0.1536 & 0.4096 & \cdots \\ 0.0001 & 0.0016 & 0.0256 & 0.1536 & \cdots \\ 0 & 0 & 0.0016 & 0.0256 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

## Reference

1. M.E. Woodward, *Communication and Computer Networks* (IEEE Computer Society Press, Los Alamitos, 1994)

# Chapter 5
# Reducible Markov Chains

## 5.1 Introduction

Reducible Markov chains describe systems that have particular states such that once we visit one of those states, we cannot visit other states. An example of systems that can be modeled by reducible Markov chains is games of chance where once the gambler is broke, the game stops and the casino either kicks him out or gives him some compensation (comp). The gambler moved from being in a state of play to being in a comp state and the game stops there. Another example of reducible Markov chains is studying the location of a fish swimming in the ocean. The fish is free to swim at any location as dictated by the currents, food, or presence of predators. Once the fish is caught in a net, it cannot escape and it has limited space where it can swim.

Consider the transition diagram in Fig. 5.1a. Starting at any state, we are able to reach any other state in the diagram directly, in one step, or indirectly, through one or more intermediate states. Such a Markov chain is termed *irreducible Markov chain* for reasons that will be explained shortly. For example, starting at $s_1$, we can directly reach $s_2$ and we can indirectly reach $s_3$ through either of the intermediate states $s_2$ or $s_5$. We encounter irreducible Markov chains in systems that can operate for long periods of time such as the state of the lineup at a bank, during business hours. The number of customers lined up changes all the time between zero to maximum. Another example is the state of buffer occupancy in a router or a switch. The buffer occupancy changes between being completely empty to being completely full depending on the arriving traffic pattern.

Consider now the transition diagram in Fig. 5.1b. Starting from any state, we might not be able to reach other states in the diagram, directly or indirectly. Such a Markov chain is termed *reducible Markov chain* for reasons that will be explained



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_5

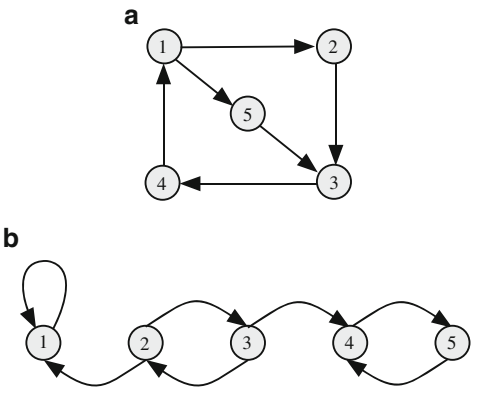



**Fig. 5.1** State transition diagrams. (**a**) Irreducible Markov chain. (**b**) Reducible Markov chain

shortly. For example, if we start at $s_1$, we can never reach any other state. If we start at state $s_4$, we can only reach state $s_5$. If we start at state $s_3$, we can reach all other states. We encounter reducible Markov chains in systems that have terminal conditions such as most games of chance like gambling. In that case the player keeps on playing till she loses all her money or wins. In either cases, she leaves the game. Another example is the game of snakes and ladders where the player keeps on playing but cannot go back to the starting position. Ultimately the player reaches the final square and could not go back again to the game.

## 5.2 Definition of Reducible Markov Chain

The traditional way to define a reducible Markov chain is as follows.

**A Markov chain is irreducible if there is some integer $k > 1$ such that all the elements of $\mathbf{P}^k$ are nonzero.**

What is the value of $k$? No one seems to know, the only advice is to keep on multiplying till the conditions are satisfied or computation noise overwhelms us!

This chapter is dedicated to shed more light on this situation and introduce, for the first time, a simple and rigorous technique for identifying a reducible Markov chain. As a bonus, the states of the Markov chain will be identified as closed or transient without much effort on our part.

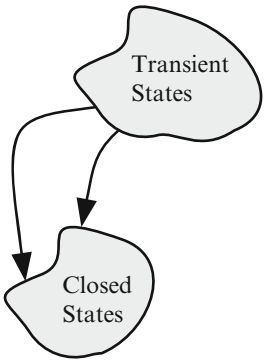



**Fig. 5.2** The states of a reducible Markov chain are divided into two sets: transient states and closed states

## 5.3 Closed and Transient States

We defined an irreducible (or regular) Markov chain as one in which every state is reachable from every other state either directly or indirectly. We also defined a reducible Markov chain as one in which some states cannot reach other states.

Thus the states of a reducible Markov chain are divided into two sets: closed states ($C$) and transient states ($T$). Figure 5.2 shows the two sets of states and the directions of transitions between the two sets of states.

When the system is in $T$, it can make a transition to either $T$ or $C$. However once our system is in $C$, it can never make a transition to $T$ again no matter how long we iterate. In other words, the probability of making a transition from a closed state to a transient state is exactly zero.

When $C$ consists of only one state, then that state is called an *absorbing* state. When $s_i$ is an absorbing state, we would have $p_{ii} = 1$. Thus inspection of the transition matrix quickly informs us of the presence of any absorbing states since the diagonal element for that state will be 1.

## 5.4 Transition Matrix of Reducible Markov Chains

Through proper state assignment, the transition matrix **P** for a reducible Markov chain could be partitioned into the *canonic form*

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.1}$$

where

$$\mathbf{C} = \text{square column stochastic matrix}$$

$$\mathbf{A} = \text{rectangular nonnegative matrix}$$

$$\mathbf{T} = \text{square column substochastic matrix}$$

Appendix D defines the meaning of nonnegative, and substochastic matrices.

The matrix **C** is a column stochastic matrix that can be studied separately from the rest of the transition matrix **P**. In fact, the eigenvalues and eigenvectors of **C** will be used to define the behavior of the Markov chain at equilibrium.

The states of the Markov chain are now partitioned into two mutually exclusive subsets as shown below.

$$C = \text{set of closed states belonging to matrix } \mathbf{C}$$

$$T = \text{set of transient states belonging to matrix } \mathbf{T}$$

The following equation explicitly shows the partitioning of the states into two sets of closed states $C$ and transient states $T$

$$\mathbf{P} = \begin{array}{c} \\ C \\ T \end{array} \begin{array}{c} \begin{array}{cc} C & T \end{array} \\ \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \end{array} \tag{5.2}$$

*Example 5.1.* The given transition matrix represents a reducible Markov chain.

$$\mathbf{P} = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{array} \begin{array}{c} \begin{array}{cccc} s_1 & s_2 & s_3 & s_4 \end{array} \\ \begin{bmatrix} 0.8 & 0 & 0.1 & 0.1 \\ 0 & 0.5 & 0 & 0.2 \\ 0.2 & 0.2 & 0.9 & 0 \\ 0 & 0.3 & 0 & 0.7 \end{bmatrix} \end{array}$$

where the states are indicated around **P** for illustration. Rearrange the rows and columns to express the matrix in the canonic form in (5.1) or (5.2) and identify the matrices **C**, **A**, and **T**. Verify the assertions that **C** is column stochastic, **A** is nonnegative, and **T** is column substochastic.

After exploring a few possible transitions starting from any initial state, we see that if we arrange the states in the order 1, 3, 2, 4, then the following state matrix is obtained

$$\mathbf{P} = \begin{array}{c} \\ s_1 \\ s_3 \\ s_2 \\ s_4 \end{array} \begin{array}{c} \begin{array}{cccc} s_1 & s_3 & s_2 & s_4 \end{array} \\ \begin{bmatrix} 0.8 & 0.1 & 0 & 0.1 \\ 0.2 & 0.9 & 0.2 & 0 \\ 0 & 0 & 0.5 & 0.2 \\ 0 & 0 & 0.3 & 0.7 \end{bmatrix} \end{array}$$

We see that the matrix exhibits the reducible Markov chain structure and matrices **C**, **A**, and **T** are

$$\mathbf{C} = \begin{bmatrix} 0.8 & 0.1 \\ 0.2 & 0.9 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 0.1 \\ 0.2 & 0 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.5 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}$$

The sum of each column of **C** is exactly 1, which indicates that it is column stochastic. The sum of columns of **T** is less than 1, which indicates that it is column substochastic.

The set of closed states is $C = \{1,\ 3\}$ and the set of transient states is $T = \{2,\ 4\}$.

Starting in state $s_2$ or $s_4$ will ultimately take us to states $s_1$ and $s_3$. Once we are there, we cannot ever go back to state $s_2$ or $s_4$ because we entered the closed states. ■

*Example 5.2.* Consider the reducible Markov chain of the previous example. Assume that the system was initially in state $s_3$. Find the distribution vector at 20 time step intervals.

We do not have to rearrange the transition matrix to do this example. We have

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0 & 0.1 & 0.1 \\ 0 & 0.5 & 0 & 0.2 \\ 0.2 & 0.2 & 0.9 & 0 \\ 0 & 0.3 & 0 & 0.7 \end{bmatrix}$$

The initial distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}^t$$

The distribution vector at 20 time step intervals is

$$\mathbf{s}(20) = \begin{bmatrix} 0.3208 & 0.0206 & 0.6211 & 0.0375 \end{bmatrix}^t$$

$$\mathbf{s}(40) = \begin{bmatrix} 0.3327 & 0.0011 & 0.6642 & 0.0020 \end{bmatrix}^t$$

$$\mathbf{s}(60) = \begin{bmatrix} 0.3333 & 0.0001 & 0.6665 & 0.0001 \end{bmatrix}^t$$

$$\mathbf{s}(80) = \begin{bmatrix} 0.3333 & 0.0000 & 0.6667 & 0.0000 \end{bmatrix}^t$$

We note that after 80 time steps, the probability of being in the transient states $s_2$ or $s_4$ is nil. The system will definitely be in the closed set composed of states $s_1$ and $s_3$. ■

## 5.5 Composite Reducible Markov Chains

In the general case, the reducible Markov chain could be composed of two or more sets of closed states. Figure 5.3 shows a reducible Markov chain with two sets of closed states. If the system is in the transient states $T$, it can move to either sets of closed states $C_1$ or $C_2$. However, if the system is in state $C_1$, it cannot move to $T$ or $C_2$. Similarly, if the system is in state $C_2$, it cannot move to $T$ or $C_1$.

In that case, the canonic form for the transition matrix $\mathbf{P}$ for a reducible Markov chain could be expanded into several subsets of non-communicating closed states

$$\mathbf{P} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{0} & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{C}_2 & \mathbf{A}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.3}$$

where

$$\begin{aligned} \mathbf{C}_1 \text{ and } \mathbf{C}_2 &= \text{square column stochastic matrices} \\ \mathbf{A}_1 \text{ and } \mathbf{A}_2 &= \text{rectangular nonnegative matrices} \\ \mathbf{T} &= \text{square column substochastic matrix} \end{aligned}$$

Since the transition matrix contains two column stochastic matrices $\mathbf{C}_1$ and $\mathbf{C}_2$, we expect to get two eigenvalues $\lambda_1 = 1$ and $\lambda_2 = 1$ also. And we will be getting

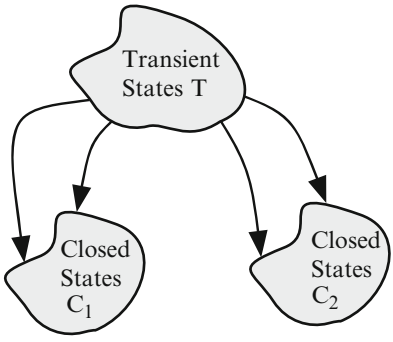



**Fig. 5.3** A reducible Markov chain with two sets of closed states

two possible steady-state distributions based on the initial value of the distribution vector $\mathbf{s}(0)$. More on that in Sects. 5.7 and 5.9.

The states of the Markov chain are now divided into three mutually exclusive sets as shown below.

$$\begin{aligned} C_1 &= \text{set of closed states belonging to matrix } \mathbf{C}_1 \\ C_2 &= \text{set of closed states belonging to matrix } \mathbf{C}_2 \\ T &= \text{set of transient states belonging to matrix } \mathbf{T} \end{aligned}$$

The following equation explicitly shows the partitioning of the states

$$\mathbf{P} = \begin{array}{c} \\ C_1 \\ C_2 \\ T \end{array} \begin{array}{c} \begin{array}{ccc} C_1 & C_2 & T \end{array} \\ \begin{bmatrix} \mathbf{C}_1 & \mathbf{0} & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{C}_2 & \mathbf{A}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{T} \end{bmatrix} \end{array} \tag{5.4}$$

Notice from the structure of $\mathbf{P}$ in (5.4) that if we were in the first set of closed states $C_1$, then we cannot escape that set to visit $C_2$ or $T$.

Similarly if we were in the second set of closed states $C_2$, then we cannot escape that set to visit $C_1$ or $T$.

On the other hand, if we were in the set of transient states $T$, then we cannot stay in that set since we will ultimately fall into $C_1$ or $C_2$.

*Example 5.3.* You play a coin tossing game with a friend. The probability that one player winning \$1 is p, and the probability that he loses \$1 is $q = 1 - p$. Assume the combined assets of both players is \$6 and the game ends when one of the players is broke. Define a Markov chain whose state $s_i$ means that you have $\$i$ and construct the transition matrix. If the Markov chain is reducible, identify the closed and transient states and rearrange the matrix to conform to the structure of (5.3) or (5.4).

Since this is a gambling game, we suspect that we have a reducible Markov chain with closed states where one player is the winner and the other is the loser.

A player could have \$0, \$1, $\cdots$, or \$6. Therefore, the transition matrix is of dimension $7 \times 7$ as shown

$$\mathbf{P} = \begin{bmatrix} 1 & q & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q & 0 & 0 & 0 & 0 \\ 0 & p & 0 & q & 0 & 0 & 0 \\ 0 & 0 & p & 0 & q & 0 & 0 \\ 0 & 0 & 0 & p & 0 & q & 0 \\ 0 & 0 & 0 & 0 & p & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & p & 1 \end{bmatrix} \tag{5.5}$$

Notice that states 0 and 6 are absorbing states since $p_{00} = p_{66} = 1$. The set $T = \{s_1, s_2, \cdots, c_5\}$ is the set of transient states. We could rearrange our transition matrix such that states $s_0$ and $s_6$ are adjacent as shown

$$\mathbf{P} = \begin{array}{c} \\ s_0 \\ s_6 \\ \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{array} \begin{array}{c} \begin{array}{cc} s_0 & s_6 \end{array} \\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \end{array} \begin{array}{c} \begin{array}{ccccc} s_1 & s_2 & s_3 & s_4 & s_5 \end{array} \\ \begin{bmatrix} q & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & p \\ \\ 0 & q & 0 & 0 & 0 \\ p & 0 & q & 0 & 0 \\ 0 & p & 0 & q & 0 \\ 0 & 0 & p & 0 & q \\ 0 & 0 & 0 & p & 0 \end{bmatrix} \end{array}$$

We have added spaces between the elements of the matrix to show the outline of the component matrices $\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{T}$. In that case each closed matrix corresponds to a single absorbing state ($s_0$ and $s_6$), while the transient states correspond to a $5 \times 5$ matrix. ■

## 5.6 Transient Analysis

We might want to know how a reducible Markov chain varies with time $n$ since this leads to useful results such as the probability of visiting a certain sate at any given time value. In other words, we want to find $\mathbf{s}(n)$ from the expression

$$\mathbf{s}(n) = \mathbf{P}^n \, \mathbf{s}(0) \tag{5.6}$$

Without loss of generality we assume the reducible transition matrix to be given in the form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.7}$$

After $n$ time steps the transition matrix of a reducible Markov chain will still be reducible and will have the form

$$\mathbf{P}^n = \begin{bmatrix} \mathbf{C}^n & \mathbf{Y}^n \\ \mathbf{0} & \mathbf{T}^n \end{bmatrix} \tag{5.8}$$

where matrix $\mathbf{Y}^n$ is given by

$$\mathbf{Y}^n = \sum_{i=0}^{n-1} \mathbf{C}^{n-i-1}\,\mathbf{A}\,\mathbf{T}^i \tag{5.9}$$

We can always find $\mathbf{C}^n$ and $\mathbf{T}^n$ using the techniques discussed in Chap. 3 such as diagonalization, finding the Jordan canonic form, or even repeated multiplications.

The stochastic matrix $\mathbf{C}^n$ can be expressed in terms of its eigenvalues using (3.80) on p. 96.

$$\mathbf{C}^n = \mathbf{C}_1 + \lambda_2^n \mathbf{C}_2 + \lambda_3^n \mathbf{C}_3 + \cdots \tag{5.10}$$

where it was assumed that $\mathbf{C}_1$ is the expansion matrix corresponding to the eigenvalue $\lambda_1 = 1$ and $\mathbf{C}$ is assumed to be of dimension $m_c \times m_c$.

Similarly, the substochastic matrix $\mathbf{T}^n$ can be expressed in terms of its eigenvalues using (3.80) on p. 96.

$$\mathbf{T}^n = \lambda_1^n \mathbf{T}_1 + \lambda_2^n \mathbf{T}_2 + \lambda_3^n \mathbf{T}_3 + \cdots \tag{5.11}$$

We should note here that all the magnitudes of the eigenvalues in the above equation are less than unity.

Equation (5.9) can then be expressed in the form

$$\mathbf{Y}^n = \sum_{j=1}^{m} \mathbf{C}_j\ \mathbf{A}\ \sum_{i=0}^{n-1} \lambda_j^{n-i-1}\,\mathbf{T}^i \tag{5.12}$$

After some algebraic manipulations, we arrive at the form

$$\mathbf{Y}^n = \sum_{j=1}^{m} \lambda_j^{n-1} \mathbf{C}_j\ \mathbf{A}\ \left[\mathbf{I} - \left(\frac{\mathbf{T}}{\lambda_j}\right)^n\right] \left(\mathbf{I} - \frac{\mathbf{T}}{\lambda_j}\right)^{-1} \tag{5.13}$$

This can be written in the form

$$\begin{aligned} \mathbf{Y}^n = {} & \mathbf{C}_1\mathbf{A}\,(\mathbf{I}-\mathbf{T})^{-1}\,[\mathbf{I}-\mathbf{T}^n] + \\ & \lambda_2^{n-1}\mathbf{C}_2\mathbf{A}\left(\mathbf{I}-\frac{1}{\lambda_2}\mathbf{T}\right)^{-1}\left[\mathbf{I}-\frac{1}{\lambda_2^n}\mathbf{T}^n\right] + \\ & \lambda_3^{n-1}\mathbf{C}_3\mathbf{A}\left(\mathbf{I}-\frac{1}{\lambda_3}\mathbf{T}\right)^{-1}\left[\mathbf{I}-\frac{1}{\lambda_3^n}\mathbf{T}^n\right] + \cdots \end{aligned} \tag{5.14}$$

If some of the eigenvalues of $\mathbf{C}$ are repeated, then the above formula has to be modified as explained in Sect. 3.14 on p. 105. Problems 5.25 discusses this situation.

*Example 5.4.* A reducible Markov chain has the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.3 & 0.1 & 0.3 & 0.1 \\ 0.5 & 0.7 & 0.2 & 0.1 & 0.3 \\ 0 & 0 & 0.2 & 0.2 & 0.1 \\ 0 & 0 & 0.1 & 0.3 & 0.1 \\ 0 & 0 & 0.4 & 0.1 & 0.4 \end{bmatrix}$$

Find the value of $\mathbf{P}^{20}$ and from that find the probability of making the following transitions:

(a) From $s_3$ when $n = 0$ to $s_2$ when $n = 20$.
(b) From $s_2$ when $n = 0$ to $s_2$ when $n = 20$.
(c) From $s_4$ when $n = 0$ to $s_1$ when $n = 20$.
(d) From $s_3$ when $n = 0$ to $s_4$ when $n = 20$.

The components of the transition matrix are

$$\mathbf{C} = \begin{bmatrix} 0.5 & 0.3 \\ 0.5 & 0.7 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0.1 & 0.3 & 0.1 \\ 0.2 & 0.1 & 0.3 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.2 & 0.2 & 0.1 \\ 0.1 & 0.3 & 0.1 \\ 0.4 & 0.1 & 0.4 \end{bmatrix}$$

We use the MATLAB function that we developed `EIGPOWERS` that we developed to expand matrix $\mathbf{C}$ in terms of its eigenpowers and we have $\lambda_1 = 1$ and $\lambda_2 = 0.2$. The corresponding matrices according to (3.80) are

$$\mathbf{C}_1 = \begin{bmatrix} 0.375 & 0.375 \\ 0.625 & 0.625 \end{bmatrix}$$

$$\mathbf{C}_2 = \begin{bmatrix} 0.625 & -0.375 \\ -0.625 & 0.375 \end{bmatrix}$$

We could now use (5.13) to find $\mathbf{P}^{20}$ but instead we use repeated multiplication here

$$\mathbf{P}^{20} = \begin{bmatrix} 0.375 & 0.375 & 0.375 & 0.375 & 0.375 \\ 0.625 & 0.625 & 0.625 & 0.625 & 0.625 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(a) $p_{32} = 0.625$
(b) $p_{22} = 0.625$
(c) $p_{14} = 0.375$
(d) $p_{43} = 0$

■

*Example 5.5.* Find an expression for the transition matrix at times $n = 4$ and $n = 20$ for the reducible Markov chain characterized by the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.9 & 0.3 & 0.3 & 0.3 & 0.2 \\ 0.1 & 0.7 & 0.2 & 0.1 & 0.3 \\ 0 & 0 & 0.2 & 0.2 & 0.1 \\ 0 & 0 & 0.1 & 0.3 & 0.1 \\ 0 & 0 & 0.2 & 0.1 & 0.2 \end{bmatrix}$$

The components of the transition matrix are

$$\mathbf{C} = \begin{bmatrix} 0.9 & 0.3 \\ 0.1 & 0.7 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0.3 & 0.1 & 0.3 \\ 0.2 & 0.1 & 0.3 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.2 & 0.2 & 0.1 \\ 0.3 & 0.3 & 0.1 \\ 0.2 & 0.1 & 0.4 \end{bmatrix}$$

$\mathbf{C}^n$ is expressed in terms of its eigenvalues as

$$\mathbf{C}^n = \lambda_1^n \mathbf{C}_1 + \lambda_2^n \mathbf{C}_2$$

where $\lambda_1 = 1$ and $\lambda_2 = 0.6$ and

$$\mathbf{C}_1 = \begin{bmatrix} 0.75 & 0.75 \\ 0.25 & 0.25 \end{bmatrix}$$

$$\mathbf{C}_2 = \begin{bmatrix} 0.25 & -0.75 \\ -0.25 & 0.75 \end{bmatrix}$$

At any time instant $n$ the matrix $\mathbf{Y}^n$ has the value

$$\mathbf{Y}^n = \mathbf{C}_1 \mathbf{A} \left(\mathbf{I} - \mathbf{T}\right)^{-1} \left[\mathbf{I} - \mathbf{T}^n\right] + (0.6)^{n-1} \mathbf{C}_2 \mathbf{A} \left(\mathbf{I} - \frac{1}{0.6}\mathbf{T}\right)^{-1} \left[\mathbf{I} - \frac{1}{0.6^n}\mathbf{T}^n\right]$$

Substituting $n = 4$, we get

$$\mathbf{P}^4 = \begin{bmatrix} 0.7824 & 0.6528 & 0.6292 & 0.5564 & 0.6318 \\ 0.2176 & 0.3472 & 0.2947 & 0.3055 & 0.3061 \\ 0 & 0 & 0.0221 & 0.0400 & 0.0180 \\ 0 & 0 & 0.0220 & 0.0401 & 0.0180 \\ 0 & 0 & 0.0320 & 0.0580 & 0.0261 \end{bmatrix}$$

Substituting $n = 20$, we get

$$\mathbf{P}^{20} = \begin{bmatrix} 0.7500 & 0.7500 & 0.7500 & 0.7500 & 0.7500 \\ 0.2500 & 0.2500 & 0.2500 & 0.2500 & 0.2500 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We see that all the columns of $\mathbf{P}^{20}$ are identical which indicates that the steady state distribution vector is independent of its initial value. ■

## 5.7 Reducible Markov Chains at Steady-State

Assume we have a reducible Markov chain with transition matrix $\mathbf{P}$ that is expressed in the canonic form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.15}$$

According to (5.8), after $n$ time steps the transition matrix will have the form

$$\mathbf{P}^n = \begin{bmatrix} \mathbf{C}^n & \mathbf{Y}^n \\ \mathbf{0} & \mathbf{T}^n \end{bmatrix} \tag{5.16}$$

where matrix $\mathbf{Y}^n$ is given by

$$\mathbf{Y}^n = \sum_{i=0}^{n-1} \mathbf{C}^{n-i-1}\,\mathbf{A}\,\mathbf{T}^i \tag{5.17}$$

To be able to see how $\mathbf{P}^n$ will be like when $n \to \infty$, we express the matrices $\mathbf{C}$ and $\mathbf{T}$ in terms of their eigenvalues as in (5.10) and (5.11).

When $n \to \infty$, matrix $\mathbf{Y}^{\infty}$ becomes

$$\mathbf{Y}^{\infty} = \mathbf{C}^{\infty}\, \mathbf{A} \sum_{i=0}^{\infty} \mathbf{T}^{i} \tag{5.18}$$

$$= \mathbf{C}_1\, \mathbf{A}\, (\mathbf{I} - \mathbf{T})^{-1} \tag{5.19}$$

where $\mathbf{I}$ is the unit matrix whose dimensions match that of $\mathbf{T}$.

We used the following matrix identity to derive the above equation

$$(\mathbf{I} - \mathbf{T})^{-1} = \sum_{i=0}^{\infty} \mathbf{T}^{i} \tag{5.20}$$

Finally we can write the steady-state expression for the transition matrix of a reducible Markov chain as

$$\mathbf{P}^{\infty} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{Y}^{\infty} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.21}$$

$$= \begin{bmatrix} \mathbf{C}_1 & \mathbf{C}_1\mathbf{A}\,(\mathbf{I} - \mathbf{T})^{-1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.22}$$

The above matrix is column stochastic since it represents a transition matrix.

We can prove that the columns of the matrix $\mathbf{C}_1\mathbf{A}\,(\mathbf{I} - \mathbf{T})^{-1}$ are all identical and equal to the columns of $\mathbf{C}_1$. This is left as an exercise (see Problem 5.16).

Since all the columns of $\mathbf{P}$ at steady-state are equal, all we have to do to find $\mathbf{P}^{\infty}$ is to find one column only of $\mathbf{C}_1$. The following examples show this.

*Example 5.6.* Find the steady-state transition matrix for the reducible Markov chain characterized by the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.4 & 0 & 0.3 & 0.1 \\ 0.2 & 0.6 & 0.2 & 0.2 & 0.3 \\ 0 & 0 & 0.2 & 0.2 & 0.1 \\ 0 & 0 & 0 & 0.3 & 0.1 \\ 0 & 0 & 0.6 & 0 & 0.4 \end{bmatrix}$$

The components of the transition matrix are

$$\mathbf{C} = \begin{bmatrix} 0.8 & 0.4 \\ 0.2 & 0.6 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 0.3 & 0.1 \\ 0.2 & 0.2 & 0.3 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.2 & 0.2 & 0.1 \\ 0 & 0.3 & 0.1 \\ 0.6 & 0 & 0.4 \end{bmatrix}$$

The steady-state value of $\mathbf{C}$ is

$$\mathbf{C}^{\infty} = \mathbf{C}_1 = \begin{bmatrix} 0.6667 & 0.6667 \\ 0.3333 & 0.3333 \end{bmatrix}$$

The matrix $\mathbf{Y}^{\infty}$ has the value

$$\mathbf{Y}^{\infty} = \mathbf{C}_1 \ \mathbf{A} \ (\mathbf{I} - \mathbf{T})^{-1} = \begin{bmatrix} 0.6667 & 0.6667 \\ 0.3333 & 0.3333 \end{bmatrix}$$

Thus the steady state value of $\mathbf{P}$ is

$$\mathbf{P}^{\infty} = \begin{bmatrix} 0.6667 & 0.6667 & 0.6667 & 0.6667 & 0.6667 \\ 0.3333 & 0.3333 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The first thing we notice about the steady state value of the transition matrix is that all columns are identical. This is exactly the same property for the transition matrix of an irreducible Markov chain.

The second observation we can make about the transition matrix at steady-state is that there is no possibility of moving to a transient state irrespective of the value of the initial distribution vector.

The third observation we can make is that no matter what the initial distribution vector was, we will always wind up in the same steady-state distribution. ■

*Example 5.7.* Find the steady-state transition matrix for the reducible Markov chain characterized by the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.3 & 0.1 & 0.3 & 0.1 \\ 0.5 & 0.7 & 0.2 & 0.1 & 0.3 \\ 0 & 0 & 1 & 0.2 & 0.1 \\ 0 & 0 & 0 & 0.3 & 0.1 \\ 0 & 0 & 0 & 0.1 & 0.4 \end{bmatrix}$$

The components of the transition matrix are

$$\mathbf{C} = \begin{bmatrix} 0.5 & 0.3 \\ 0.5 & 0.7 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0.1 & 0.3 & 0.1 \\ 0.2 & 0.1 & 0.3 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.2 & 0.2 & 0.1 \\ 0.1 & 0.3 & 0.1 \\ 0.4 & 0.1 & 0.4 \end{bmatrix}$$

The steady-state value of $\mathbf{C}$ is

$$\mathbf{C}^{\infty} = \mathbf{C}_1 = \begin{bmatrix} 0.375 & 0.375 \\ 0.625 & 0.625 \end{bmatrix}$$

The matrix $\mathbf{Y}^{\infty}$ has the value

$$\mathbf{Y}^{\infty} = \mathbf{C}_1 \, \mathbf{A} \, (\mathbf{I} - \mathbf{T})^{-1} = \begin{bmatrix} 0.375 & 0.375 & 0.375 \\ 0.625 & 0.625 & 0.625 \end{bmatrix}$$

Thus the steady-state value of $\mathbf{P}$ is

$$\mathbf{P}^{\infty} = \begin{bmatrix} 0.375 & 0.375 & 0.375 & 0.375 & 0.375 \\ 0.625 & 0.625 & 0.625 & 0.625 & 0.625 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

■

## 5.8 Reducible Composite Markov Chains at Steady-State

In this section we will study the steady state behavior of reducible composite Markov chains. In the general case, the reducible Markov chain could be composed of two or more closed states. Figure 5.3 shows a reducible Markov chain with two sets of closed states. If the system is in the transient states $T$, it can move to either sets of closed states $C_1$ or $C_2$. However, if the system is in state $C_1$, it cannot move to $T$ or $C_2$. Similarly, if the system is in state $C_2$, it cannot move to $T$ or $C_1$.

Assume the transition matrix is given by the canonic form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{0} & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{C}_2 & \mathbf{A}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.23}$$

where

$$\begin{aligned} \mathbf{C}_1 \text{ and } \mathbf{C}_2 &= \text{square column stochastic matrices} \\ \mathbf{A}_1 \text{ and } \mathbf{A}_2 &= \text{rectangular nonnegative matrices} \\ \mathbf{T} &= \text{square column substochastic matrix} \end{aligned}$$

It is easy to verify that the steady-state transition matrix for such a system will be

$$\mathbf{P}^{\infty} = \begin{bmatrix} \mathbf{C}_1' & \mathbf{0} & \mathbf{Y}_1 \\ \mathbf{0} & \mathbf{C}_1'' & \mathbf{Y}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.24}$$

where

$$\mathbf{C}_1' = \mathbf{C}_1^{\infty} \tag{5.25}$$

$$\mathbf{C}_1'' = \mathbf{C}_2^{\infty} \tag{5.26}$$

$$\mathbf{Y}_1 = \mathbf{C}_1' \mathbf{A}_1 (\mathbf{I} - \mathbf{T})^{-1} \tag{5.27}$$

$$\mathbf{Y}_2 = \mathbf{C}_1'' \mathbf{A}_2 (\mathbf{I} - \mathbf{T})^{-1} \tag{5.28}$$

Essentially $\mathbf{C}_1'$ is the matrix that is associated with $\lambda = 1$ in the expansion of $\mathbf{C}_1$ in terms of its eigenvalues. The same also applies to $\mathbf{C}_1''$ which is the matrix that is associated with $\lambda = 1$ in the expansion of $\mathbf{C}_2$ in terms of its eigenvalues.

We observe that each column of matrix $\mathbf{Y}_1$ is a scaled copy of the columns of $\mathbf{C}_1'$. Also the sum of each column of $\mathbf{Y}_1$ is lesser than one. We can make the same observations about matrix $\mathbf{Y}_2$.

*Example 5.8.* The given transition matrix corresponds to a composite reducible Markov chain.

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.3 & 0 & 0 & 0.1 & 0.4 \\ 0.5 & 0.7 & 0 & 0 & 0.3 & 0.1 \\ 0 & 0 & 0.2 & 0.7 & 0.1 & 0.2 \\ 0 & 0 & 0.8 & 0.3 & 0.1 & 0.1 \\ 0 & 0 & 0 & 0 & 0.1 & 0.2 \\ 0 & 0 & 0 & 0 & 0.3 & 0 \end{bmatrix}$$

Find its eigenvalues and eigenvectors then find the steady-state distribution vector.

The components of the transition matrix are

$$\mathbf{C}_1 = \begin{bmatrix} 0.5 & 0.3 \\ 0.5 & 0.7 \end{bmatrix}$$

$$\mathbf{C}_2 = \begin{bmatrix} 0.2 & 0.7 \\ 0.8 & 0.3 \end{bmatrix}$$

$$\mathbf{A}_1 = \begin{bmatrix} 0.2 & 0.4 \\ 0.2 & 0.1 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} 0.1 & 0.2 \\ 0.1 & 0.1 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.1 & 0.2 \\ 0.3 & 0 \end{bmatrix}$$

The steady-state value of $\mathbf{C}_1$ is

$$\mathbf{C}_1' = \begin{bmatrix} 0.375 & 0.375 \\ 0.625 & 0.625 \end{bmatrix}$$

The matrix $\mathbf{Y}_1$ has the value

$$\mathbf{Y}_1 = \mathbf{C}_1' \, \mathbf{A}_1 \, \, (\mathbf{I} - \mathbf{T})^{-1} = \begin{bmatrix} 0.2455 & 0.2366 \\ 0.4092 & 0.3943 \end{bmatrix}$$

The steady-state value of $\mathbf{C}_2$ is

$$\mathbf{C}_1'' = \begin{bmatrix} 0.4667 & 0.4667 \\ 0.5333 & 0.5333 \end{bmatrix}$$

The matrix $\mathbf{Y}_2$ has the value

$$\mathbf{Y}_2 = \mathbf{C}_1'' \, \mathbf{A}_2 \, \, (\mathbf{I} - \mathbf{T})^{-1} = \begin{bmatrix} 0.1611 & 0.1722 \\ 0.1841 & 0.1968 \end{bmatrix}$$

Thus the steady-state value of $\mathbf{P}$ is

$$\mathbf{P}^{\infty} = \begin{bmatrix} 0.375 & 0.375 & 0 & 0 & 0.2455 & 0.2366 \\ 0.625 & 0.625 & 0 & 0 & 0.4092 & 0.3943 \\ 0 & 0 & 0.4667 & 0.4667 & 0.1611 & 0.1722 \\ 0 & 0 & 0.5333 & 0.5333 & 0.1841 & 0.1968 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We notice that all columns are identical for the closed state matrices. However, the columns for the matrices corresponding to the transient states ($\mathbf{Y}_1$ and $\mathbf{Y}_2$) are not.

The second observation we can make about the transition matrix at steady-state is that there is no possibility of moving to a transient state irrespective of the value of the initial distribution vector.

The third observation we can make is that no matter what the initial distribution vector was, we will always wind up in the same steady-state distribution. ■

## 5.9 Identifying Reducible Markov Chains

We saw above that reducible Markov chains have a transition matrix that can be expressed, by proper reordering of the states, into the canonic form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.29}$$

This rearranged matrix allowed us to determine the closed and transient states. We want to show in this section how to easily identify a reducible Markov chain and how to find its closed and transient states without having to rearrange the matrix. The following theorem helps us determine if our Markov chain is reducible or not by observing the structure of its eigenvector corresponding the eigenvalue $\lambda = 1$.

**Theorem 5.1.** *Let* $\mathbf{P}$ *be the transition matrix of a Markov chain whose eigenvalue* $\lambda = 1$ *corresponds to an eigenvector* $\mathbf{s}$*. Then this chain is reducible if and only if* $\mathbf{s}$ *has one or more zero elements.*

*Proof.* We start by assuming that the eigenvector $\mathbf{s}$ has $k$ nonzero elements and $m-k$ zero elements where $m$ is the number of rows and columns of $\mathbf{P}$. Without loss of generality we can write $\mathbf{s}$ in the canonic form

$$\mathbf{s} = \begin{bmatrix} \mathbf{a} \\ \mathbf{0} \end{bmatrix} \tag{5.30}$$

where the vector $\mathbf{a}$ has $k$ elements none of which is zero such that $0 < k < m$. Partition $\mathbf{P}$ into the form

$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \tag{5.31}$$

where $\mathbf{A}$ is a square $k \times k$ matrix, $\mathbf{D}$ is a square $(m-k) \times (m-k)$ matrix, and the other two matrices are rectangular with the proper dimensions. Since $\mathbf{s}$ is the eigenvector corresponding to $\lambda = 1$, we can write

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{5.32}$$

or

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{a} \\ \mathbf{0} \end{bmatrix} \tag{5.33}$$

This equation results in

$$\mathbf{A}\,\mathbf{a} = \mathbf{a} \tag{5.34}$$

and

$$\mathbf{C}\,\mathbf{a} = \mathbf{0} \tag{5.35}$$

Having $\mathbf{a} = \mathbf{0}$ is contrary to our assumptions. Since the above two equations are valid for any nonzero value of $\mathbf{a}$, we conclude that $\mathbf{A}$ is column stochastic and $\mathbf{C} = \mathbf{0}$.

Thus the transition matrix reduces to the form

$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} \end{bmatrix} \tag{5.36}$$

This is the general canonic form for a reducible Markov chain and this completes one part of the proof.

Now let us assume that $\mathbf{P}$ corresponds to a reducible Markov chain. In that case we can write $\mathbf{P}$ in the canonic form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.37}$$

There are two cases to consider here: $\mathbf{A} = \mathbf{0}$ and $\mathbf{A} \neq \mathbf{0}$.

**Case 1: $\mathbf{A} = \mathbf{0}$**

This is the case when we have

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.38}$$

We have in reality two independent and non-communicating Markov systems. Assume vector $\mathbf{s}$ is the distribution vector associated with the unity eigenvalue for matrix $\mathbf{C}$. In that case we can express $\mathbf{s}$ as

$$\mathbf{s} = \begin{bmatrix} \mathbf{a} & \mathbf{b} \end{bmatrix}^t \tag{5.39}$$

and $\mathbf{s}$ satisfies the equations

$$\begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \tag{5.40}$$

We can write

$$\mathbf{C}\,\mathbf{a} = \mathbf{a} \tag{5.41}$$

$$\mathbf{T}\,\mathbf{b} = \mathbf{b} \tag{5.42}$$

The first equation indicates that $\mathbf{a}$ is an eigenvector of $\mathbf{C}$ and it should be a valid distribution vector. Since the sum of the components of $\mathbf{a}$ must be unity, the sum of the components of $\mathbf{b}$ must be zero which is possible only when

$$\mathbf{b} = \mathbf{0} \tag{5.43}$$

This completes the second part of the proof for Case 1.

The same is true for the eigenvector corresponding to unity eigenvalue for matrix $\mathbf{T}$. In that case $\mathbf{a}$ will be null and $\mathbf{b}$ will be the valid distribution vector. Either way, this completes the second part of the proof for Case 1.

**Case 2: $\mathbf{A} \neq \mathbf{0}$**

This is the case when we have

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.44}$$

In that case $\mathbf{T}$ is a substochastic matrix and $\mathbf{T}^{\infty} = 0$. Now for large time values ($n \to \infty$) we have

$$\mathbf{P}^{\infty} = \begin{bmatrix} \mathbf{X} \\ \mathbf{0} \end{bmatrix} \tag{5.45}$$

But we can also write

$$\mathbf{P}^{\infty}\,\mathbf{s} = \mathbf{s} \tag{5.46}$$

We partition $\mathbf{s}$ into the form

$$\mathbf{s} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \tag{5.47}$$

Substitute the above equation into (5.46) to get

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \tag{5.48}$$

And we get the two equations

$$\mathbf{X}\,\mathbf{a} = \mathbf{a} \tag{5.49}$$

and

$$\mathbf{b} = \mathbf{0} \tag{5.50}$$

Thus the distribution vector corresponding to the eigenvalue $\lambda = 1$ will have the form

$$\mathbf{s} = \begin{bmatrix} \mathbf{a} \\ \mathbf{0} \end{bmatrix} \tag{5.51}$$

This completes the proof of the theorem for Case 2. □

*Example 5.9.* Prove that the given transition matrix corresponds to a reducible Markov chain.

$$\mathbf{P} = \begin{bmatrix} 0 & 0.7 & 0.1 & 0 & 0 \\ 0.3 & 0 & 0.1 & 0 & 0 \\ 0.1 & 0.1 & 0.2 & 0 & 0 \\ 0.4 & 0 & 0.1 & 0.6 & 0.7 \\ 0.2 & 0.2 & 0.5 & 0.4 & 0.3 \end{bmatrix}$$

We calculate the eigenvalues and eigenvectors for the transition matrix. The distribution vector associated with the eigenvalue $\lambda = 1$ is

$$\mathbf{s} = \begin{bmatrix} 0 & 0 & 0 & 0.6364 & 0.3636 \end{bmatrix}^t$$

Since we have zero elements, we conclude that we have a reducible Markov chain. ■

### 5.9.1 Determining Closed and Transient States

Now that we know how to recognize a reducible Markov chain, we need to know how to recognize its closed and transient states. The following theorem provides the answer.

**Theorem 5.2.** *Let* $\mathbf{P}$ *be the transition matrix of a reducible Markov chain whose eigenvalue* $\lambda = 1$ *corresponds to an eigenvector* $\mathbf{s}$*. The closed states of the chain correspond to the nonzero elements of* $\mathbf{s}$ *and the transient states of the chain correspond to the zero elements of* $\mathbf{s}$*.*

*Proof.* Since we are dealing with a reducible Markov chain, then without loss of generality, the transition matrix can be arranged in the canonic form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.52}$$

where it is assumed that $\mathbf{C}$ is a $k \times k$ matrix and $\mathbf{T}$ is a $(m-k) \times (m-k)$ matrix. The first $k$ states correspond to closed states and the last $m-k$ states correspond to transient states.

Assume the eigenvector $\mathbf{s}$ is expressed in the form

$$\mathbf{s} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}^t \tag{5.53}$$

where some of the elements of $\mathbf{s}$ are zero according to the previous Theorem 5.1. Since this is the eigenvector corresponding to unity eigenvalue, we must have

$$\begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \tag{5.54}$$

And we get the two equations

$$\mathbf{C}\,\mathbf{a} + \mathbf{A}\,\mathbf{b} = \mathbf{a} \tag{5.55}$$

and

$$\mathbf{T}\,\mathbf{b} = \mathbf{b} \tag{5.56}$$

The above equation seems to indicate that $\mathbf{T}$ has an eigenvector $\mathbf{b}$ with unity eigenvalue. However, this is a contradiction since $\mathbf{T}$ is column substochastic and it cannot have a unity eigenvalue. The absolute values of all the eigenvalues of $\mathbf{T}$ are less than unity [1]. For such a matrix we say that its spectral radius cannot equal unity.[1] The above equation is only satisfied if

$$\mathbf{b} = \mathbf{0} \tag{5.57}$$

In that case (5.55) becomes

$$\mathbf{C}\,\mathbf{a} = \mathbf{a} \tag{5.58}$$

Thus the eigenvector $\mathbf{s}$ will have the form

$$\mathbf{s} = \begin{bmatrix} \mathbf{a} \\ \mathbf{0} \end{bmatrix}^t \tag{5.59}$$

where $\mathbf{a}$ is a $k$-distribution vector corresponding to unity eigenvalue of $\mathbf{C}$.

[1] Spectral radius equals the largest absolute value of the eigenvalues of a matrix.

We can therefore associate the closed states with the nonzero components of **s** and associate the transient states with the zero components of **s**.

So far we have proven that **s** has the form given in (5.59). We must prove now that *all* the components of **a** are nonzero. This will allow us to state with certainty that any zero component of **s** belongs solely to a transient state.

We prove this by proving that a contradiction results if **a** is assumed to have one or more zero components in it.

Assume that **a** has one or more zero components. We have proven however, that **a** satisfies the equation

$$\mathbf{C}\,\mathbf{a} = \mathbf{a} \tag{5.60}$$

where **C** is a nonreducible matrix. Applying Theorem 5.1, on p. 174, to the above equation would indicate that **C** is reducible. This is a contradiction since **C** is a nonreducible matrix.

Thus the $k$ closed states correspond to the nonzero elements of **s** and the transient states of the chain correspond to the zero elements of **s**. This proves the theorem.□

*Example 5.10.* Prove that the given transition matrix corresponds to a reducible Markov chain.

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.2 & 0.3 & 0.4 & 0.1 & 0.2 \\ 0 & 0.1 & 0.2 & 0 & 0 & 0 \\ 0 & 0.2 & 0.1 & 0 & 0 & 0 \\ 0.4 & 0.1 & 0.2 & 0.1 & 0.2 & 0.3 \\ 0.1 & 0.1 & 0.2 & 0.2 & 0.3 & 0.4 \\ 0.2 & 0.3 & 0 & 0.3 & 0.4 & 0.1 \end{bmatrix}$$

We calculate the eigenvalues and eigenvectors for the transition matrix. The distribution vector associated with the eigenvalue $\lambda = 1$ is

$$\mathbf{s} = \begin{bmatrix} 0.25 & 0 & 0 & 0.25 & 0.25 & 0.25 \end{bmatrix}^t$$

Since we have zero elements, we conclude that we have a reducible Markov chain. The zero elements identify the transient states and the nonzero elements identify the closed states:

| Closed states | Transient states |
|---|---|
| 1, 4, 5, 6 | 2, 3 |

■

## 5.10 Identifying Reducible Composite Matrices

We can generalize Theorem 5.2 as follows. Let $\mathbf{P}$ be the transition matrix of a composite reducible Markov chain with $u$ mutually exclusive closed states corresponding to the sets $C_1, C_2, \cdots C_u$. The canonic form for the transition matrix of such a system will be

$$\mathbf{P} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{C}_1 & \cdots & \mathbf{0} & \mathbf{A}_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{C}_u & \mathbf{A}_u \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{5.61}$$

The eigenvalue $\lambda = 1$ corresponds to the eigenvectors $\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_u$ such that

$$\mathbf{P}\,\mathbf{s}_1 = \mathbf{s}_1 \tag{5.62}$$

$$\mathbf{P}\,\mathbf{s}_2 = \mathbf{s}_2 \tag{5.63}$$

$$\vdots$$

$$\mathbf{P}\,\mathbf{s}_u = \mathbf{s}_u \tag{5.64}$$

The eigenvectors also satisfy the equations

$$\mathbf{C}_1\,\mathbf{s}_1 = \mathbf{s}_1 \tag{5.65}$$

$$\mathbf{C}_2\,\mathbf{s}_2 = \mathbf{s}_2 \tag{5.66}$$

$$\vdots$$

$$\mathbf{C}_u\,\mathbf{s}_u = \mathbf{s}_u \tag{5.67}$$

We can in fact write each eigenvector $\mathbf{s}_i$ in block form as

$$\mathbf{s}_1 = \begin{bmatrix} \mathbf{a}_1\ \mathbf{0}\ \mathbf{0} \cdots \mathbf{0}\ \mathbf{0} \end{bmatrix}^t \tag{5.68}$$

$$\mathbf{s}_2 = \begin{bmatrix} \mathbf{0}\ \mathbf{a}_2\ \mathbf{0} \cdots \mathbf{0}\ \mathbf{0} \end{bmatrix}^t \tag{5.69}$$

$$\vdots$$

$$\mathbf{s}_u = \begin{bmatrix} \mathbf{0}\ \mathbf{0}\ \mathbf{0} \cdots \mathbf{a}_u\ \mathbf{0} \end{bmatrix}^t \tag{5.70}$$

where each $\mathbf{a}_i$ is a nonzero vector whose dimension matches $\mathbf{C}_i$ such that

$$\mathbf{C}_1\,\mathbf{a}_1 = \mathbf{a}_1 \tag{5.71}$$

$$\mathbf{C}_2\ \mathbf{a}_2 = \mathbf{a}_2 \tag{5.72}$$

$$\vdots$$

$$\mathbf{C}_u\ \mathbf{a}_u = \mathbf{a}_u \tag{5.73}$$

which means that $\mathbf{a}_i$ is a distribution (sum of its components is unity).

Vector $\mathbf{s}_i$ corresponds to the set of closed states $C_i$ and the transient states of the chain correspond to the zero elements common to all the vectors $\mathbf{s}_i$.

*Example 5.11.* Assume a composite reducible transition matrix where the number of closed states is $u = 3$ such that the partitioned matrices are:

$$\mathbf{C}_1 = \begin{bmatrix} 0.3 & 0.6 \\ 0.7 & 0.4 \end{bmatrix}$$

$$\mathbf{C}_2 = \begin{bmatrix} 0.5 & 0.1 \\ 0.5 & 0.9 \end{bmatrix}$$

$$\mathbf{C}_3 = \begin{bmatrix} 0.2 & 0.3 \\ 0.8 & 0.7 \end{bmatrix}$$

$$\mathbf{A}_1 = \begin{bmatrix} 0.1 & 0 \\ 0.1 & 0.1 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} 0.3 & 0.1 \\ 0.2 & 0.1 \end{bmatrix}$$

$$\mathbf{A}_3 = \begin{bmatrix} 0 & 0.2 \\ 0.1 & 0 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.1 & 0.2 \\ 0.1 & 0.3 \end{bmatrix}$$

Determine the eigenvectors corresponding to the eigenvalue $\lambda = 1$ and identify the closed and transient states with the elements of those eigenvectors.

The composite transition matrix $\mathbf{P}$ is given by

$$\mathbf{P} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{0} & \mathbf{0} & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{C}_2 & \mathbf{0} & \mathbf{A}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_3 & \mathbf{A}_3 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T} \end{bmatrix}$$

Let us find the eigenvalues and eigenvectors for $\mathbf{P}$. The eigenvalues are

$$\lambda_1 = -0.3$$

$$\lambda_2 = \ \ 1$$

$$\begin{aligned}\lambda_3 &= 0.4\\ \lambda_4 &= 1\\ \lambda_5 &= -0.1\\ \lambda_6 &= 1\\ \lambda_7 &= 0.0268\\ \lambda_8 &= 0.3732\end{aligned}$$

The eigenvectors corresponding to unity eigenvalue (after normalization so their sums is unity) are

$$\begin{aligned}\mathbf{s}_1 &= \begin{bmatrix}0.4615 & 0.5385 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^t\\ \mathbf{s}_2 &= \begin{bmatrix}0 & 0 & 0.1667 & 0.8333 & 0 & 0 & 0 & 0\end{bmatrix}^t\\ \mathbf{s}_3 &= \begin{bmatrix}0 & 0 & 0 & 0 & 0.2727 & 0.7273 & 0 & 0\end{bmatrix}^t\end{aligned}$$

The sets of closed and transient states are as follows:

| Set | States |
|---|---|
| $C_1$ | 1, 2 |
| $C_2$ | 3, 4 |
| $C_3$ | 5, 6 |
| $T$ | 7, 8 |

■

## 5.11 Problems

### *Reducible Markov Chains*

For Problems 5.1–5.8: (a) Determine whether the given Markov matrices have absorbing or closed states; (b) Express such matrices in the form given in Eqs. (5.2) or (5.3); (c) Identify the component matrices **C**, **A**, and **T**; and (d) Identify the closed and transient states.

**5.1.**

$$\mathbf{P} = \begin{bmatrix}0.3 & 0\\ 0.7 & 1\end{bmatrix}$$

**5.2.**

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0 & 0.2 \\ 0.3 & 1 & 0.3 \\ 0.2 & 0 & 0.5 \end{bmatrix}$$

**5.3.**

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0.3 & 0.5 & 0 \\ 0.2 & 0 & 1 \end{bmatrix}$$

**5.4.**

$$\mathbf{P} = \begin{bmatrix} 0.7 & 0.5 & 0.1 \\ 0.3 & 0.5 & 0 \\ 0 & 0 & .9 \end{bmatrix}$$

**5.5.**

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0 & 0 \\ 0.3 & 1 & 0 \\ 0.5 & 0 & 1 \end{bmatrix}$$

**5.6.**

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0 & 0.5 & 0 \\ 0.2 & 1 & 0 & 0 \\ 0.3 & 0 & 0.5 & 0 \\ 0.4 & 0 & 0 & 1 \end{bmatrix}$$

**5.7.**

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0.2 \\ 0 & 0.2 & 1 & 0 \\ 0 & 0.3 & 0 & 0.8 \end{bmatrix}$$

**5.8.**

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0.5 \\ 0 & 0.2 & 1 & 0 \\ 0 & 0.7 & 0 & 0.5 \end{bmatrix}$$

## *Composite Reducible Markov Chains*

The transition matrices in Problems 5.9–5.12 represent composite reducible Markov chains. Identify the sets of closed and transient states, find the eigenvalues and eigenvectors for the matrices, and find the value of each matrix for large values of $n$, say when $n = 50$.

**5.9.**

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0.3 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 1 & 0.6 \\ 0 & 0 & 0 & 0.1 \end{bmatrix}$$

**5.10.**

$$\mathbf{P} = \begin{bmatrix} 0.7 & 0 & 0 & 0 \\ 0.1 & 0.9 & 0.2 & 0 \\ 0.1 & 0.1 & 0.8 & 0 \\ 0.1 & 0 & 0 & 1 \end{bmatrix}$$

**5.11.**

$$\mathbf{P} = \begin{bmatrix} 0.4 & 0 & 0 & 0 & 0 \\ 0.2 & 0.5 & 0.8 & 0 & 0 \\ 0.1 & 0.5 & 0.2 & 0 & 0 \\ 0.1 & 0 & 0 & 0.7 & 0.9 \\ 0.2 & 0 & 0 & 0.3 & 0.1 \end{bmatrix}$$

**5.12.**

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0 & 0 & 0.2 & 0 & 0.6 \\ 0 & 0.3 & 0 & 0.3 & 0 & 0 \\ 0 & 0 & 0.2 & 0.2 & 0.3 & 0 \\ 0 & 0.3 & 0 & 0.3 & 0 & 0 \\ 0 & 0.2 & 0.8 & 0 & 0.7 & 0 \\ 0.8 & 0.2 & 0 & 0 & 0 & 0.4 \end{bmatrix}$$

## *Transient Analysis*

**5.13.** Check whether the given transitions matrix is reducible or irreducible. Identify the closed and transient states and express the matrix in the form of Eqs. (5.2) or (5.3) and identify the component matrices **C**, **A**, and **T**.

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0.5 \\ 0 & 0.5 & 0 & 0.25 & 0 \\ 0 & 0 & 1 & 0.25 & 0 \\ 0 & 0.5 & 0 & 0.25 & 0 \\ 0.5 & 0 & 0 & 0.25 & 0.5 \end{bmatrix}$$

Find the value of $\mathbf{P}^{10}$ using (5.8) and (5.9) on p. 164, and verify your results using repeated multiplications.

**5.14.** Assume the transition matrix $\mathbf{P}$ has the structure given in Eqs. (5.1) or (5.2) Prove that $\mathbf{P}^n$ also possesses the same structure as the original matrix and prove also that the component matrices $\mathbf{C}$, $\mathbf{A}$, and $\mathbf{T}$ have the same properties as the original component matrices.

**5.15.** Find an expression for the transition matrix, using (5.13) at time $n = 4$ and $n = 20$ for the reducible Markov chain characterized by the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0.7 & 0.9 & 0.3 & 0.3 & 0.2 \\ 0.3 & 0.1 & 0.2 & 0.1 & 0.3 \\ 0 & 0 & 0.2 & 0.2 & 0 \\ 0 & 0 & 0.2 & 0.2 & 0.3 \\ 0 & 0 & 0.1 & 0.2 & 0.2 \end{bmatrix}$$

Find the value of $\mathbf{P}^{10}$ using (5.8) and verify your results using repeated multiplications.

## *Reducible Markov Chains at Steady-State*

**5.16.** In Sect. 5.7 it was asserted that the transition matrix for a reducible Markov chain will have the form in (5.22) where all the columns of the matrix are identical. Prove that assertion knowing that:

(a) All the columns of $\mathbf{C}_1$ are all identical.
(b) Matrix $\mathbf{C}_1$ is column stochastic.
(c) Matrix $\mathbf{Y}^{\infty}$ is column stochastic.
(d) The columns of $\mathbf{Y}^{\infty}$ are identical to the columns of $\mathbf{C}_1$.

**5.17.** Find the steady-state transition matrix and distribution vector for the reducible Markov chain characterized by the matrix

$$\mathbf{P} = \begin{bmatrix} 0.9 & 0.2 & 0.5 & 0.1 & 0.1 \\ 0.1 & 0.8 & 0.1 & 0.2 & 0.3 \\ 0 & 0 & 0.2 & 0.2 & 0.1 \\ 0 & 0 & 0.2 & 0.2 & 0.1 \\ 0 & 0 & 0 & 0.3 & 0.4 \end{bmatrix}$$

**5.18.** Find the steady-state transition matrix and distribution vector for the reducible Markov chain characterized by the matrix

$$\mathbf{P} = \begin{bmatrix} 0.9 & 0.2 & 0.5 & 0.1 \\ 0.1 & 0.8 & 0.1 & 0.2 \\ 0 & 0 & 0.4 & 0.2 \\ 0 & 0 & 0 & 0.5 \end{bmatrix}$$

**5.19.** Find the steady-state transition matrix and distribution vector for the reducible Markov chain characterized by the matrix

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.2 & 0.5 & 0.1 & 0.1 \\ 0.5 & 0.7 & 0.1 & 0.2 & 0.3 \\ 0.4 & 0.1 & 0.4 & 0.2 & 0.1 \\ 0 & 0 & 0 & 0.2 & 0.1 \\ 0 & 0 & 0 & 0.3 & 0.4 \end{bmatrix}$$

**5.20.** Find the steady-state transition matrix and distribution vector for the reducible Markov chain characterized by the matrix

$$\mathbf{P} = \begin{bmatrix} 1 & 0.2 & 0.2 & 0.1 & 0.1 \\ 0 & 0.3 & 0.1 & 0.2 & 0.3 \\ 0 & 0.1 & 0.4 & 0.2 & 0.1 \\ 0 & 0.3 & 0.1 & 0.2 & 0.1 \\ 0 & 0.1 & 0.2 & 0.3 & 0.4 \end{bmatrix}$$

**5.21.** Find the steady state transition matrix and distribution vector for the reducible Markov chain characterized by the matrix

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0.2 & 0.1 & 0.1 \\ 0 & 1 & 0.1 & 0.2 & 0.3 \\ 0 & 0 & 0.4 & 0.2 & 0.1 \\ 0 & 0 & 0.1 & 0.2 & 0.1 \\ 0 & 0 & 0.2 & 0.3 & 0.4 \end{bmatrix}$$

Note that this matrix has two absorbing states.

**5.22.** Consider the state transition matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 \\ 1-p & 0 & q \\ p & 0 & 1-q \end{bmatrix}$$

(a) Can this matrix represent a reducible Markov chain?
(b) Find the distribution vector at equilibrium.
(c) What values of $p$ and $q$ give $s_0 = s_1 = s_2$?

**5.23.** Consider a discrete-time Markov chain in which the transition probabilities are given by

$$p_{ij} = q^{|i-j|}p$$

For a $3 \times 3$ case, what are the values of $p$ and $q$ to make this a reducible Markov chain? What are the values of $p$ and $q$ to make this an irreducible Markov chain and find the steady-state distribution vector.

**5.24.** Consider the coin tossing Example 5.3 on p. 163. Derive the equilibrium distribution vector and comment on it for the cases $p < q$, $p = q$, and $p > q$.

**5.25.** Rewrite (5.10) on p. 165 to take into account the fact that some of the eigenvalues of $\mathbf{C}$ might be repeated using the results of Sect. 3.14 on p. 105.

## *Identification of Reducible Markov Chains*

Use the results of Sect. 5.9 to verify that the transition matrices in the following problems correspond to reducible Markov chains and identify the closed and transient states. Rearrange each matrix to the standard form in (5.2) or (5.3).

**5.26.**

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.1 & 0.4 \\ 0 & 0.1 & 0 \\ 0.7 & 0.8 & 0.6 \end{bmatrix}$$

**5.27.**

$$\mathbf{P} = \begin{bmatrix} 0.4 & 0.6 & 0 \\ 0.4 & 0.3 & 0 \\ 0.2 & 0.1 & 1 \end{bmatrix}$$

**5.28.**

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.3 & 0 & 0 \\ 0.1 & 0.1 & 0 & 0 \\ 0.3 & 0.4 & 0.2 & 0.9 \\ 0.1 & 0.2 & 0.8 & 0.1 \end{bmatrix}$$

**5.29.**

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0.3 & 0.1 & 0.5 & 0.3 \\ 0.4 & 0.3 & 0.4 & 0.2 \\ 0.2 & 0.6 & 0.1 & 0.5 \end{bmatrix}$$

**5.30.**

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0.3 & 0 & 0.3 & 0 \\ 0.2 & 0.4 & 0 & 0 & 0 \\ 0.3 & 0 & 0.5 & 0.1 & 0.2 \\ 0.2 & 0 & 0 & 0.4 & 0 \\ 0.1 & 0.3 & 0.5 & 0.2 & 0.8 \end{bmatrix}$$

**5.31.**

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0.1 & 0.3 & 0.1 & 0.6 \\ 0 & 0.3 & 0 & 0.2 & 0 \\ 0.2 & 0.2 & 0.4 & 0.1 & 0.3 \\ 0 & 0.1 & 0 & 0.4 & 0 \\ 0.6 & 0.3 & 0.3 & 0.2 & 0.1 \end{bmatrix}$$

**5.32.**

$$\mathbf{P} = \begin{bmatrix} 0.8 & 0.4 & 0 & 0.5 & 0 \\ 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0.8 & 0 & 0.3 \\ 0.2 & 0.2 & 0 & 0.5 & 0 \\ 0 & 0.2 & 0.2 & 0 & 0.7 \end{bmatrix}$$

**5.33.**

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0 & 0.2 & 0 & 0 & 0.3 \\ 0.2 & 0.5 & 0.1 & 0 & 0.4 & 0.1 \\ 0.1 & 0 & 0.3 & 0 & 0 & 0.3 \\ 0.2 & 0 & 0.1 & 1 & 0 & 0 \\ 0.3 & 0.5 & 0.2 & 0 & 0.6 & 0.1 \\ 0.1 & 0 & 0.1 & 0 & 0 & 0.2 \end{bmatrix}$$

**5.34.**

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.5 & 0 & 0.1 & 0 & 0 & 0.6 \\ 0.7 & 0.3 & 0 & 0.2 & 0 & 0 & 0.3 \\ 0 & 0 & 0.5 & 0.3 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & 0 & 0.3 & 0 \\ 0 & 0 & 0.5 & 0 & 0.9 & 0.2 & 0 \\ 0 & 0 & 0 & 0.2 & 0 & 0.4 & 0 \\ 0.2 & 0.2 & 0 & 0 & 0 & 0.1 & 0.1 \end{bmatrix}$$

## Reference

1. R.A. Horn, C.R. Johnson, *Matrix Analysis* (Cambridge University Press, Cambridge, 1985)

# Chapter 6
# Periodic Markov Chains

## 6.1 Introduction

We saw in Chap. 4 that a Markov chain settles down to steady-state distribution vector $\mathbf{s}$ when $n \to \infty$. This is true for most transition matrices representing most Markov chains we studied. There are other times however when the Markov chain never settles down to an equilibrium distribution vector no matter how long we iterate. So this chapter will illustrate *periodic Markov chains* whose distribution vector $\mathbf{s}(n)$ repeats its values at regular intervals of time and never settles down to an equilibrium value no matter how long we iterate.

Periodic Markov chains could be found in systems that show repetitive behavior or task sequences. An intuitive example of a periodic Markov chain is the population of wild salmon. In that fish species, we can divide the life cycle as eggs, hatchlings, subadults, and adults. Once the adults reproduce, they die and the resulting eggs hatch and repeat the cycle as shown in Fig. 6.1. Fluctuations in the salmon population can thus be modeled as a periodic Markov chain. It is interesting to note that other fishes that lay eggs without dying can be modeled as nonperiodic Markov chains.

Another classic example from nature where periodic Markov chains apply is the predator–prey relation—where the population of deer, say, is related to the population of wolves. When deer numbers are low, the wolf population is low. This results in more infant deer survival rate and the deer population grows during the next year. When this occurs, the wolves start having more puppies and the wolf population also increases. However, the large number of wolves results in more deer kills and the deer population diminishes. The reduced number of deer results in wolf starvation and the number of wolves decreases also. This cycle repeats as discussed in Problem 6.14.

Another example for periodic Markov chains in communications is data transmission. In such a system data is packetized to be transmitted then the packets are sent over a channel. The received packets are then analyzed for presence of errors. Based on the number of bits in error the receiver is able to correct for the errors



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_6

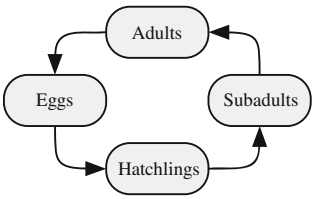



**Fig. 6.1** Wild salmon can be modeled as a periodic Markov chain with the states representing the number of each phase of the fish life cycle

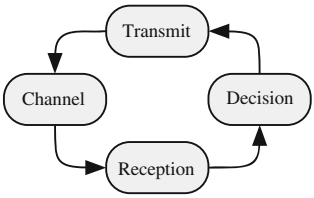



**Fig. 6.2** Data communication over a noisy channel can be modeled as a periodic Markov chain with the states representing the state of each phase

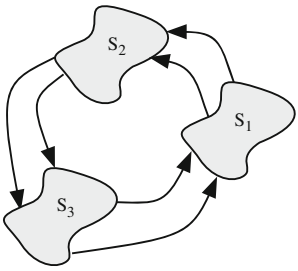



**Fig. 6.3** A periodic Markov chain where states are divided into groups and allowed transitions occur only between adjacent groups

or inform the transmitter to retransmit perhaps even with a higher level of data redundancy. Figure 6.2 shows these states which are modeled as a periodic Markov chain. In each transmission phase, there could be several states indicating the level of decoding required or the number of random errors introduced.

Consider the abstract transition diagram shown in Fig. 6.3 where the states of the Markov chain are divided into groups and allowed transitions occur only between adjacent groups. The sets of states $S_1$ , $S_2$, $\cdots$ are called *periodic classes* of the Markov chain. A state in set $S_1$ is allowed to make a transition to any other state in set $S_2$ only. Thus the states in the set $S_1$ cannot make a transition to any state in $S_1$ or $S_3$. A similar argument applies to the states in sets $S_2$ or $S_3$.

## 6.2 Definition

A *periodic Markov chain* has the property that the number of single-step transitions that must be made after leaving a state to return to that state is a multiple of some integer $\gamma > 1$ [1]. This definition implies that the distribution vector never settles down to a fixed steady state value no matter how long we iterate.

Having mentioned that Markov chains could be periodic, we naturally want to know how to recognize that the transition matrix **P** represents a periodic Markov

chain. In addition, we will want also to know the period of such a system. The theorems presented here will specify the properties of the transition matrix when the Markov chain is periodic.

Of course, a Markov chain that is not periodic would be called nonperiodic. A *nonperiodic Markov chain* does not show repetitive behavior as time progresses and its distribution vector settles down to a fixed steady state value.

## 6.3 Types of Periodic Markov Chains

There are two types of periodic Markov chains

1. *Strongly periodic Markov chains* where the distribution vector repeats its values with a period $\gamma > 1$. We will find out in the next section that the state transition matrix satisfies the relation

$$\mathbf{P}^{\gamma} = \mathbf{I} \tag{6.1}$$

   In other words, in a strongly periodic Markov chain the probability of returning to the starting state after $\gamma$ time steps is unity for all states of the system.
2. *Weakly periodic Markov chains*, where the system shows periodic behavior only when $n \to \infty$. In other words, the distribution vector repeats its values with a period $\gamma > 1$ only when $n \to \infty$. We will find out in Sect. 6.12 that the state transition matrix satisfies the relation

$$\mathbf{P}^{\gamma} \neq \mathbf{I} \tag{6.2}$$

   In other words, in a weakly periodic Markov chain the probability of returning to the starting state after $\gamma$ time steps is less than unity for some or all states of the system.

In both cases there is no equilibrium distribution vector. Strongly periodic Markov chains are not encountered in practice since they are a special case of the more widely encountered weakly periodic Markov chains. We start this chapter, however, with the strongly periodic Markov chains since they are easier to study and they will pave the ground for studying the weakly periodic type.

## 6.4 The Transition Matrix

Let us start by making general observations on the transition matrix of a strongly periodic Markov chain. Assume that somehow we have a strongly periodic Markov chain with period $\gamma$. Then the probability of making a transition from state $j$ to state $i$ at time instant $n$ will repeat its value at instant $n + \gamma$. This is true for all valid values of $i$ and $j$:

$$p_{ij}(n+\gamma) = p_{ij}(n) \tag{6.3}$$

It is remarkable that this equation is valid for all valid values of $n$, $i$, and $j$. But then again, this is what a strongly periodic Markov chain does.

We can apply the above equation to the components of the distribution vector and write

$$\mathbf{s}(n+\gamma) = \mathbf{s}(n) \tag{6.4}$$

But the two distributions vectors are also related to each other through the transition matrix

$$\mathbf{s}(n+\gamma) = \mathbf{P}^{\gamma}\, \mathbf{s}(n) \tag{6.5}$$

From the above two equations we can write

$$\mathbf{P}^{\gamma}\, \mathbf{s}(n) = \mathbf{s}(n) \tag{6.6}$$

or

$$(\mathbf{P}^{\gamma} - \mathbf{I})\ \mathbf{s}(n) = \mathbf{0} \tag{6.7}$$

and this equation is valid for all values of $n$ and $\mathbf{s}(\mathbf{n})$. The solution to the above equations is

$$\mathbf{P}^{\gamma} = \mathbf{I} \tag{6.8}$$

where $\mathbf{I}$ is the unit matrix and $\gamma > 1$. The case when $\gamma = 1$ is trivial which indicates that $\mathbf{P}$ is the identity matrix.

*Example 6.1.* The following transition matrix corresponds to a strongly periodic Markov chain. Estimate the period of the chain.

$$\mathbf{P} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

We start by performing repeated multiplications to see when $\mathbf{P}^k = \mathbf{I}$ for some value of $k$. We are lucky since

$$\mathbf{P}^2 = \mathbf{I}$$

The period of this Markov chain is $\gamma = 2$. The given transition matrix is also known as a circulant matrix where the adjacent rows or columns advance by one position. The matrix $\mathbf{P}$ could also be considered a permutation matrix or exchange matrix where rows 1 and 2 are exchanged after premultiplying any $2 \times m$ matrix [2]. ■

## 6.5 The Transition Matrix Determinant

This section provides one specification on the transition matrix of a strongly periodic Markov chain, Theorem 6.1 indicates the allowed values of the determinant of the transition matrix.

**Theorem 6.1.** *Let* $\mathbf{P}$ *be the transition matrix of a strongly periodic Markov chain. The determinant of* $\mathbf{P}$ *will be given by [3]*

$$\Delta = \pm 1$$

*Proof.* We start by assuming that the Markov chain is strongly periodic. We have from the assumptions

$$\mathbf{P}^{\gamma} = \mathbf{I} \tag{6.9}$$

Equate the determinants of both sides

$$\Delta^{\gamma} = 1 \tag{6.10}$$

where $\Delta$ is the determinant of $\mathbf{P}$ and $\Delta^{\gamma}$ is the determinant of $\mathbf{P}^{\gamma}$.

Taking the $\gamma$-root of the above equation we find that $\Delta$ is the $\gamma$-root of unity:

$$\Delta = \exp\left(j2\pi \times \frac{k}{\gamma}\right) \qquad k = 1, 2, \cdots, \gamma \tag{6.11}$$

But $\Delta$ must be real since the components of $\mathbf{P}$ are all real. Thus the only possible values for $\Delta$ are $\pm 1$. This proves the theorem.

□

From the properties of the determinant of the transition matrix of a strongly periodic Markov chain, we conclude that $\mathbf{P}$ must have the following equivalent properties

1. The $m \times m$ transition matrix $\mathbf{P}$ of a strongly periodic Markov chain is full rank, i.e. $\text{rank}(\mathbf{P}) = m$.
2. The rows and columns of the transition matrix $\mathbf{P}$ of a strongly periodic Markov chain are linearly independent.
3. $\lambda < 1$ can never be an eigenvalue for the transition matrix $\mathbf{P}$ of a strongly periodic Markov chain. Any value of $\lambda < 1$ would produce a determinant that is not equal to $\pm 1$.
4. All eigenvalues must obey the relation $|\lambda| = 1$. This will be proven in the next section.

## 6.6 Transition Matrix Diagonalization

The following theorem indicates that the transition matrix of a strongly periodic Markov chain can be diagonalized. This fact leads naturally to great simplification in the study of strongly periodic Markov chains.

**Theorem 6.2.** *Let* $\mathbf{P}$ *be the transition matrix of a strongly periodic Markov chain with period* $\gamma > 1$. *Then* $\mathbf{P}$ *is diagonalizable.*

*Proof.* If $\mathbf{P}$ is diagonalizable, then its Jordan canonic form will turn into a diagonal matrix. Let us assume that $\mathbf{P}$ is not diagonalizable. In that case $\mathbf{P}$ is similar to its Jordan canonic form

$$\mathbf{P} = \mathbf{U}\mathbf{J}\mathbf{U}^{-1} \tag{6.12}$$

Since $\mathbf{P}$ is periodic, we must have

$$\mathbf{P}^{\gamma} = \mathbf{U}\mathbf{J}^{\gamma}U^{-1} = \mathbf{I} \tag{6.13}$$

Multiplying both sides of the equation from the right by $\mathbf{U}$ we get

$$\mathbf{U}\mathbf{J}^{\gamma} = \mathbf{I}\mathbf{U} = \mathbf{U}\mathbf{I} \tag{6.14}$$

This implies that we must have

$$\mathbf{J}^{\gamma} = \mathbf{I} \tag{6.15}$$

The above equation states that the matrix $\mathbf{J}^{\gamma}$ is equal to the diagonal matrix $\mathbf{I}$. However, $\mathbf{J}$ can never be diagonal if it is not already so. Thus the above equation is only possible when the Jordan canonic form $\mathbf{J}$ is diagonal which happens when $\mathbf{P}$ is diagonalizable and the theorem is proved.

□

*Example 6.2.* Verify that the following transition matrix is diagonalizable.

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We start by finding the Jordan canonic form for the matrix $\mathbf{P}$.

```
[V,J] = jordan(P)

V =
 0.3333    0.3333             0.3333                    0
```

```
    0.3333   -0.1667 - 0.2887j   -0.1667 + 0.2887j         0
    0.3333   -0.1667 + 0.2887j   -0.1667 - 0.2887j         0
    1.0000         0                    0              1.0000

J =

    1.0000         0                    0                   0
         0    -0.5000 + 0.8660j         0                   0
         0         0              -0.5000 - 0.8660j         0
         0         0                    0              1.0000
```

Thus we see that **P** is diagonalizable. We also see that all the eigenvalues all lie on the unit circle.

$$\lambda_1 = \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_2 = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_3 = \exp\left(j2\pi \times \frac{3}{3}\right)$$

$$\lambda_4 = \exp\left(j2\pi \times \frac{3}{3}\right)$$

where $j = \sqrt{-1}$. ■

The following theorem will add one more specification on the transition matrix of a strongly periodic Markov chain.

**Theorem 6.3.** *Let* **P** *be the transition matrix of a strongly periodic Markov chain. Then* **P** *is unitary (orthogonal)*

*Proof.* Assume **x** is an eigenvector of the transition matrix **P**, then we can write

$$\mathbf{P}\,\mathbf{x} = \lambda\,\mathbf{x} \tag{6.16}$$

Transposing and taking the complex conjugate of both sides of the above equation, we get

$$\mathbf{x}^H\,\mathbf{P}^H = \lambda^*\,\mathbf{x}^H \tag{6.17}$$

where the symbol $H$ indicates complex conjugate of the transposed matrix or vector and $\lambda^*$ is the complex conjugate of $\lambda$.

Now multiply the corresponding sides of Eqs. (6.16) and (6.17) to get

$$\mathbf{x}^H\,\mathbf{P}^H\,\mathbf{P}\,\mathbf{x} = \lambda^*\,\lambda\,\mathbf{x}^H\,\mathbf{x} \tag{6.18}$$

or

$$\mathbf{x}^H \, \mathbf{P}^H \, \mathbf{P} \, \mathbf{x} = |\lambda|^2 \; |\mathbf{x}|^2 \tag{6.19}$$

where

$$|\mathbf{x}|^2 = x_1^* \; x_1 + x_2^* \; x_2 + \cdots + x_m^* \; x_m \tag{6.20}$$

We know from Theorem 6.5 that the eigenvalues of $\mathbf{P}$ lie on the unit circle, and (6.19) can be written as

$$\mathbf{x}^H \; \mathbf{Y} \; \mathbf{x} = |\mathbf{x}|^2 \tag{6.21}$$

where $\mathbf{Y} = \mathbf{P}^H \; \mathbf{P}$. The above equation can be written as

$$\sum_{i=1}^{m} \sum_{j=1}^{m} x_i^* \; y_{ij} \; x_j = \sum_{i=1}^{m} x_i^* \; x_i \tag{6.22}$$

This equation can only be satisfied for arbitrary values of the eigenvectors when

$$\sum_{j=1}^{m} x_i^* \; y_{ij} \; x_j = x_i^* \; x_i \tag{6.23}$$

Similarly, this equation can only be satisfied for arbitrary values of the eigenvectors when

$$y_{ij} = \delta_{ij} \tag{6.24}$$

where $\delta_{ij}$ is the Kronecker delta which satisfies the equation

$$\delta_{ij} = \begin{cases} 1 & \text{when } i = j \\ 0 & \text{when } i \neq j \end{cases} \tag{6.25}$$

We conclude, therefore, that

$$\mathbf{P}^H \; \mathbf{P} = \mathbf{I} \tag{6.26}$$

Thus $\mathbf{P}$ is a unitary matrix whose inverse equals the complex conjugate of its transpose. Since $\mathbf{P}$ is real, the unitary matrix is usually called orthogonal matrix. This proves the theorem.

□

## 6.7 Transition Matrix Eigenvalues

So far we have discovered several restrictions on the transition matrix of a strongly periodic Markov chain. The following theorem specifies the allowed magnitudes of the eigenvalues for strongly periodic Markov chains.

**Theorem 6.4.** *Let* **P** *be the transition matrix of a Markov chain. The Markov chain will be strongly periodic if and only if the eigenvalues of* **P** *all lie on the unit circle.*

*Proof.* Let us start by assuming that the Markov chain is strongly periodic. According to Theorem 6.1 the determinant of the transition matrix is

$$\Delta = \pm 1$$

The determinant can be written as the product of all the eigenvalues of the transition matrix

$$\Delta = \prod_i \lambda_i \tag{6.27}$$

but we know that $\Delta = \pm 1$ and we can write

$$\prod_i \lambda_i = \pm 1 \tag{6.28}$$

Since **P** is column stochastic, then all its eigenvalues must satisfy the inequality

$$|\lambda_i| \leq 1 \tag{6.29}$$

The above inequality together with (6.28) implies that

$$|\lambda_i| = 1 \tag{6.30}$$

This proves one side of the theorem.

Let us now assume that all the eigenvalues of **P** all lie on the unit circle. In that case we can write the eigenvalues as

$$\lambda_i = \exp\left(\frac{j2\pi}{\gamma_i}\right) \tag{6.31}$$

Therefore we can write the transition matrix determinant as

$$\Delta = \prod_i \lambda_i \tag{6.32}$$

Raising the above equation to some power $\gamma$, we get

$$\Delta^{\gamma} = \prod_i \lambda_i^{\gamma} \tag{6.33}$$

Thus we have

$$\Delta^{\gamma} = \exp\left[ j2\pi \left( \frac{\gamma}{\gamma_1} + \frac{\gamma}{\gamma_2} + \cdots + \frac{\gamma}{\gamma_m} \right) \right] \tag{6.34}$$

If $\gamma$ is chosen to satisfy the equation

$$\gamma = \text{lcm}(\gamma_i) \tag{6.35}$$

where *lcm* is the least common multiple, then we can write

$$\Delta^{\gamma} = 1 \tag{6.36}$$

According to Theorem 6.1 this proves that the Markov chain is strongly periodic. This proves the other part of the theorem.

□

Figure 6.4 shows the locations of the eigenvalues of **P** in the complex plane. We see that all the eigenvalues of a strongly periodic Markov chain lie on the unit circle as indicated by the ×'s. Since **P** is column stochastic, the eigenvalue $\lambda = 1$ must be present, as is also indicated. The figure also indicates that complex eigenvalues appear in complex conjugate pairs.

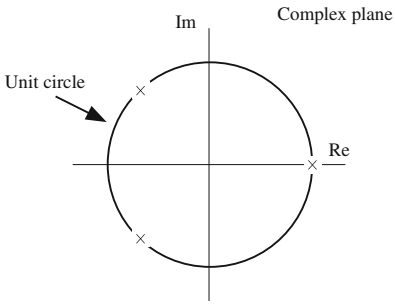



**Fig. 6.4** The eigenvalues of a strongly periodic Markov chain all lie on the unit circle in the complex plane

*Example 6.3.* Consider the following transition matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Verify that this matrix is strongly periodic and find its period.

The determinant is $det(\mathbf{P}) = 1$ and the rank of the transition matrix is $rank(\mathbf{P}) = 4$.

Performing repeated multiplications we find that

$$\mathbf{P}^3 = \mathbf{I}$$

The period of the given transition matrix is $\gamma = 3$.

The eigenvalues of the transition matrix are

$$\lambda_1 = \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_2 = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_3 = \exp\left(j2\pi \times \frac{3}{3}\right)$$

$$\lambda_4 = \exp\left(j2\pi \times \frac{3}{3}\right)$$

Thus all the eigenvalues lie on the unit circle. ■

The following theorem defines the allowed eigenvalues for the transition matrix of a strongly periodic Markov chain.

**Theorem 6.5.** *Let* $\mathbf{P}$ *be the transition matrix of an irreducible strongly periodic Markov chain with period* $\gamma > 1$. *Then the eigenvalues of* $\mathbf{P}$ *are the* $\gamma$*-roots of unity, i.e.*

$$\lambda_i = exp\left(j2\pi \times \frac{i}{\gamma}\right) \qquad i = 1, 2, \cdots, \gamma \tag{6.37}$$

*where* $j = \sqrt{-1}$.

*Proof.* Since we proved in Theorem 6.2 that $\mathbf{P}$ is diagonalizable, then it is similar to a diagonal matrix $\mathbf{D}$ such that

$$\mathbf{P} = \mathbf{X}\mathbf{D}\mathbf{X}^{-1} \tag{6.38}$$

where $\mathbf{X}$ is the matrix whose columns are the eigenvectors of $\mathbf{P}$ and $\mathbf{D}$ is the diagonal matrix whose diagonal elements are the eigenvalues of $\mathbf{P}$.

Since $\mathbf{P}$ is periodic, we must have

$$\mathbf{P}^{\gamma} = \mathbf{X}\mathbf{D}^{\gamma}\mathbf{X}^{-1} = \mathbf{I} \tag{6.39}$$

Multiplying both sides from the right by $\mathbf{X}$, we get

$$\mathbf{X}\mathbf{D}^{\gamma} = \mathbf{I}\mathbf{X} = \mathbf{X}\mathbf{I} \tag{6.40}$$

Therefore we must have

$$\mathbf{D}^{\gamma} = \mathbf{I} \tag{6.41}$$

This implies that any diagonal element $d_i^{\gamma}$ of $\mathbf{D}^{\gamma}$ must obey the relation

$$d_i^{\gamma} = 1 \tag{6.42}$$

Therefore, the eigenvalues of $\mathbf{P}$ are the $\gamma$-root of unity and are given by the equation

$$d_i = \exp\left(j2\pi \times \frac{i}{\gamma}\right) \qquad i = 1, 2, \cdots m \tag{6.43}$$

This proves the theorem. Note that the theorem does not preclude repeated eigenvalues having the same value as long as $\mathbf{P}$ can be diagonalized.

□

*Example 6.4.* The given transition matrix corresponds to a strongly periodic Markov chain. Confirm the conclusions of Theorems 6.1, 6.2, and 6.5

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

The period of the given matrix is 4 since $\mathbf{P}^4 = \mathbf{I}$. Using MATLAB, we find that the determinant of $\mathbf{P}$ is $\Delta = -1$. The eigenvalues of $\mathbf{P}$ are

$$\lambda_1 = \quad 1 = \exp\left(j2\pi \times \frac{4}{4}\right)$$

$$\lambda_2 = \quad j = \exp\left(j2\pi \times \frac{1}{4}\right)$$

$$\lambda_3 = -j = \exp\left(j2\pi \times \frac{3}{4}\right)$$
$$\lambda_4 = -1 = \exp\left(j2\pi \times \frac{2}{4}\right)$$

The matrix can be diagonalized as

$$\mathbf{P} = \mathbf{XDX}^{-1}$$

where

$$\mathbf{X} = \begin{bmatrix} 0.5 & -0.49 - 0.11j & -0.49 + 0.11j & -0.5 \\ -0.5 & 0.11 - 0.40j & 0.11 + 0.49j & -0.5 \\ 0.5 & 0.40 + 0.11j & 0.49 - 0.11j & -0.5 \\ -0.5 & -0.11 + 0.40j & -0.11 - 0.49j & -0.5 \end{bmatrix}$$

and

$$\mathbf{D} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & j & 0 & 0 \\ 0 & 0 & -j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We see that Theorems 6.1, 6.2, and 6.5 are verified. ■

## 6.8 Transition Matrix Elements

The following theorem imposes surprising restrictions on the values of the elements of the transition matrix for a strongly periodic Markov chain.

**Theorem 6.6.** *Let* $\mathbf{P}$ *be the* $m \times m$ *transition matrix of a Markov chain. The Markov chain is strongly periodic if and only if the elements of* $\mathbf{P}$ *are all zeros except for* $m$ *elements that have 1's arranged such that each column and each row contains only a single 1 entry in a unique location.*

*Proof.* We begin by assuming that the transition matrix $\mathbf{P}$ corresponds to a strongly periodic Markov chain. We proved that $\mathbf{P}$ must be unitary which implies that the rows and columns are orthonormal. This implies that no two rows shall have nonzero elements at the same location since all elements are nonnegative. Similarly, no two columns shall have nonzero elements at the same location. This is only possible if all rows and columns are zero except for a single element at a unique location in each row or column.

Further, since we are dealing with a Markov matrix, these nonzero elements should be equal to 1. This completes the first part of the proof.

Now let us assume that the transition matrix $\mathbf{P}$ is all zeros except for $m$ elements that have 1's arranged such that each column and each row of $\mathbf{P}$ contains only a single 1 entry in a unique location.

The unique arrangement of 1's implies that $\mathbf{P}$ is column stochastic. The unique arrangement of 1's also implies that the determinant of $\mathbf{P}$ must be $\pm 1$. Thus the product of the eigenvalues of $\mathbf{P}$ is given by

$$\prod_i \lambda_i = \pm 1 \tag{6.44}$$

Since $\mathbf{P}$ was proven to be column stochastic, we have

$$|\lambda_i| \leq 1 \tag{6.45}$$

The above equation together with (6.44) implies that

$$|\lambda_i| = 1 \tag{6.46}$$

Thus we proved that the eigenvalues of $\mathbf{P}$ all lie on the unit circle of the complex plane when $\mathbf{P}$ has a unique distribution of 1's among its rows and columns. According to Theorem 6.4, this implies that the Markov chain is strongly periodic. This proves the theorem.

□

## 6.9 Canonic Form for P

A strongly periodic Markov chain will have its $m \times m$ transition matrix expressed in the canonic form

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \tag{6.47}$$

This matrix can be obtained by proper ordering of the states and will have a period $\gamma = m$ which can be easily proven (see Problem 6.5). This matrix is also known as

a *circulant matrix* since multiplying any matrix by this matrix will shift the rows or columns by one location.

## 6.10 Transition Diagram

Based on the above theorems specifying the structure of a strongly periodic Markov chain, we find that the transition diagram for a strongly periodic Markov chain of period $\gamma = 3$ is as shown in Fig. 6.5. We see that each set of periodic classes consists of one state only and the number of states equals the period of the Markov chain.

*Example 6.5.* Prove that the given transition matrix is periodic and determine the period.

$$\mathbf{P} = \begin{bmatrix} 0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1 \\ 1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0 \\ 0\,0\,0\,1\,0 \end{bmatrix}$$

The given matrix is column stochastic, full rank and all its rows and columns are all zeros except for five elements that contain 1 at unique locations in each row and column. Thus $\mathbf{P}$ is periodic according to Theorem 6.6. From MATLAB the eigenvalues of $\mathbf{P}$ are given by

$$\lambda_1 = \quad 1\angle 144^\circ = \quad \exp\left(j2\pi \times \frac{2}{5}\right)$$

$$\lambda_2 = 1\angle -144^\circ = \exp\left(j2\pi \times \frac{3}{5}\right)$$

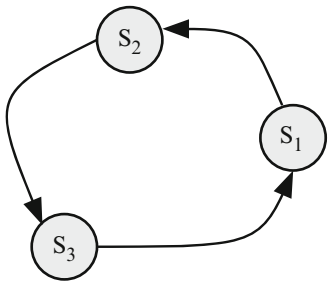



**Fig. 6.5** Transition diagram for a strongly periodic Markov chain with period $\gamma = 3$

$$\lambda_3 = \quad 1\angle 72° = \quad \exp\left(j2\pi \times \frac{1}{5}\right)$$

$$\lambda_4 = 1\angle -72° = \exp\left(j2\pi \times \frac{4}{5}\right)$$

$$\lambda_5 = \quad 1\angle 0° = \quad \exp\left(j2\pi \times \frac{5}{5}\right)$$

Thus the period is $\gamma = 5$. As a verification, MATLAB assures us that $\mathbf{P}^5 = \mathbf{I}$. ■

## 6.11 Composite Strongly Periodic Markov Chains

In general a composite strongly periodic Markov chain can be expressed, through proper ordering of states, in the canonic form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_2 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_3 & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{C}_{h-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{C}_h \end{bmatrix} \tag{6.48}$$

where $\mathbf{C}_i$ is an $m_i \times m_i$ circulant matrix whose period is $\gamma_i = m_i$. The period of the composite Markov chain is given by the equation

$$\gamma = \text{lcm}\,(\gamma_1,\ \gamma_2,\ \cdots,\ \gamma_h) \tag{6.49}$$

Figure 6.6 shows the periodic classes of a composite strongly periodic Markov chain. The states are divided into non-communicating sets of periodic classes.

*Example 6.6.* Find the period of the following periodic transition matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

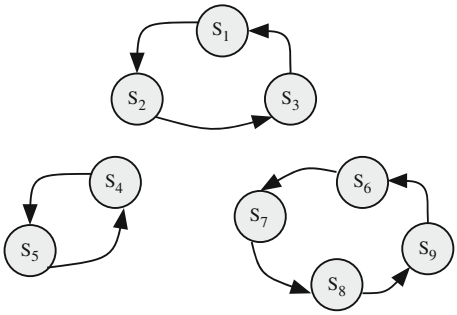



**Fig. 6.6** Transition diagram for a composite strongly periodic Markov chain. The states are divided into non-communicating sets of periodic classes

We identify $\mathbf{C}_1$ as a circulant $4 \times 4$ matrix with period $\gamma_1 = 4$ and we identify $\mathbf{C}_2$ as a circulant $3 \times 3$ matrix with period $\gamma_2 = 3$. Indeed, the eigenvalues of $\mathbf{P}$ are

$$\lambda_1 = \quad 1\angle 180^\circ = \quad \exp\left(j2\pi \times \frac{2}{4}\right)$$

$$\lambda_2 = \quad 1\angle 90^\circ = \quad \exp\left(j2\pi \times \frac{1}{4}\right)$$

$$\lambda_3 = 1\angle -90^\circ = \exp\left(j2\pi \times \frac{3}{4}\right)$$

$$\lambda_4 = \quad 1\angle 0^\circ = \quad \exp\left(j2\pi \times \frac{4}{4}\right)$$

$$\lambda_5 = \quad 1\angle 120^\circ = \quad \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_6 = 1\angle -120^\circ = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_7 = \quad 1\angle 0^\circ = \quad \exp\left(j2\pi \times \frac{3}{3}\right)$$

We expect $\mathbf{P}$ to have a period equal to the least common multiple of 3 and 4 which is 12. Indeed this is verified by MATLAB where the smallest power $k$ for which $\mathbf{P}^k = \mathbf{I}$ is when $k = 12$. ■

*Example 6.7.* Find the period of the following transition matrix for a strongly periodic Markov chain and express **P** in the form given by (6.48).

$$\mathbf{P} = \begin{bmatrix} 0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0 \\ 1&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0&0 \\ 0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \end{bmatrix}$$

By inspection, we know that the given matrix is periodic since the elements are either 1 or zero and each row or column has a single 1 at a unique location.

Indeed, the eigenvalues of **P** all lie on the unit circle

$$\lambda_1 = \quad 1\angle 120^\circ = \quad \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_2 = 1\angle -120^\circ = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_3 = \quad 1\angle 0^\circ = \quad \exp\left(j2\pi \times \frac{3}{3}\right)$$

$$\lambda_4 = \quad 1\angle 0^\circ = \quad \exp\left(j2\pi \times \frac{2}{2}\right)$$

$$\lambda_5 = \quad 1\angle 180^\circ = \quad \exp\left(j2\pi \times \frac{1}{2}\right)$$

$$\lambda_6 = \quad 1\angle 180^\circ = \quad \exp\left(j2\pi \times \frac{2}{4}\right)$$

$$\lambda_7 = \quad 1\angle 90^\circ = \quad \exp\left(j2\pi \times \frac{1}{4}\right)$$

$$\lambda_8 = 1\angle -90^\circ = \exp\left(j2\pi \times \frac{3}{4}\right)$$

$$\lambda_9 = 1\angle 0^\circ = \exp\left(j2\pi \times \frac{4}{4}\right)$$

Using MATLAB we find that $\mathbf{P}^3 \neq \mathbf{I}$, $\mathbf{P}^2 \neq \mathbf{I}$, and $\mathbf{P}^4 \neq \mathbf{I}$. However the LCM of 2, 3, and 4 is 12 and $\mathbf{P}^{12} = \mathbf{I}$.

We notice that there are three eigenvalues equal to 1 mainly:

$$\lambda_3 = \lambda_4 = \lambda_9 = 1$$

The eigenvectors corresponding to these eigenvalues give three sets of periodic classes:

$$C_1 = \{1,\ 3,\ 4\}$$
$$C_2 = \{2,\ 8\}$$
$$C_3 = \{5,\ 6,\ 7,\ 9\}$$

Each set is identified by the nonzero components of the corresponding eigenvector.

To group each set of periodic states together, we exchange states 2 and 4 so that $C_1$ will be contain the new states 1, 3, and 4. This is done using the elementary exchange matrix

$$\mathbf{E}(2,4) = \begin{bmatrix} 1&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&1 \end{bmatrix}$$

The new matrix will be obtained with the operations

$$\mathbf{P}' = \mathbf{E}(2,4) \times \mathbf{P} \times \mathbf{E}(2,4)$$

Next, we group the states of $C_2$ together by exchanging states 5 and 8 so that $C_2$ will contain the new states 2 and 8 and $C_3$ will contain the states 5, 6, 7, and 9. The rearranged matrix will be

$$\mathbf{P} = \begin{bmatrix} 0&1&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&1&0 \end{bmatrix}$$

From the structure of the matrix, we see that the periods of the three diagonal circulant matrices are $\gamma_1 = 4$, $\gamma_2 = 2$, and $\gamma_3 = 3$. The period of the matrix is $\gamma = \text{lcm}(4, 2, 3) = 12$. As a verification, we find that the first time $\mathbf{P}^n$ becomes the identity matrix is when $n = 12$. ■

## 6.12 Weakly Periodic Markov Chains

Periodic behavior can sometimes be observed in Markov chains even when some of the eigenvalues of $\mathbf{P}$ lie on the unit circle while other eigenvalues lie inside the unit circle. In spite of that, periodic behavior is observed because the structure of the Matrix is closely related to the canonic form for a periodic Markov chain in (6.47) on page 204. To generalize the structure of a circulant matrix we replace each "1" with a block matrix and obtain the *canonic* form for a weakly periodic Markov chain:

$$\mathbf{P} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \cdots & \mathbf{0} & \mathbf{0} & \mathbf{W}_h \\ \mathbf{W}_1 & \mathbf{0} & \mathbf{0} \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_2 & \mathbf{0} \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots \ddots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \cdots & \mathbf{W}_{h-2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \cdots & \mathbf{0} & \mathbf{W}_{h-1} & \mathbf{0} \end{bmatrix} \tag{6.50}$$

where the block-diagonal matrices are square zero matrices and the nonzero matrices $\mathbf{W}_i$ could be rectangular but the sum of each of their columns is unity since $\mathbf{P}$ is column stochastic. Such a matrix will exhibit periodic behavior with a period $\gamma = h$ where $h$ is the number of $\mathbf{W}$ blocks.

As an example consider the following transition matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0.1 & 0.6 & 0.5 \\ 0 & 0 & 0.9 & 0.4 & 0.5 \\ 0.5 & 0.2 & 0 & 0 & 0 \\ 0.1 & 0.4 & 0 & 0 & 0 \\ 0.4 & 0.4 & 0 & 0 & 0 \end{bmatrix} \equiv \begin{bmatrix} \mathbf{0} & \mathbf{W}_2 \\ \mathbf{W}_1 & \mathbf{0} \end{bmatrix} \tag{6.51}$$

We know this matrix does not correspond to a strongly periodic Markov chain because it is not a 0–1 matrix whose elements are 0 or 1. However, let us look at the eigenvalues of this matrix:

$$\begin{aligned} \lambda_1 &= -1 \\ \lambda_2 &= \phantom{-}1 \\ \lambda_3 &= \phantom{-}0.3873j \\ \lambda_4 &= -0.3873j \\ \lambda_5 &= \phantom{-}0 \end{aligned}$$

Some of the eigenvalues are inside the unit circle and represent decaying modes, but two eigenvalues lie on the unit circle. It is this extra eigenvalue on the unit circle that is responsible for the periodic behavior.

Let us now see the long-term behavior of the matrix. When $n > 25$ the contribution of the decaying modes will be $< 10^{-10}$ so let us see how $\mathbf{P}^n$ behaves when $n > 25$.

$$\mathbf{P}^{25} = \begin{bmatrix} 0 & 0 & 0.4 & 0.4 & 0.4 \\ 0 & 0 & 0.6 & 0.6 & 0.6 \\ 0.32 & 0.32 & 0 & 0 & 0 \\ 0.28 & 0.28 & 0 & 0 & 0 \\ 0.40 & 0.40 & 0 & 0 & 0 \end{bmatrix} \tag{6.52}$$

$$\mathbf{P}^{26} = \begin{bmatrix} 0.28 & 0.28 & 0 & 0 & 0 \\ 0.40 & 0.40 & 0 & 0 & 0 \\ 0 & 0 & 0.4 & 0.4 & 0.4 \\ 0 & 0 & 0.6 & 0.6 & 0.6 \\ 0.32 & 0.32 & 0 & 0 & 0 \end{bmatrix} \tag{6.53}$$

$$\mathbf{P}^{27} = \begin{bmatrix} 0 & 0 & 0.4 & 0.4 & 0.4 \\ 0 & 0 & 0.6 & 0.6 & 0.6 \\ 0.32 & 0.32 & 0 & 0 & 0 \\ 0.28 & 0.28 & 0 & 0 & 0 \\ 0.40 & 0.40 & 0 & 0 & 0 \end{bmatrix} \tag{6.54}$$

We see that $\mathbf{P}^n$ repeats its structure every two iterations.

We can make several observations about this transition matrix:

1. The transition matrix displays periodic behavior for large value of $n$.
2. $\mathbf{P}^n$ has a block structure that does not disappear. The blocks just move vertically at different places after each iteration.
3. The columns of each block in $\mathbf{P}^n$ are identical and the distribution vector will be independent of its initial value.
4. The period of $\mathbf{P}^n$ is 2.

Let us see now the distribution vector values for $n = 25$, 26, and 27. The initial value of the distribution vector is not important and we arbitrarily pick

$$\mathbf{s}(0) = \begin{bmatrix} 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \end{bmatrix}^t \tag{6.55}$$

We get

$$\mathbf{s}(25) = \mathbf{P}^{25}\mathbf{s}(0) = \begin{bmatrix} 0.240 & 0.360 & 0.128 & 0.112 & 0.160 \end{bmatrix}^t \tag{6.56}$$

$$\mathbf{s}(26) = \begin{bmatrix} 0.160 & 0.240 & 0.192 & 0.168 & 0.240 \end{bmatrix}^t \tag{6.57}$$

$$\mathbf{s}(27) = \begin{bmatrix} 0.240 & 0.360 & 0.128 & 0.112 & 0.160 \end{bmatrix}^t \tag{6.58}$$

We notice that the distribution vector repeats its value every two iterations. Specifically we see that $\mathbf{s}(25) = \mathbf{s}(27)$ and so on.

*Example 6.8.* The following transition matrix can be expressed in the form of (6.50). Find this form and estimate the period of the Markov chain.

$$\mathbf{P} = \begin{bmatrix} 0 & 0.5 & 0.1 & 0 & 0 & 0 \\ 0 & 0.3 & 0.5 & 0 & 0 & 0 \\ 0 & 0.2 & 0.4 & 0 & 0 & 0 \\ 0.9 & 0 & 0 & 0 & 0 & 0 \\ 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Our strategy for rearranging the states is to make the diagonal zero matrices appear in ascending order. The following ordering of states gives the desired result:

$$1 \leftrightarrow 6 \; 2 \leftrightarrow 5 \; 3 \leftrightarrow 4$$

The rearranged matrix will be

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0.9 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 0.4 & 0 & 0 & 0 \\ 0 & 0.3 & 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0.1 & 0 & 0 & 0 \end{bmatrix}$$

The structure of the matrix is now seen to be in the form

$$\mathbf{P} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{W}_3 \\ \mathbf{W}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_2 & \mathbf{0} \end{bmatrix}$$

where

$$\mathbf{W}_1 = \begin{bmatrix} 0.1 \\ 0.9 \end{bmatrix}$$

$$\mathbf{W}_2 = \begin{bmatrix} 0.2 & 0.4 \\ 0.3 & 0.5 \\ 0.5 & 0.1 \end{bmatrix}$$

$$\mathbf{W}_3 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

The eigenvalues for $\mathbf{P}$ are

$$\lambda_1 = \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_2 = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_3 = 1 = \exp\left(j2\pi \times \frac{3}{3}\right)$$

$$\lambda_4 = 0$$

$$\lambda_5 = 0$$

$$\lambda_6 = 0$$

The period of $\mathbf{P}$ is 3. As a verification, we chose

$$\mathbf{s}(0) = \begin{bmatrix} 0.2 & 0.2 & 0.2 & 0.2 & 0.2 & 0 \end{bmatrix}^t$$

and found the following distribution vectors

$$\mathbf{s}(3) = \begin{bmatrix} 0.2000 & 0.0400 & 0.3600 & 0.1520 & 0.1920 & 0.0560 \end{bmatrix}^t$$

$$\mathbf{s}(4) = \begin{bmatrix} 0.4000 & 0.0200 & 0.1800 & 0.1520 & 0.1920 & 0.0560 \end{bmatrix}^t$$

$$\mathbf{s}(5) = \begin{bmatrix} 0.4000 & 0.0400 & 0.3600 & 0.0760 & 0.0960 & 0.0280 \end{bmatrix}^t$$

$$\mathbf{s}(6) = \begin{bmatrix} 0.2000 & 0.0400 & 0.3600 & 0.1520 & 0.1920 & 0.0560 \end{bmatrix}^t$$

$$\mathbf{s}(7) = \begin{bmatrix} 0.4000 & 0.0200 & 0.1800 & 0.1520 & 0.1920 & 0.0560 \end{bmatrix}^t$$

We see that the distribution vector repeats itself over a period of three iterations. Specifically we see that $\mathbf{s}(3) = \mathbf{s}(6)$ and $\mathbf{s}(4) = \mathbf{s}(7)$ and so on. ■

*Example 6.9.* The following transition matrix has the form of (6.50). Estimate the period of the Markov chain and study the distribution vector after the transients have decayed.

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0.3 & 1 \\ 0 & 0 & 0 & 0 & 0.7 & 0 \\ 0.9 & 0.6 & 0 & 0 & 0 & 0 \\ 0.1 & 0.4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.9 & 0 & 0 \\ 0 & 0 & 0.5 & 0.1 & 0 & 0 \end{bmatrix}$$

The eigenvalues for this matrix are

$$\lambda_1 = \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_2 = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_3 = 1 = \exp\left(j2\pi \times \frac{3}{3}\right)$$

$$\lambda_4 = 0.438 \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_5 = 0.438 \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_6 = 0.438 \exp\left(j2\pi \times \frac{3}{3}\right)$$

The period of this matrix is $\gamma = 3$.

The transients would die away after about 30 iterations since the value of the eigenvalues within the unit circle would be

$$\lambda^{30} = 0.438^{30} = 1.879 \times 10^{-11}$$

The value of $\mathbf{P}^n$ at high values for $n$ would start to approach an equilibrium pattern

$$\mathbf{P}^n = \begin{bmatrix} 0.5873 & 0.5873 & 0 & 0 & 0 & 1 \\ 0.4127 & 0.4127 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.7762 & 0.7762 & 0 & 0 \\ 0 & 0 & 0.2238 & 0.2238 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5895 & 0.5895 \\ 0 & 0 & 0 & 0 & 0.4105 & 0.4105 \end{bmatrix}$$

where $n > 30$. As a verification, we chose

$$\mathbf{s}(0) = \begin{bmatrix} 0.2 & 0.2 & 0.2 & 0.2 & 0.2 & 0 \end{bmatrix}^t$$

and found the following distribution vectors

$$\mathbf{s}(30) = \begin{bmatrix} 0.2349 & 0.1651 & 0.3105 & 0.0895 & 0.1179 & 0.0821 \end{bmatrix}^t$$

$$\mathbf{s}(31) = \begin{bmatrix} 0.1175 & 0.0825 & 0.3105 & 0.0895 & 0.2358 & 0.1642 \end{bmatrix}^t$$

$$\mathbf{s}(32) = \begin{bmatrix} 0.2349 & 0.1651 & 0.1552 & 0.0448 & 0.2358 & 0.1642 \end{bmatrix}^t$$

$$\mathbf{s}(33) = \begin{bmatrix} 0.2349 & 0.1651 & 0.3105 & 0.0895 & 0.1179 & 0.0821 \end{bmatrix}^t$$

$$\mathbf{s}(34) = \begin{bmatrix} 0.1175 & 0.0825 & 0.3105 & 0.0895 & 0.2358 & 0.1642 \end{bmatrix}^t$$

We see that the distribution vector repeats itself with a period of three iterations. Specifically we see that $\mathbf{s}(30) = \mathbf{s}(33)$ and $\mathbf{s}(31) = \mathbf{s}(34)$ and so on. ■

*Example 6.10.* Assume a wireless packet transmission system that employs an adaptive forward error correction scheme. There are two levels of data forward error correction that could be employed depending on the number of errors detected in the received packet as shown in Fig. 6.7. The upper row of states corresponds to lower error levels and the lower row of states corresponds to higher error levels.

For example, if the coder is in state $s_1$ and errors occur during transmission, we move to state $s_4$. If the receiver is able to correct the errors, we move to state $s_8$. We then conclude that the error correction coding is adequate and we move to state $s_1$ for the next transmission. If the errors cannot be corrected, we conclude that the error coding is not adequate and we move from state $s_9$ to $s_2$ at the next transmission.

Write down the transition matrix and show that it corresponds to a weakly periodic Markov chain.

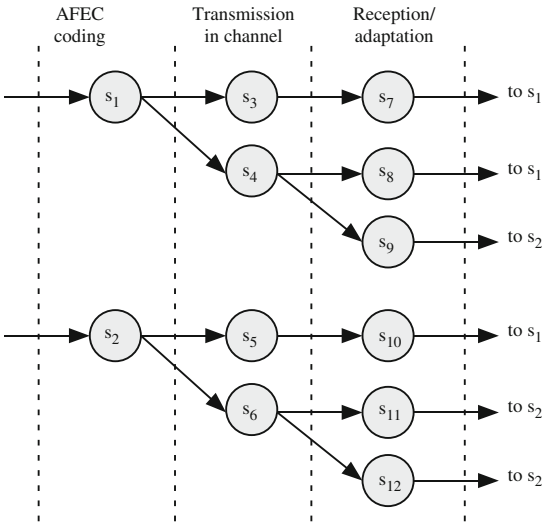



**Fig. 6.7** An adaptive forward error correction scheme that uses two levels of encoding depending on the number of errors introduced in the channel

The sets of periodic states are identified at the bottom of the figure. We can see that we have three sets of states such that the states in each set make transitions only to the next set. This seems to imply a weakly periodic Markov chain. As a verification, we construct the transition matrix and see its structure.

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 0 & p_{15} & p_{16} \\ 0 & 0 & 0 & 0 & p_{25} & p_{26} \\ p_{31} & p_{32} & 0 & 0 & 0 & 0 \\ p_{41} & p_{42} & 0 & 0 & 0 & 0 \\ 0 & 0 & p_{53} & p_{54} & 0 & 0 \\ 0 & 0 & p_{63} & p_{64} & 0 & 0 \end{bmatrix}$$

We see that the transition matrix has the same structure as a weakly periodic Markov chain with period $\gamma = 3$. The exact values of the transition probabilities will depend on the details of the system being investigated. ■

## 6.13 Reducible Periodic Markov Chains

A reducible periodic Markov chain is one in which the transition matrix can be partitioned into the canonic form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{6.59}$$

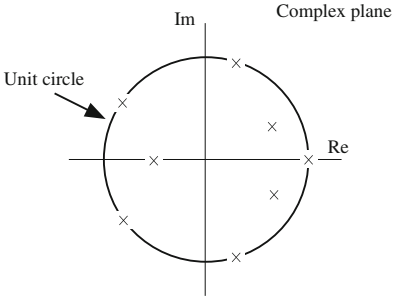



**Fig. 6.8** The eigenvalues of a reducible periodic Markov chain. Some of the eigenvalues lie on the unit circle in the complex plane and some lie inside the unit circle

where

$$\mathbf{C} = \text{square column stochastic periodic matrix}$$

$$\mathbf{A} = \text{rectangular nonnegative matrix}$$

$$\mathbf{T} = \text{square column substochastic matrix}$$

Some of the eigenvalues of the transition matrix will lie on the unit circle. The other eigenvalues will be inside the unit circle as shown in Fig. 6.8. Note that the periodic matrix **C** could be strongly periodic or could be weakly periodic.

*Example 6.11.* Check to see if the given matrix below corresponds to a reducible periodic Markov chain.

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0.1 & 0 & 0.1 & 1 \\ 0 & 0.3 & 0 & 0.2 & 0 \\ 1 & 0.2 & 0 & 0.2 & 0 \\ 0 & 0.3 & 0 & 0.4 & 0 \\ 0 & 0.1 & 1 & 0.1 & 0 \end{bmatrix} \end{array}$$

where the state indices are indicated around **P** for illustration. Rearrange the rows and columns to express the matrix in the form of (6.59) and identify the matrices **C**, **A**, and **T**. Verify the assertions that **C** is column stochastic, **A** is nonnegative, and **T** is column substochastic.

The best way to study a Markov chain is to explore its eigenvalues.

$$\lambda_1 = \exp\left(j2\pi \times \frac{1}{3}\right)$$

$$\lambda_2 = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_3 = \exp\left(j2\pi \times \frac{3}{3}\right)$$
$$\lambda_4 = 0.6\exp(j2\pi)$$
$$\lambda_5 = 0.1\exp(j2\pi)$$

Thus we see we have two decaying modes but three other eigenvalues lie on the unit circle. We classify this system as a weakly periodic Markov chain.

The vector corresponding to the unity eigenvalue is given by

$$\mathbf{x} = \begin{bmatrix} 0.5774 & 0 & 0.5774 & 0 & 0.5774 \end{bmatrix}^t$$

The zero components of the eigenvector indicate that we have transient states, namely $s_2$ and $s_4$. The fact that we have only one eigenvalue that is unity indicates that we have one set only of closed states: $C = 1,\ 3,\ ,5$. Based on that we further classify this system as reducible weakly periodic Markov chain.

We cluster states 1, 3 and 5 together since they correspond to the closed states and cluster states 2 and 4 together since they correspond to the transient states. We perform this rearranging through the elementary exchange matrix $\mathbf{E}(2,5)$ which exchanges states 2 and 5:

$$\mathbf{E}(2,5) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

The exchange of states is achieved by pre and post multiplying the transition matrix:

$$\mathbf{P}' = \mathbf{E}(2,5)\ \mathbf{P}\ \mathbf{E}(2,5)$$

This results in

$$\mathbf{P}' = \begin{array}{c} \\ 1 \\ 5 \\ 3 \\ 4 \\ 2 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 5 & 3 & 4 & 2 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0.1 & 0.1 \\ 0 & 0 & 1 & 0.1 & 0.1 \\ 1 & 0 & 0 & 0.2 & 0.2 \\ 0 & 0 & 0 & 0.4 & 0.1 \\ 0 & 0 & 0 & 0.2 & 0.3 \end{bmatrix} \end{array}$$

We see that the transition matrix represents a reducible periodic Markov chain and matrices **C**, **A**, and **T** are

$$\mathbf{C} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0.1 & 0.1 \\ 0.1 & 0.1 \\ 0.2 & 0.2 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.5 & 0.1 \\ 0.2 & 0.3 \end{bmatrix}$$

The sum of each column of $\mathbf{C}$ is exactly 1, which indicates that it is column stochastic and strongly periodic. The sum of columns of $\mathbf{T}$ is less than 1, which indicates that it is column substochastic.

The set of closed periodic states is $C = \{1, 3, 5\}$ and the set of transient states is $T = \{2, 4\}$

Starting in state 2 or 4, we will ultimately go to states 1, 3, or 5. Once we are there, we cannot ever go back to state 2 or 4 because we entered the closed periodic states. ■

## 6.14 Transient Analysis

After $n$ time steps the transition matrix of a reducible Markov chain will still be reducible and will have the form

$$\mathbf{P}^n = \begin{bmatrix} \mathbf{C}^n & \mathbf{Y}^n \\ \mathbf{0} & \mathbf{T}^n \end{bmatrix} \tag{6.60}$$

where matrix $\mathbf{Y}^n$ is given by

$$\mathbf{Y}^n = \sum_{i=0}^{n-1} \mathbf{C}^{n-i-1}\,\mathbf{A}\,\mathbf{T}^i \tag{6.61}$$

We can always find $\mathbf{C}^n$ and $\mathbf{T}^n$ using the techniques discussed in Chap. 3 such as diagonalization, finding the Jordan canonic form, or even repeated multiplications.

The stochastic matrix $\mathbf{C}^n$ can be expressed in terms of its eigenvalues using (3.80) on page 96.

$$\mathbf{C}^n = \mathbf{C}_1 + \lambda_2^n \mathbf{C}_2 + \lambda_3^n \mathbf{C}_3 + \cdots \tag{6.62}$$

where it was assumed that $\mathbf{C}_1$ is the expansion matrix corresponding to the eigenvalue $\lambda_1 = 1$.

Similarly, the substochastic matrix $\mathbf{T}^n$ can be expressed in terms of its eigenvalues using (3.80) on page 96.

$$\mathbf{T}^n = \lambda_1^n \mathbf{T}_1 + \lambda_2^n \mathbf{T}_2 + \lambda_3^n \mathbf{T}_3 + \cdots \tag{6.63}$$

Equation (6.61) can then be expressed in the form

$$\mathbf{Y}^n = \sum_{j=1}^{m} \mathbf{C}_j \ \mathbf{A} \ \sum_{i=0}^{n-1} \lambda_j^{n-i-1} \mathbf{T}^i \tag{6.64}$$

After some algebraic manipulations, we arrive at the form

$$\mathbf{Y}^n = \sum_{j=1}^{m} \lambda_j^{n-1} \mathbf{C}_j \ \mathbf{A} \left[ \mathbf{I} - \left( \frac{\mathbf{T}}{\lambda_j} \right)^n \right] \left( \mathbf{I} - \frac{\mathbf{T}}{\lambda_j} \right)^{-1} \tag{6.65}$$

This can be written in the form

$$\begin{aligned} \mathbf{Y}^n = {} & \mathbf{C}_1 \mathbf{A} \left( \mathbf{I} - \mathbf{T} \right)^{-1} \left[ \mathbf{I} - \mathbf{T}^n \right] + \\ & \lambda_2^{n-1} \mathbf{C}_2 \mathbf{A} \left( \mathbf{I} - \frac{1}{\lambda_2} \mathbf{T} \right)^{-1} \left[ \mathbf{I} - \frac{1}{\lambda_2^n} \mathbf{T}^n \right] + \\ & \lambda_3^{n-1} \mathbf{C}_3 \mathbf{A} \left( \mathbf{I} - \frac{1}{\lambda_3} \mathbf{T} \right)^{-1} \left[ \mathbf{I} - \frac{1}{\lambda_3^n} \mathbf{T}^n \right] + \cdots \end{aligned} \tag{6.66}$$

If some of the eigenvalues of $\mathbf{C}$ are repeated, then the above formula has to be modified as explained in Sect. 3.14 on page 105.

*Example 6.12.* Consider the reducible weakly periodic Markov chain of the previous example. Assume that the system was initially in state 4. Explore how the distribution vector changes as time progresses.

We do not have to rearrange the transition matrix to do this example. We have

$$\mathbf{P} = \begin{bmatrix} 0 & 0.1 & 0 & 0.1 & 1 \\ 0 & 0.3 & 0 & 0.2 & 0 \\ 1 & 0.2 & 0 & 0.2 & 0 \\ 0 & 0.3 & 0 & 0.4 & 0 \\ 0 & 0.1 & 1 & 0.1 & 0 \end{bmatrix}$$

The eigenvalues for this matrix are

$$\lambda_1 = \exp\left( j 2\pi \times \frac{1}{3} \right)$$

$$\lambda_2 = \exp\left(j2\pi \times \frac{2}{3}\right)$$

$$\lambda_3 = \exp\left(j2\pi \times \frac{3}{3}\right)$$

$$\lambda_4 = 0.1\exp\left(j2\pi \times \frac{3}{3}\right)$$

We see that the period of this system is $\gamma = 3$. The initial distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}^t$$

The distribution vector at the start is given by

$$\mathbf{s}(0) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}^t$$

$$\mathbf{s}(1) = \begin{bmatrix} 0.1000 & 0.2000 & 0.2000 & 0.4000 & 0.1000 \end{bmatrix}^t$$

$$\mathbf{s}(2) = \begin{bmatrix} 0.1600 & 0.1400 & 0.2200 & 0.2200 & 0.2600 \end{bmatrix}^t$$

$$\mathbf{s}(3) = \begin{bmatrix} 0.2960 & 0.0860 & 0.2320 & 0.1300 & 0.2560 \end{bmatrix}^t$$

$$\mathbf{s}(4) = \begin{bmatrix} 0.2776 & 0.0518 & 0.3392 & 0.0778 & 0.2536 \end{bmatrix}^t$$

$$\mathbf{s}(5) = \begin{bmatrix} 0.2666 & 0.0311 & 0.3035 & 0.0467 & 0.3522 \end{bmatrix}^t$$

Continuing the iterations, the distribution vector settles down to the following sequence.

$$\mathbf{s}(19) = \begin{bmatrix} 0.3265 & 0.0000 & 0.3775 & 0.0000 & 0.2959 \end{bmatrix}^t$$

$$\mathbf{s}(20) = \begin{bmatrix} 0.2959 & 0.0000 & 0.3265 & 0.0000 & 0.3775 \end{bmatrix}^t$$

$$\mathbf{s}(21) = \begin{bmatrix} 0.3775 & 0.0000 & 0.2959 & 0.0000 & 0.3265 \end{bmatrix}^t$$

$$\mathbf{s}(22) = \begin{bmatrix} 0.3265 & 0.0000 & 0.3775 & 0.0000 & 0.2959 \end{bmatrix}^t$$

$$\mathbf{s}(23) = \begin{bmatrix} 0.2959 & 0.0000 & 0.3265 & 0.0000 & 0.3775 \end{bmatrix}^t$$

$$\mathbf{s}(24) = \begin{bmatrix} 0.3775 & 0.0000 & 0.2959 & 0.0000 & 0.3265 \end{bmatrix}^t$$

We note that after about 20 time steps, the probability of being in the transient states 2 or 4 is nil. The system will definitely be in the closed set composed of states 1, 3, or 5. The distribution vector will show periodic behavior with a period $\gamma = 3$. ■

## 6.15 Asymptotic Behavior

Assume we have a reducible periodic Markov chain with transition matrix $\mathbf{P}$ that is expressed in the form

$$\mathbf{P} = \begin{bmatrix} \mathbf{C} & \mathbf{A} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \tag{6.67}$$

According to (6.60), after $n$ time steps the transition matrix will have the form

$$\mathbf{P}^n = \begin{bmatrix} \mathbf{C}^n & \mathbf{Y}^n \\ \mathbf{0} & \mathbf{T}^n \end{bmatrix} \tag{6.68}$$

where matrix $\mathbf{Y}^n$ is given by (6.65) in the form

$$\mathbf{Y}^n = \sum_{j=1}^{m} \lambda_j^{n-1} \mathbf{C}_j \; \mathbf{A} \left[ \mathbf{I} - \left( \frac{\mathbf{T}}{\lambda_j} \right)^n \right] \left( \mathbf{I} - \frac{\mathbf{T}}{\lambda_j} \right)^{-1} \tag{6.69}$$

when $n \to \infty$, $\mathbf{T}^n$ will become zero since it is column substochastic. Furthermore, the eigenvalues of $\mathbf{C}$ satisfy the following equations because of the periodicity of $\mathbf{C}$.

$$|\lambda_i| = 1 \qquad 1 \leq i \leq K \tag{6.70}$$

$$|\lambda_i| < 1 \qquad K < i \leq m \tag{6.71}$$

The eigenvalues that lie in the unit circle will have no contribution at large values of $n$ and matrix $\mathbf{P}^\infty$ becomes

$$\mathbf{P}^\infty = \begin{bmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{6.72}$$

The matrices $\mathbf{E}$ and $\mathbf{F}$ are given by

$$\mathbf{E} = \sum_{k=1}^{K} \lambda_k^i \mathbf{C}_k \tag{6.73}$$

$$\mathbf{F} = \left[ \sum_{i=1}^{\gamma} \sum_{k=1}^{K} \lambda_k^{i-1} \mathbf{C}_k \; \mathbf{A} \; \mathbf{T}^{i-1} \right] (\mathbf{I} - \mathbf{T}^\gamma)^{-1} \tag{6.74}$$

where $\mathbf{I}$ is the unit matrix whose dimensions match that of $\mathbf{T}$.

*Example 6.13.* Find the asymptotic transition matrix for the reducible weakly periodic Markov chain characterized by the transition matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0.9 & 0.1 & 0.1 & 0.3 & 0.1 \\ 0 & 0 & 0.1 & 0.9 & 0.2 & 0.1 & 0.3 \\ 0.2 & 0.8 & 0 & 0 & 0 & 0.1 & 0 \\ 0.8 & 0.2 & 0 & 0 & 0.1 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0.2 & 0.2 & 0.1 \\ 0 & 0 & 0 & 0 & 0.1 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0.3 & 0.2 & 0.4 \end{bmatrix}$$

The components of the transition matrix are

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0.9 & 0.1 \\ 0 & 0 & 0.1 & 0.9 \\ 0.2 & 0.8 & 0 & 0 \\ 0.8 & 0.2 & 0 & 0 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0.1 & 0.3 & 0.1 \\ 0.2 & 0.1 & 0.3 \\ 0 & 0.1 & 0 \\ 0.1 & 0 & 0.1 \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} 0.2 & 0.2 & 0.1 \\ 0.1 & 0.1 & 0 \\ 0.3 & 0.2 & 0.4 \end{bmatrix}$$

**C** is a weakly periodic Markov chain whose eigenvalues are

$$\begin{aligned} \lambda_1 &= \quad 1 \\ \lambda_2 &= -1 \\ \lambda_3 &= \quad 0.6928j \\ \lambda_3 &= -0.6928j \end{aligned}$$

and $\mathbf{C}^i$ can be decomposed into the form

$$\mathbf{C}^i = \mathbf{C}_1 + (-1)^i \mathbf{C}_2$$

where

$$\mathbf{C}_1 = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix}$$

$$\mathbf{C}_2 = \begin{bmatrix} 0.25 & 0.25 & -0.25 & -0.25 \\ 0.25 & 0.25 & -0.25 & -0.25 \\ -0.25 & -0.25 & 0.25 & 0.25 \\ -0.25 & -0.25 & 0.25 & 0.25 \end{bmatrix}$$

According to (6.72) the matrix $\mathbf{E}$ has the value

$$\mathbf{E} = \begin{bmatrix} 0.2742 & 0.3044 & 0.3018 \\ 0.2742 & 0.3044 & 0.3018 \\ 0.2258 & 0.1956 & 0.1982 \\ 0.2258 & 0.1956 & 0.1982 \end{bmatrix}$$

According to (6.72) the matrix $\mathbf{F}$ has the value

$$\mathbf{F} = \begin{bmatrix} 0.2742 & 0.3044 & 0.3018 \\ 0.2742 & 0.3044 & 0.3018 \\ 0.2258 & 0.1956 & 0.1982 \\ 0.2258 & 0.1956 & 0.1982 \end{bmatrix}$$

Thus the asymptotic value of $\mathbf{P}$ is

$$\mathbf{P}^{\infty} = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0.2258 & 0.1956 & 0.1982 \\ 0.5 & 0.5 & 0 & 0 & 0.2258 & 0.1956 & 0.1982 \\ 0 & 0 & 0.5 & 0.5 & 0.2742 & 0.3044 & 0.3018 \\ 0 & 0 & 0.5 & 0 & 0.2742 & 0.3044 & 0.3018 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We can make several observations on the asymptotic value of the transition matrix.

(a) The columns are not identical as in nonperiodic Markov chains.
(b) There is no possibility of moving to a transient state irrespective of the value of the initial distribution vector.
(c) The closed states are divided into sets such that the system makes periodic transitions between them and a steady state value can never be reached. ■

## 6.16 Identification of Markov Chains

We are now able to discuss how we can determine if the given transition matrix corresponds to a periodic Markov chain or not and to determine if the chain is strongly or weakly periodic. Further, we are also able to tell if the Markov chain is reducible or irreducible and to identify the transient and closed states.

Inspection of the elements of the transition matrix does not help much except for the simplest case when we are dealing with a strongly periodic Markov chain. In that case the transition matrix will be a 0–1 matrix. However, determining the period of the chain is not easy since we have to rearrange the matrix to correspond to the form (6.48) on page 206.

A faster and more direct way to classify a Markov chain is to simply study its eigenvalues and eigenvector corresponding to $\lambda = 1$. The following subsections summarize the different cases that can be encountered.

### 6.16.1 Nonperiodic Markov Chain

This is the case when only one eigenvalue is 1 and all other eigenvalues lie inside the unit circle:

$$|\lambda_i| < 1 \tag{6.75}$$

For large values of the time index $n \to \infty$ all modes will decay except the one corresponding to $\lambda_1 = 1$ which gives us the steady state distribution vector.

### 6.16.2 Strongly Periodic Markov Chain

This is the case when all the eigenvalues of the transition matrix lie on the unit circle:

$$\lambda_i = \exp\left(j2\pi \times \frac{i}{\gamma}\right) \tag{6.76}$$

where $1 \leq i \leq \gamma$.

For all values of the time index the distribution vector will exhibit periodic behavior and the period of the system is $\gamma$.

### 6.16.3 Weakly Periodic Markov Chain

This is the case when $\gamma$ eigenvalues of the transition matrix lie on the unit circle and the rest of the eigenvalues lie inside the unit circle. Thus we can write

$$|\lambda_i| = 1 \qquad \text{when } 1 \leq i \leq \gamma \tag{6.77}$$

$$|\lambda_i| < 1 \qquad \text{when } \gamma < i \leq m \tag{6.78}$$

The eigenvalues that lie on the unit circle will be given by

$$\lambda_i = \exp\left(j2\pi \times \frac{i}{\gamma}\right) \tag{6.79}$$

where $1 \leq i \leq \gamma$.

For large values of the time index $n \to \infty$ some of the modes will decay but $\gamma$ of them will not and the distribution vector will never settle down to a stable value. The period of the system is $\gamma$.

## 6.17 Problems

### *Strongly Periodic Markov Chains*

**6.1.** The following matrix is a circulant matrix of order 3, we denote it by $\mathbf{C}_3$. If it corresponds to a transition matrix, then the resulting Markov chain is strongly periodic. Find the period of the Markov chain.

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

**6.2.** Prove Theorem 6.5 using the result of Theorem 6.1 and the fact that all eigenvalues of a column stochastic matrix are in the range $0 \leq |\lambda| \leq 1$.

**6.3.** Does the following transition matrix represent a strongly periodic Markov chain?

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

**6.4.** Verify that the following transition matrix corresponds to a periodic Markov chain and find the period. Find also the values of all the eigenvalues.

$$\mathbf{P} = \begin{bmatrix} 0\,0\,0\,0\,0\,0\,1 \\ 0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0 \\ 0\,0\,1\,0\,0\,0\,0 \\ 1\,0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0 \end{bmatrix}$$

**6.5.** Prove that the $m \times m$ circulant matrix in (6.47) has a period $\gamma = m$. You can do that by observing the effect of premultiplying any $m \times k$ matrix by this matrix. From that you will be able to find a pattern for the repeated multiplication of the transition matrix by itself.

## *Weakly Periodic Markov Chains*

**6.6.** Does the following transition matrix represent a periodic Markov chain?

$$\mathbf{P} = \begin{bmatrix} 1/3 & 0 & 0 & 0 & 0 & 1/3 & 1/3 \\ 1/3 & 1/3 & 0 & 0 & 0 & 0 & 1/3 \\ 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 \end{bmatrix}$$

**6.7.** The weather in a certain island in the middle of nowhere is either sunny or rainy. A recent shipwreck survivor found that if the day is sunny, then the next day will be sunny with 80 % probability. If the day is rainy, then the next day it will be rainy with 70 % probability. Find the asymptotic behavior of the weather on this island.

**6.8.** A traffic source is modeled as a periodic Markov chain with three stages. Each stage has three states: silent, transmitting at rate $\lambda_1$, and transmitting at rate $\lambda_2$.

(a) Draw the periodic transition diagram.
(b) Write down the transition matrix.
(c) Assign transition probabilities to the system and find the asymptotic behavior of the source.
(d) Plot the state of the system versus time by choosing the state with the highest probability at each time instant. Can you see any periodic behavior in the traffic pattern?

**6.9.** A computer goes through the familiar fetch, decode and execute stages for instruction execution. Assume the fetch stage has three states depending on the location of the operands in cache, RAM, or in main memory. The decode stage has three states depending on the instruction type. The execute state has three states also depending on the length of the instruction.

(a) Draw the periodic transition diagram.
(b) Write down the transition matrix.
(c) Assign transition probabilities to the system and find the asymptotic behavior of the program being run on the machine.

**6.10.** Table lookup for packet routing can be divided into three phases: database search, packet classification, and packet processing. Assume the database search phase consists of four states depending on the location of the data among the different storage modules. The packet classification phase consists of three states depending on the type of packet received. The packet processing phase consists of four different states depending on the nature of the processing being performed.

(a) Draw the periodic transition diagram.
(b) Write down the transition matrix.
(c) Assign transition probabilities to the system and find the asymptotic behavior of the lookup table operation.

**6.11.** A bus company did a study on commuter habits during a typical week. Essentially, the company wanted to know the percentage of commuters that use the bus, their own car, or stay at home each day of the week. Based on this study, the company can plan ahead and assign more busses or even bigger busses during heavy usage days.

(a) Draw the periodic transition diagram.
(b) Write down the transition matrix.
(c) Assign transition probabilities to the system and find the asymptotic behavior of the commuter patterns.

**6.12.** Assume a certain species of wild parrot have two colors: red and blue. If was found that when a red–red pair breeds, their offspring are red with a 90 % probability due to some obscure reasons related to recessed genes and so on. When a blue–blue pair breeds, their offspring are blue with a 70 % probability. When a red–blue pair breeds, their offspring are blue with a 50 % probability.

(a) Draw the periodic transition diagram.
(b) Write down the transition matrix.
(c) Assign transition probabilities to the system and find the asymptotic behavior of the parrot colors in the wild.

**6.13.** A packet source is modeled using periodic Markov chain. The source is assumed to go through four repeating phases and each phase has two states: idle and active. Transitions from phase 1 to phase 2 are such that the source switches to the other state with a probability 0.05. Transitions from phase 2 to phase 3 are

such that the source switches to the other state with a probability 0.1. Transitions from phase 3 to phase 4 are such that the source switches to the other state with a probability 0.2. Transitions from phase 4 to phase 1 are such that the source switches to the other state with a probability 0.90.

(a) Draw the periodic transition diagram.
(b) Write down the transition matrix.
(c) Assign transition probabilities to the system and find the asymptotic behavior of the source.

**6.14.** The predator and prey populations are related as was explained at the start of this chapter. Assume the states of the predator–prey population are as follows.

| State | Predator population | Prey population |
|---|---|---|
| $s_0$ | Low | Low |
| $s_1$ | Low | High |
| $s_2$ | High | Low |
| $s_3$ | High | High |

Assume that the populations undergo changes each year and the time step is chosen equal to 1 year also. The state of the system must change from one value to another value after each time step.

1. Construct state transition diagram and a Markov transition matrix.
2. Justify the entries you choose for the matrix.
3. Study the periodicity of the system.

## References

1. W.J. Stewart, *Introduction to Numerical Solutions of Markov Chains* (Princeton University Press, Princeton, 1994)
2. R. Horn, C.R. Johnson, *Matrix Analysis* (Cambridge University Press, Cambridge, 1985)
3. Private discussions with W.-S. Lu of the Department of Electronics and Computer Engineering University of Victoria

# Chapter 7
# Queuing Analysis

## 7.1 Introduction

Queuing analysis is one of the most important tools for studying communication systems. The analysis allows us to answer endless questions about the system performance. This chapter explains that queuing analysis is a special case of Markov chains. Some examples of queues are:

- The number of patients in a doctor's waiting room.
- The number of customers in a store checkout line.
- The number of packets stored in a router's buffer.
- The number of print jobs present in a printer's queue.
- The number of workstations requesting access to the LAN.

Without being specific to a certain system, we can state that *queuing analysis* deals with *queues* where *customers* compete to be processed by shared *servers*. The *queue size* is the waiting room provided for the customers that have not been served yet plus the customers that are being served. We will talk about packets instead of customers in this chapter since most of networking analysis deals with transmitting and processing packets.

The objective of queuing analysis is to predict the system performance such as how many customers get processed per time step, the average delay a customer endures before being served, and the size of the queue or waiting room required. These performance measures have obvious applications in telecommunication systems and the design of hardware for such systems.

We list here some typical examples of queues and point out the customers and servers in each.

- People lining up at a bank where the bank teller is the server and the bank patrons are of course the customers.
- Workstations connected in a local-area network (LAN) where the communication medium (e.g., Ethernet cable) represents the shared resource while the commu-



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_7

nicating applications represent the customers and the server is the media access control (MAC) protocol that enables access to the medium.
- A parallel processing system in which a common shared memory is accessed by all the computers. The computers, or rather the memory requests, represent the customers and the arbitration protocol that resolves memory access conflicts represents the server.

Most of the literature and textbooks deals with continuous-time systems. In these systems only single customer arrival or departure takes place at a given time instant. However, analyzing the systems for general arrival or departure statistics proved to be difficult such that most textbooks simply provide tables of performance formulas for the most common situations. Subsequent researchers simply adopt these formulas without any form of adaptation or innovation.

Most of computer and communication systems, however, are migrating to the digital domain. In this domain, time is measured in discrete units or steps of finite size. As a consequence, there could be multiple arrivals or departures at a given time step. We will find that discrete-time systems are simple to analyze compared to continuous-time systems. Adapting discrete-time systems to a large variety of situations is really simple using the techniques we provide in this book.

In this chapter we study different types of discrete-time queues characterized by the following attributes:

1. The total number of customers in the system.
2. The number of customers that could possibly request service at a given time step.
3. The arrival process statistics for the customers.
4. The number of servers which dictate how many customers can leave the queue at a given time step.
5. The service discipline for deciding which customer or customers is to be served. Examples of service disciplines are first-come-first-serve (FIFO), random selection, polling, priority, etc. [1].
6. The size of the queue to accommodate customers waiting for service.

## *Kendall's Notation*

Kendall's notation is frequently used to succinctly describe a queuing system. This notation is represented as $A/B/c/n/p$, where [2]:

$A$ : Arrival statistics
$B$ : Service or departure statistics
$c$ : Number of servers
$n$ : Queue size
$p$ : Customer population size

The final two fields are optional and are assumed infinite if they are omitted [3]. The letters $A$ and $B$ in the notation A/B/c/n/p denoting arrival and server statistics are given the following notations:

D : *Deterministic*, process has fixed arrival or service rates
M : *Markovian*, process is Poisson or binomial
G : *General* or constant time.

A common practice is to attach a superscript to the letters $A$ and $B$ in the notation A/B/c/n/p to denote multiple arrivals or "batch service." Using this notation, the discrete-time $M/M/1$ queue has binomial, or Poisson, arrivals and departures. At a given time step at most one customer arrives and at most one customer departs. The queue has infinite buffer size and the population size is also infinite.

The queue $M/M/1/B$ has binomial, or Poisson, arrivals and departures. At a given time step at most one customer arrives and at most one customer departs. The queue has a finite buffer of size $B$ and the population size is infinite. This queue is frequently encountered when the time step is so short that only one customer can arrive or depart in that time.

The queue $M/M/J/B$ has binomial, or Poisson, arrivals and departures. At a given time step at most one customer arrives and at most one customer could depart through one of the $J$ available servers. The queue has a finite buffer of size $B$ and the population size is infinite. This type of queue might be encountered when a server might be busy serving a customer at a given time step and the remaining servers become available to serve other customers in the queue.

The queue $M^m/M/1/B$ has binomial, or Poisson, arrivals and departures. There is one server in the queue and at a given time step at most $m$ customers arrive and at most one customer departs because there is one server. The queue has a finite buffer of size $B$ and the population size is infinite.

The queue $M/M^m/1/B$ has binomial, or Poisson, arrivals and departures. There is one server in the queue and at a given time step at most one customer arrives and at most $J$ customers depart because the server could handle $J$ customers in one time step. The queue has a finite buffer of size $B$ and the population size is infinite.

The queues we shall deal with here will be one of the following types:

1. Single arrival, single departure infinite-size queues in which the transition matrix $\mathbf{P}$ is tridiagonal. Such a queue will be denoted by the symbols $M/M/1$.
2. Single arrival, single departure finite-size queues in which $\mathbf{P}$ is tridiagonal. Such a queue will be denoted by the symbols $M/M/1/B$.
3. Multiple arrival, single departure finite queues in which $\mathbf{P}$ is lower Hessenberg. Such a queue will be denoted by the symbols $M^m/M/1/B$.
4. Single arrival, multiple departure finite-size queues in which $\mathbf{P}$ is upper Hessenberg. Such a queue will be denoted by the symbols $M/M^m/1/B$.

## 7.2 Queue Throughput ($Th$)

Most often we are interested in estimating the rate of customers leaving the queue which is expressed as customers per time step or customers per second. We call this rate the *average output traffic* $N_a(out)$, or *throughput* ($Th$) of a queue. The throughput is given by:

$$Th = \text{ output data rate } = N_a(out) \tag{7.1}$$

The units of $Th$ in the above expression are packets/time step. Notice that this definition implies that $Th$ could never be negative.

When we talk about throughput of packets, we usually mean *goodput* which represents the packets that got through intact without collisions or other problems. Sometimes the authors talk about throughput to include good and corrupted packets. The difference between throughput and goodput is seldom discussed and the reader is advised to make certain which quantity is being dealt with in any discussion. Throughout this book we shall use the term throughput to imply good packets that got sent without corruption.

## 7.3 Efficiency ($\eta$) or Access Probability ($p_a$)

The *efficiency* ($\eta$) of the queue or its *access probability* ($p_a$) essentially gives the percentage of customers or packets transmitted in one time step through the system relative to the total number of arriving customers or packets one time step also. We shall use the term efficiency when we talk about queues and switches and will use the term access probability when we talk about interconnection networks. Efficiency or access probability essentially measures the effectiveness of the queue at processing data present at the input.

We define the *access probability* ($p_a$) or *efficiency* $\eta$ as the ratio of the average output traffic relative to the average input traffic.

$$p_a = \eta = \frac{N_a(out)}{N_a(in)} \tag{7.2}$$

This can be expressed in terms of the throughput

$$p_a = \eta = \frac{N_a(out)}{N_a(in)} = \frac{Th}{N_a(in)} \qquad \leq 1 \tag{7.3}$$

Notice that the access probability or efficiency could never be negative and could never be more than one.

If customers or packets are created within the queue, then we have to modify our definitions of both the input and output traffic to ensure that the efficiency never exceeds unity. This situation could in fact happen. For example, consider a packet switch that receives packets at its inputs then it routes these packets to the output ports. We expect that the number of packets leaving the switch per unit time will be smaller than or equal to the number of packets coming to the switch per unit time. The output traffic could be smaller than the input traffic when packets are lost within the switch when its internal buffers become full. However, the switch itself might generate its own packets to communicate with other switches or routers in the network. In that case, the traffic at the input could in fact be smaller than the output traffic due to the internally generated traffic. Let us leave this point to the problems at the end of the chapter.

Throughput and access probability are very useful in describing the behavior of a system. The two concepts are not equivalent and in fact we will see that the curves showing variation of efficiency with the queue parameters are not scaled version of throughput variation with queue parameters.

If the queue produces three customers on the average per time step while five customers could arrive, then the throughput is $Th = 3$ while its efficiency is $\eta = 3/5 = 60\,\%$. On the other hand, if the queue produces four customers on the average per time step while a maximum of six customers could potentially leave, then the throughput is $Th = 4$ and its efficiency is $\eta = 4/6 = 66.67\,\%$.

## 7.4 Traffic Conservation

When the queue reaches steady state, the incoming customers or packets have two options when they arrive at a queue: they either get processed and move through the queue or they get lost. We can therefore write the traffic conservation as

$$N_a(in) = N_a(out) + N_a(lost) \tag{7.4}$$

where $N_a(lost)$ is the average number of lost traffic or customers per unit time. The above equation is valid at steady state since the number of packets in the queue will be constant.

Dividing the above equation by $N_a(in)$ to normalize we get:

$$\eta + L = 1 \tag{7.5}$$

where $L$ is the customer, or traffic loss probability

$$L = 1 - \eta \tag{7.6}$$

Systems that have high efficiency will have low loss probability and vice versa. This situation is very similar to mechanical or electrical energy conversion systems characterized by input power, output power, and lost power due to friction or resistive effects.

## 7.5 M/M/1 Queue

In an $M/M/1$ queue at any time step, at most one customer could arrive and at most one customer could leave. An example of an $M/M/1$ queue is a first-in-first-out (FIFO) buffer of a communication system. This gives rise to simplest of queues in queuing analysis. The queue size here is assumed infinite. This is the discrete time equivalent to the famous $M/M/1$ queue for the continuous time case. At a certain time step the probability of packet arrival is $a$, which is equivalent to a birth event or increase in the queue population. The probability that a packet did not arrive is $b = 1 - a$. The probability that a packet leaves the queue is $c$, which is equivalent to death or reduction in the queue population. The probability that a packet does not leave the queue is $d = 1 - c$. The probability $c$ is representative of the server ability to process the customers or packets in the queue in one time step.

The number of customers or packets stored in the queue is the state of our system. Thus the queue is in state $s_i$ when there are $i$ customers or packets in the queue. The state transition diagram for the discrete-time M/M/1 queue is shown in Fig. 7.1. Changes in the queue size occur by at most one, i.e. only one packet could arrive or depart. Thus we expect the transition matrix to be tridiagonal.

The future state of the queue depends only on its current state. Thus we can model the queue as a discrete-time Markov chain. Since packet arrivals and departures are independent of the time index value, we have a homogeneous Markov chain. We assume that when a packet arrives, it could be serviced at the same time step and it could leave the queue with probability $c$. This results in the transition matrix given by:

$$\mathbf{P} = \begin{bmatrix} f_0 & bc & 0 & 0 & \cdots \\ ad & f & bc & 0 & \cdots \\ 0 & ad & f & bc & \cdots \\ 0 & 0 & ad & f & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{7.7}$$

where $b = 1 - a$, $d = 1 - c$, $f_0 = 1 - ad$, and $f = ac + bd$.

For example, starting with an empty queue (state $s_0$), Fig. 7.1 indicates that we move to state $s_1$ only when a packet arrives and no packet can depart, which is term $ad$ in the diagram. This transition is also indicated in the above transition matrix as the term at location (2,1) of the transition matrix.

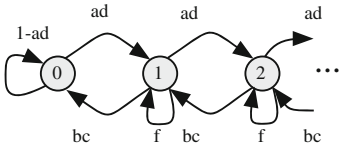



**Fig. 7.1** State transition diagram for the discrete-time M/M/1 queue

In order for the queue to be stable, the arrival probability must be smaller than the departure probability ($a < c$).

Since the dimension of $\mathbf{P}$ is infinite, we are going to obtain an expression for the distribution vector $\mathbf{s}$ using difference equations techniques instead of studying the eigenvectors of the matrix. The difference equations for the steady-state distribution vector are obtained from the equation:

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{7.8}$$

which produces the following difference equations:

$$ad\ s_0 - bc\ s_1 = 0 \tag{7.9}$$

$$ad\ s_0 - g\ s_1 + bc\ s_2 = 0 \tag{7.10}$$

$$ad\ s_{i-1} - g\ s_i + bc\ s_{i+1} = 0 \qquad i > 0 \tag{7.11}$$

where $g = 1 - f$ and $s_i$ is the probability that the system is in state $i$. For an infinite-size queue we have $i \geq 0$.

The solution to the above equations is given as:

$$s_1 = \left(\frac{a\ d}{b\ c}\right) s_0$$

$$s_2 = \left(\frac{a\ d}{b\ c}\right)^2 s_0$$

$$s_3 = \left(\frac{a\ d}{b\ c}\right)^3 s_0$$

and in general:

$$s_i = \left(\frac{a\ d}{b\ c}\right)^i s_0 \qquad i \geq 0$$

It is more convenient to write $s_i$ in the form:

$$s_i = \rho^i\ s_0 \qquad i \geq 0 \tag{7.12}$$

where $\rho$ is the *distribution index*:

$$\rho = \frac{a\ d}{b\ c} < 1 \tag{7.13}$$

The value of the distribution index will affect the component values of the distribution vector.

The complete solution is obtained from the above equations, plus the condition: $\sum_{i=0}^{\infty} s_i = 1$.

$$s_0 \sum_{i=0}^{\infty} \rho^i = 1 \tag{7.14}$$

Thus we obtain:

$$\frac{s_0}{1-\rho} = 1 \tag{7.15}$$

from which we obtain:

$$s_0 = 1 - \rho \tag{7.16}$$

and the components of the equilibrium distribution vector are given from (7.12) by:

$$s_i = (1-\rho)\rho^i \qquad i \geq 0 \tag{7.17}$$

It is interesting to compare this expression with the continuous-time $M/M/1$ queue. The components of the distribution vector for a continuous-time queue are given by

$$s_i = (1-\rho)\rho^i \qquad i \geq 0 \tag{7.18}$$

where $\rho$ for the continuous-time queue is called the *traffic intensity* and equals the ratio of the arrival rate to the service rate. The two expressions are identical for the simple $M/M/1$ queue.

### 7.5.1 $M/M/1$ Queue Performance

Once **s** is found, we can find the queue performance such as the throughput, average queue size, and packet delay.

The output traffic or average number of packets leaving the queue per time step is given by

$$N_a(out) = ac\ s_0 + \sum_{i=1}^{\infty} c\ s_i \tag{7.19}$$

The first term on RHS is the number of packets leaving the queue given that the queue is empty. The second term on RHS is the average number of packets leaving the queue when it is not empty. Simplifying we get:

$$\begin{aligned} N_a(out) &= a\ c\ s_0 + c\ (1 - s_0) \\ &= c - bc\ s_0 \\ &= a \end{aligned} \tag{7.20}$$

The units of $N_a(out)$ are packets/time step.

The throughput for the $M/M/1$ queue is given by:

$$Th = N_a(out) = a \tag{7.21}$$

The throughput is measured in units of packets/time step. To obtain the throughput in units of packets/s, we use the time step value:

$$Th' = \frac{Th}{T} \tag{7.22}$$

The input traffic or average number of packets entering the queue per time step is given by:

$$N_a(in) = 1 \times a + 0 \times b = a \tag{7.23}$$

This output traffic is measured in units of packets/time step.

The efficiency of the $M/M/1$ queue is given by:

$$\eta = \frac{N_a(out)}{N_a(in)} = 1 \tag{7.24}$$

The $M/M/1$ queue is characterized by maximum efficiency since input and output data rates are equal. There is no chance for packets to be lost since the infinite queue dues not ever fill up.

The average queue size is given by the equation

$$Q_a = \sum_{i=0}^{\infty} i\ s_i \tag{7.25}$$

Using (7.17) we get

$$Q_a = \frac{\rho}{1 - \rho} \tag{7.26}$$

The queue size is measured in units of packets or customers.

Figure 7.2 shows the exponential growth of the average queue size as the distribution index increases. A semilog plot was chosen here to show in more detail the size of the queue.

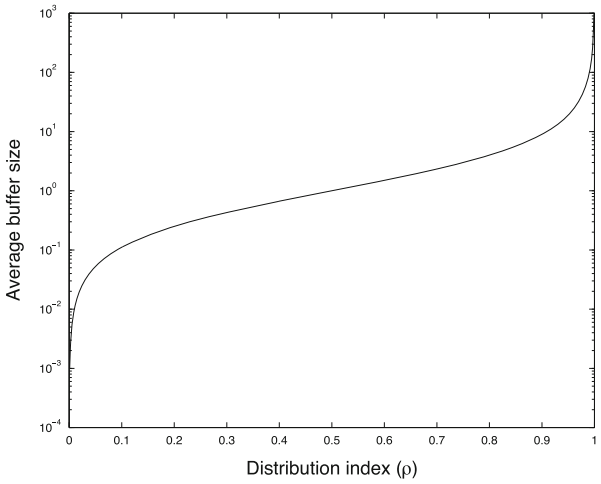



**Fig. 7.2** Average queue size versus the distribution index $\rho$ for the $M/M/1$ queue

We can invoke Little's result to estimate the *wait time*, which is the average number of time steps a packet spends in the queue before it is routed, as:

$$Q_a = W \times Th \tag{7.27}$$

where $W$ is the average number of time steps that a packet spends in the queue. Thus $W$ is given by:

$$W = \frac{\rho}{a(1-\rho)} \tag{7.28}$$

This wait time is measured in units of time steps. The wait time in units of seconds is given by the unnormalized version of Little's result:

$$W' = \frac{Q_a}{Th'} \tag{7.29}$$

*Example 7.1.* Consider the $M/M/1$ queue with the following parameters $a = 0.6$ and $c = 0.8$. Find the equilibrium distribution vector and the queue performance.

From (7.7), the transition matrix is:

$$\mathbf{P} = \begin{bmatrix} 0.88 & 0.32 & 0 & 0 & 0 & \cdots \\ 0.12 & 0.56 & 0.32 & 0 & 0 & \cdots \\ 0 & 0.12 & 0.56 & 0.32 & 0 & \cdots \\ 0 & 0 & 0.12 & 0.56 & 0.32 & \cdots \\ 0 & 0 & 0 & 0.12 & 0.56 & \cdots \\ 0 & 0 & 0 & 0 & 0.12 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

The steady-state distribution vector is found using (7.17):

$$\mathbf{s} = \begin{bmatrix} 0.625\,0 & 0.234\,4 & 0.087\,9 & 0.033\,0 & 0.012\,4 & 0.004\,6 & 0.001\,7 & \cdots \end{bmatrix}^t$$

The probability of being in state $i$ decreases exponentially as $i$ increases.

The queue performance is as follows:

$$\begin{aligned} Th &= 0.6 && \text{packets/time step} \\ \eta &= 1 \\ Q_a &= 0.6 && \text{packets} \\ W &= 1 && \text{time steps} \end{aligned}$$ ■

*Example 7.2.* Investigate the queue in the previous example when the arrival probability is very close to the departure probability.

For the queue to remain stable, we must have $a < c$. Let us try $a = 0.6$ and $c = a + 0.01$. The steady-state distribution vector is found using (7.17)

$$\mathbf{s} = \begin{bmatrix} 0.0410 & 0.0393 & 0.0377 & 0.0361 & 0.0347 & 0.0332 & 0.0319 & \cdots \end{bmatrix}^t$$

Comparing this distribution vector with its counterpart in the previous example, we see that the probability of being in state $i$ is increased for $i > 0$. This is an indication that the queue is getting close to being unstable.

The queue performance is as follows:

$$\begin{aligned} Th &= 0.6 && \text{packets/time step} \\ \eta &= 1 \\ Q_a &= 23.4 && \text{packets} \\ W &= 39 && \text{time steps} \end{aligned}$$

We see that the throughput is increased since the probability that the queue is empty (state 0) is decreased. ■

## 7.6 M/M/1/B Queue

This queue is similar to the discrete-time $M/M/1$ queue except that the queue has finite size $B$. The state transition diagram is shown in Fig. 7.3.

Since packet arrivals and departures are independent of the time index value, we have a homogeneous Markov chain. We assume that when a packet arrives, it could

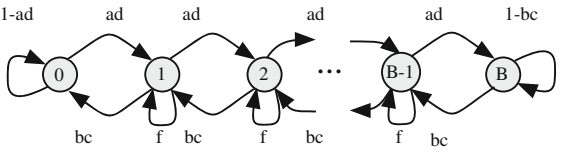



**Fig. 7.3** State transition diagram for the discrete-time $M/M/1/B$ queue

be serviced at the same time step and it could leave the queue with probability $c$. This results in the transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix} f_0 & bc & 0 & \cdots & 0 & 0 & 0 \\ ad & f & bc & \cdots & 0 & 0 & 0 \\ 0 & ad & f & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & f & bc & 0 \\ 0 & 0 & 0 & \cdots & ad & f & bc \\ 0 & 0 & 0 & \cdots & 0 & ad & 1-bc \end{bmatrix} \tag{7.30}$$

where $f_0 = 1 - ad$ and $f = ac + bd$.

Since the dimension of $\mathbf{P}$ is arbitrary, we are going to obtain an expression for the equilibrium distribution vector $\mathbf{s}$ using difference equations techniques. The difference equations for the state probability vector are given by:

$$ad\ s_0 - bc\ s_1 = 0 \tag{7.31}$$

$$ad\ s_0 - g\ s_1 + bc\ s_2 = 0 \tag{7.32}$$

$$ad\ s_{i-1} - g\ s_i + bc\ s_{i+1} = 0 \qquad 0 < i < B \tag{7.33}$$

where $g = ad + bc$ and $s_i$ is the component of the distribution vector corresponding to state $i$.

The solution to the above equations is given as:

$$s_1 = \left(\frac{a\ d}{b\ c}\right) s_0$$

$$s_2 = \left(\frac{a\ d}{b\ c}\right)^2 s_0$$

$$s_3 = \left(\frac{a\ d}{b\ c}\right)^3 s_0$$

and in general

$$s_i = \left(\frac{a\ d}{b\ c}\right)^i s_0 \qquad i \geq 0$$

It is more convenient to write $s_i$ in the form

$$s_i = \rho^i \; s_0 \qquad 0 \leq i \leq B \tag{7.34}$$

where $\rho$ is the distribution index for the $M/M/1/B$ queue:

$$\rho = \frac{a \; d}{b \; c}$$

The complete solution is obtained from the above equations plus the condition $\sum_{i=0}^{B} s_i = 1$ which gives:

$$s_0 \sum_{i=0}^{B} \rho^i = 1 \tag{7.35}$$

from which we obtain $s_0$, which is the probability that the queue is empty:

$$s_0 = \frac{1-\rho}{1-\rho^{B+1}} \tag{7.36}$$

and the equilibrium distribution for the other states is given from (7.34) by:

$$s_i = \frac{(1-\rho)\rho^i}{1-\rho^{B+1}} \qquad 0 \leq i \leq B \tag{7.37}$$

Note that $\rho$ for the finite-size queue *can* be more than one. In that case the queue will not be stable in the following sense:

$$s_0 < s_1 < s_2 \cdots < s_B \tag{7.38}$$

This indicates that the probability that the queue is full ($s_B$) is bigger than the probability that it is empty ($s_0$).

### 7.6.1 M/M/1/B Queue Performance

The throughput or output traffic for the $M/M/1/B$ queue is given by:

$$\begin{aligned} Th &= N_a(out) \\ &= ac \; s_0 + \sum_{i=1}^{B} c \; s_i \\ &= ac \; s_0 + c \; (1 - s_0) \\ &= c \; \; (1 - b \; s_0) \end{aligned} \tag{7.39}$$

This throughput is measured in units of packets/time step. The throughput in units of packets/s is:

$$Th' = \frac{Th}{T} \tag{7.40}$$

The input traffic is given by

$$N_a(in) = 1 \times a + 0 \times b = a \tag{7.41}$$

Input traffic is measured in units of packets/time step.

The efficiency of the $M/M/1/B$ queue is given by

$$\begin{aligned} \eta &= \frac{N_a(out)}{N_a(in)} \\ &= \frac{Th}{a} \\ &= \frac{c\,(1 - b\,s_0)}{a} \end{aligned} \tag{7.42}$$

Data is lost in the $M/M/1/B$ queue when it is full and packets arrive but does not leave. The average lost traffic $N_a(lost)$ is given by

$$N_a(lost) = s_B\ a\ d \tag{7.43}$$

The above equation is simply the probability that a packet is lost which equals the probability that the queue is full, and a packet arrives, and no packets can leave.

Lost traffic is measured in units of packets/time step. The average lost traffic per second is given by

$$N'_a(lost) = \frac{N_a(lost)}{T} \tag{7.44}$$

The packet loss probability $L$ is the ratio of lost traffic relative to the input traffic:

$$L = \frac{N_a(lost)}{N_a(in)} = s_B\ d \tag{7.45}$$

The average queue size is given by the equation:

$$Q_a = \sum_{i=0}^{B} i\ s_i \tag{7.46}$$

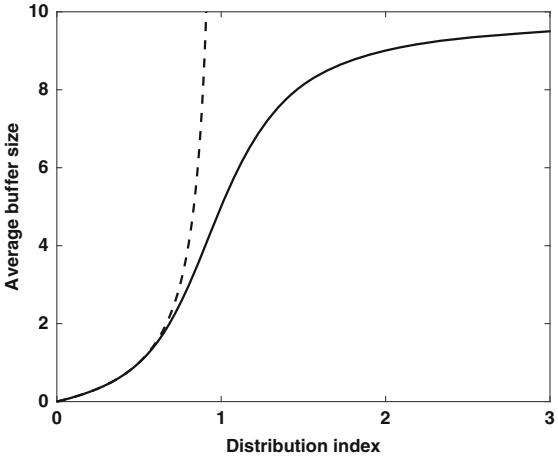



**Fig. 7.4** Average queue size versus the distribution index $\rho$ for the $M/M/1/B$ queue when $B = 10$ (*solid line*). The *dotted line* is average queue size for an infinite-size $M/M/1$ queue

Queue size is measured in units of packets. Using (7.37) the average queue size is given by:

$$Q_a = \frac{\rho \times \left[1 - (B+1)\rho^B + B\rho^{B+1}\right]}{(1-\rho) \times (1-\rho^{B+1})} \tag{7.47}$$

Figure 7.4 shows the exponential growth of the average queue size as the distribution index increases ($B = 10$ in that case). The solid line is for the $M/M/1/B$ queue and the dotted line is for the $M/M/1$ queue for comparison. We see that $Q_a$ for the finite-size queue grows at a slower rate with increasing distribution index compared to the infinite-size queue. Furthermore, $\rho$ for the infinite-size queue could increase beyond unity value.

We can invoke Little's result to estimate the average number of time steps a packet spends in the queue before it is routed as

$$Q_a = W \times Th \tag{7.48}$$

where $W$ is the wait time, or average number of time steps, that a packet spends in the queue. The throughput in the above expression for the wait time must be in units of packet/time step. The wait time is simply

$$W = \frac{Q_a}{Th} \tag{7.49}$$

Wait time is measured in units of time steps. The wait time in units of seconds is given by

$$W' = \frac{Q_a}{Th'} \qquad \text{s} \tag{7.50}$$

*Example 7.3.* Consider the $M/M/1/B$ queue with the following parameters $a = 0.6$, $c = 0.8$, and $B = 4$. Find the equilibrium distribution vector and the queue performance.

From (7.30), the transition matrix is

$$\mathbf{P} = \begin{bmatrix} 0.88 & 0.32 & 0 & 0 & 0 & 0 & \cdots \\ 0.12 & 0.56 & 0.32 & 0 & 0 & 0 & \cdots \\ 0 & 0.12 & 0.56 & 0.32 & 0 & 0 & \cdots \\ 0 & 0 & 0.12 & 0.56 & 0.32 & 0 & \cdots \\ 0 & 0 & 0 & 0.12 & 0.56 & 0.32 & \cdots \\ 0 & 0 & 0 & 0 & 0.12 & 0.56 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

The steady-state distribution vector is found using (7.37):

$$\mathbf{s} = \begin{bmatrix} 0.6297 & 0.2361 & 0.0885 & 0.0332 & 0.0125 \end{bmatrix}^t$$

Compare this distribution vector with the distribution vector of Example 7.1, which described an infinite-size queue with the same arrival and departure statistics. We see that components of the distribution vector here are slightly larger than their counterparts in the infinite-size queue as expected.

The queue performance is as follows:

$$\begin{aligned} N_a(out) = Th &= 0.5985 && \text{packets/time step} \\ \eta &= 0.9975 && \\ N_a(lost) &= 1.5 \times 10^{-3} && \text{packets/time step} \\ L &= 0.0025 && \\ Q_a &= 0.5626 && \text{packets} \\ W &= 0.9401 && \text{time steps} \end{aligned}$$

We note that the $M/M/1/B$ queue has smaller average size $Q_a$ and smaller wait time $W$ compared to the $M/M/1$ queue with the same arrival and departure statistics. As expected we have:

$$N_a(out) + N_a(lost) = N_a(in)$$ ■

*Example 7.4.* Find the performance of the queue in the previous example when the queue size becomes $B = 20$.

The queue performance is as follows:

$$\begin{aligned} N_a(out) = Th &= 0.6 && \text{packets/time step} \\ \eta &= 1 \\ N_a(lost) &= 2.2682 \times 10^{-10} && \text{packets/time step} \\ L &= 3.7804 \times 10^{-10} \\ Q_a &= 0.6 && \text{packets} \\ W &= 1 && \text{time steps} \end{aligned}$$

We see that increasing the queue size exponentially decreases the loss probability. The throughput increases by 0.25 % but the wait time is slightly increased due to the increased average queue size. ■

*Example 7.5.* Plot the $M/M/1/B$ performance when the input traffic varies between $0 \leq a \leq 1$ for $B = 10$ and $c = 0.5$.

Figure 7.5 shows the throughput, efficiency, loss probability, and delay to plot these quantities versus input traffic.

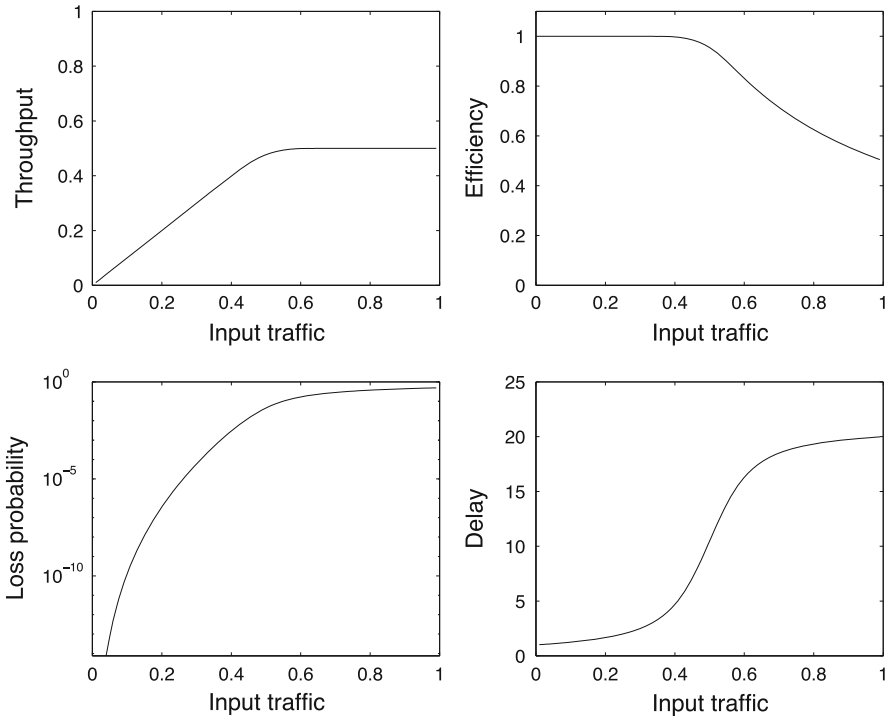



**Fig. 7.5** $M/M/1/B$ throughput, efficiency, loss probability, and delay to plot versus input traffic when $B = 10$ and $c = 0.5$

The important things to note from this example are:

1. The throughput of the queue could not exceed the maximum value for the average output traffic. Section 7.6.2 below will prove that this maximum value is simply $c$.
2. The efficiency of the queue is very close to 100 % until the input traffic approaches the maximum output traffic $c$.
3. Packet loss probability is always present but starts to increase when the input traffic approaches the packet maximum output traffic $c$.
4. Packet delay increases sharply when the input traffic approaches the packet maximum output traffic $c$.
5. Congestion conditions occur as soon as the input traffic exceeds the maximum output traffic $c$. Congestion is characterized by decreased efficiency, increased packet loss and increased delay.
6. The delay reaches a maximum value determined by the maximum size of the queue and the maximum output traffic $c$. The maximum delay could be approximated as

$$\text{Maximum Delay} = \frac{B}{c} = 20 \qquad \text{time steps}$$

■

### 7.6.2 *Performance Bounds on M/M/1/B Queue*

The previous example helped us get some rough estimates for the performance bounds of the $M/M/1/B$ queue. This section formalizes these estimates.

Under full load conditions, the $M/M/1/B$ become full and we can assume:

$$a \to 1 \tag{7.51}$$

$$b \to 0 \tag{7.52}$$

$$s_0 \to 0 \tag{7.53}$$

$$s_B \to 1 \tag{7.54}$$

$$Q_a \to B \tag{7.55}$$

The maximum throughput is given from (7.39) by:

$$\begin{aligned} Th(\max) &= N_a(out)_{\max} \\ &= c \end{aligned} \tag{7.56}$$

The departure probability is most important for determining the maximum throughput of the queue.

The minimum efficiency of the queue is given from (7.42) by

$$\eta(\min) = c \tag{7.57}$$

The departure probability is most important for determining the efficiency of the queue.

The maximum lost traffic is given from (7.43) by

$$N_a(lost)_{\max} = d = 1 - c \tag{7.58}$$

The maximum packet loss probability is given from (7.45) by

$$L(\max) = 1 - c \tag{7.59}$$

The maximum wait time is given by the approximate formula

$$W(\max) = \frac{B}{c} \tag{7.60}$$

Larger queues result in larger wait times as expected.

## 7.7 $M^m/M/1/B$ Queue

In an $M^m/M/1$ queue at any time step, at most $m$ customers could arrive and at most one customer could leave. We shall encounter this type of queue when we study network switches where FIFO queues exist at each output port. For each queue, a maximum of $m$ packets arrive at the queue input but only one packet can leave the queue. Therefore, the queue size can increase by more than one, but can only decrease by one in each time step. Assume the binomial probability of $k$ arrivals at instant $n$ is given by:

$$a_k = \binom{m}{k} a^k \, b^{m-k} \tag{7.61}$$

where $a$ is the probability that a packet arrives, $b = 1 - a$ and $m$ is the maximum number of packets that could arrive at the queue input. The queue size can only decrease by at most one at any instant with probability $c$. The probability that no packet leaves the queue is $d = 1 - c$. We assume that when a packet arrives, it could be serviced at the same time step and it could leave the queue with probability $c$.

The condition for the stability of the queue is:

$$\sum_{k=0}^{m} k\, a_k = a\, m < c \tag{7.62}$$

which indicates that the average number of arrivals at a given time is less than the average number of departures from the system. The resulting state transition matrix is a lower $(B+1) \times (B+1)$ Hessenberg matrix in which all the elements $p_{ij} = 0$ for $j > i + 1$.

$$\mathbf{P} = \begin{bmatrix} x & y_0 & 0 & \cdots & 0 & 0 \\ y_2 & y_1 & y_0 & \ldots & 0 & 0 \\ y_3 & y_2 & y_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ y_B & y_{B-1} & y_{B-2} & \cdots & y_1 & y_0 \\ z_B & z_{B-1} & z_{B-2} & \cdots & z_1 & z_0 \end{bmatrix} \tag{7.63}$$

where

$$x = a_1\, c + a_0 \tag{7.64}$$

$$y_i = a_i\, c + a_{i-1}\, d \tag{7.65}$$

$$z_i = a_i\, d + \sum_{k=i+1}^{m} a_k \tag{7.66}$$

where we assumed

$$a_k = 0 \qquad k < 0 \tag{7.67}$$

$$a_k = 0 \qquad k > m \tag{7.68}$$

The above transition matrix has $m$ subdiagonals when $m \leq B$.

### 7.7.1 $M^m/M/1/B$ Queue Performance

To calculate the throughput of the $M^m/M/1/B$ we need to consider the queue in two situations: when it is empty and when it is not.

The throughput of the $M^m/M/1/B$ queue when the queue is in state $s_0$ is given by

$$Th_0 = (1 - a_0)\, c \tag{7.69}$$

This is simply the probability that one or more packets arrive and one packet leaves the queue. When the queue is in any other state, the throughput is given by

$$Th_i = c \qquad 1 \leq i \leq B \tag{7.70}$$

This is simply the probability that a packet leaves the queue. The average throughput is estimated as

$$\begin{aligned} Th &= \sum_{i=0}^{B} Th_i \ s_i \\ &= c \ (1 - a_0 \ s_0) \end{aligned} \tag{7.71}$$

The throughput is measured in units of packets/time step. The throughput in units of packets/s is

$$Th' = \frac{Th}{T} \tag{7.72}$$

The input traffic is given by

$$\begin{aligned} N_a(in) &= \sum_{i=0}^{m} i \ a_i \\ &= m \ a \end{aligned} \tag{7.73}$$

The efficiency of the $M^m/M/1/B$ queue is given by

$$\begin{aligned} \eta &= \frac{N_a(out)}{N_a(in)} \\ &= \frac{Th}{m \ a} \\ &= \frac{c \ (1 - a_0 \ s_0)}{m \ a} \end{aligned} \tag{7.74}$$

Data is lost in the $M^m/M/1/B$ queue when it becomes full and packets arrive but does not leave. However, due to multiple arrivals, packets could be lost even when the queue is not completely full. For example, we could still have one location left in the queue but three customers arrive. Definitely packets will be lost then. Therefore, we conclude that average lost traffic $N_a(lost)$ is a bit difficult to obtain. However the traffic conservation principle is useful in getting a simple expression for lost traffic. The average lost traffic $N_a(lost)$ is given by

$$\begin{aligned} N_a(lost) &= N_a(in) - N_a(out) \\ &= m \ a - c \ (1 - a_0 \ s_0) \end{aligned} \tag{7.75}$$

The lost traffic is measured in units of packets per time step. The average lost traffic measured in packets per second is given by

$$N'_a(lost) = \frac{N_a(lost)}{T} \tag{7.76}$$

The packet loss probability $L$ is the ratio of lost traffic relative to the input traffic

$$\begin{aligned} L &= \frac{N_a(lost)}{N_a(in)} = 1 - \eta \\ &= 1 - \frac{c\ (1 - a_0\ s_0)}{m\ a} \end{aligned} \tag{7.77}$$

The average queue size is given by the equation

$$Q_a = \sum_{i=0}^{B} i\ s_i \tag{7.78}$$

We can invoke Little's result to estimate the *wait time*, which is the average number of time steps a packet spends in the queue before it is routed, as

$$Q_a = W \times Th \tag{7.79}$$

where $W$ is the average number of time steps that a packet spends in the queue. Thus $W$ is given by

$$W = \frac{Q_a}{Th} \tag{7.80}$$

The wait time is measured in units of time steps. The wait time in units of seconds is given by the unnormalized version of Little's result.

$$W' = \frac{Q_a}{Th'} \tag{7.81}$$

*Example 7.6.* Consider the $M^m/M/1/B$ queue with the following parameters $a = 0.04$, $m = 2$, $c = 0.1$, and $B = 5$. Check its stability condition and find the equilibrium distribution vector and queue performance.

The packet arrival probability is

$$a_k = \binom{2}{k}(0.04)^k(0.96)^{2-k}$$

The stability condition is found from (7.62)

$$\sum_{k=0}^{2} k\, a_k = 0.08 < c$$

The queue is stable and the transition matrix is

$$\mathbf{P} = \begin{bmatrix} 0.9293 & 0.0922 & 0 & 0 & 0 & 0 \\ 0.0693 & 0.8371 & 0.0922 & 0 & 0 & 0 \\ 0.0014 & 0.0693 & 0.8371 & 0.0922 & 0 & 0 \\ 0 & 0.0014 & 0.0693 & 0.8371 & 0.0922 & 0 \\ 0 & 0 & 0.0014 & 0.0693 & 0.8371 & 0.0922 \\ 0 & 0 & 0 & 0.0014 & 0.0707 & 0.9078 \end{bmatrix}$$

The transition matrix has two subdiagonals because $m = 2$.

The steady-state distribution vector is the eigenvector of $\mathbf{P}$ that corresponds to unity eigenvalue. We have

$$\mathbf{s} = \begin{bmatrix} 0.2843 & 0.2182 & 0.1719 & 0.1353 & 0.1065 & 0.0838 \end{bmatrix}^t$$

We see that the most probable state for the queue is state $s_0$ which occurs with probability 28.43 %.

The queue performance is as follows:

$$\begin{aligned} Th &= 0.0738 && \text{packets/time step} \\ \eta &= 0.9225 && \\ N_a(lost) &= 6.2 \times 10^{-3} && \text{packets/time step} \\ L &= 0.0775 && \\ Q_a &= 1.813 && \text{packets} \\ W &= 24.5673 && \text{time steps} \end{aligned}$$

Flow conservation is verified since the sum of the throughput and the lost traffic equal the input traffic $N_a(in) = m\, a$. ■

*Example 7.7.* Find the performance of the queue in the previous example when the queue size becomes $B = 20$.

The queue performance is as follows:

$$\begin{aligned} Th &= 0.0799 && \text{packets/time step} \\ \eta &= 0.9984 && \end{aligned}$$

$$
\begin{aligned}
N_a(lost) &= 1.3167 \times 10^{-4} && \text{packets/time step} \\
L &= 0.0016 && \\
Q_a &= 1.813 && \text{packets} \\
W &= 44.3432 && \text{time steps}
\end{aligned}
$$

We see that increasing the queue size decreases the loss probability which results in only a slight increase in the throughput. The wait time is almost doubled. ■

In Chap. 4 we explored different techniques for finding the equilibrium distribution for the distribution vector **s**. For simple situations when the value of $m$ is small (1 or 2) we can use the difference equations approach. When $m$ is large, we can use the z-transform technique as was done in Sect. 4.8.

*Example 7.8.* Plot the $M^m/M/1/B$ performance when $m = 2$ and the input traffic varies between $0 \leq a \leq 1$ for $B = 10$ and $c = 0.5$.

Figure 7.6 shows the variation of the throughput, efficiency, loss probability, and delay versus the versus average input traffic.

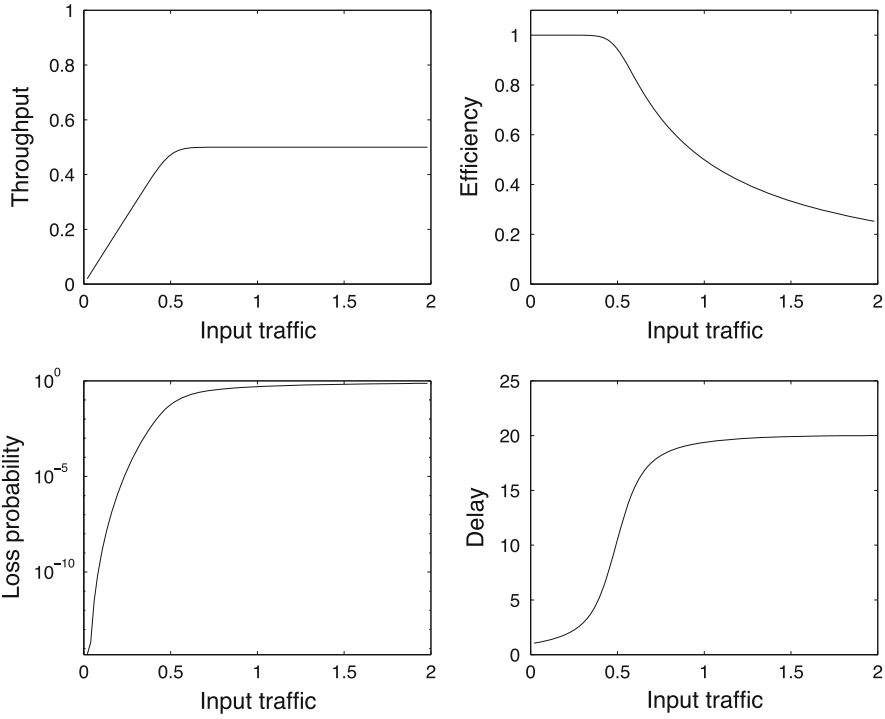



**Fig. 7.6** $M^m/M/1/B$ throughput, efficiency, loss probability, and delay to plot versus input traffic when $m = 2$, $B = 10$ and $c = 0.5$

The important things to note from this example are:

1. The throughput of the queue could not exceed the maximum output traffic $c$.
2. The efficiency of the queue is very close to 100 % until the input traffic approaches the maximum output traffic $c$.
3. Packet loss probability is always present but starts to increase when the input traffic approaches the packet maximum output traffic $c$.
4. Packet delay increases sharply when the input traffic approaches the packet maximum output traffic $c$.
5. Congestion conditions occur as soon as the input traffic exceeds the maximum output traffic $c$. Congestion is characterized by decreased efficiency, increased packet loss, and increased delay. ■

### 7.7.2 *Performance Bounds on $M^m/M/1/B$ Queue*

The previous example helped us get some rough estimates for the performance bounds of the $M^m/M/1/B$ queue. This section formalizes these estimates.

Under full load conditions, the $M^m/M/1/B$ become full and we can assume

$$a \to 1 \tag{7.82}$$

$$b \to 0 \tag{7.83}$$

$$s_0 \to 0 \tag{7.84}$$

$$s_B \to 1 \tag{7.85}$$

$$Q_a \to B \tag{7.86}$$

The maximum value for the throughput from (7.71) becomes

$$Th(\max) = c \tag{7.87}$$

The packet departure probability determines the maximum throughput of the queue.

The minimum efficiency of the $M^m/M/1/B$ queue is given from (7.74) by

$$\eta(\min) = \frac{c}{m} \tag{7.88}$$

The maximum lost traffic is given from (7.77) by

$$N_a(lost)_{\max} = m - c \tag{7.89}$$

The asymptotic value for packet loss probability in (7.77) is obtained when $s_0$ is zero and is given by:

$$L(\max) = 1 - \frac{c}{m} \tag{7.90}$$

When more customers arrive per time step (large $m$), the probability of loss increases. Maximum loss probability increases when packet departure probability decreases and when the number of arriving customers increases.

The maximum delay is given from (7.80) by

$$W(\max) = \frac{B}{Th(\max)} = \frac{B}{c} \tag{7.91}$$

### 7.7.3 *Alternative Solution Method*

When $B$ is large, it is better to use numerical techniques such as forward- or backward substitution using Givens rotations.[1]

At steady state we can write

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{7.92}$$

We can use the technique explained in Sect. 4.10 on page 146 and in reference [4] to construct a system of linear equations that can be solved using any of the specialized software designed to solve large systems of linear equations.

## 7.8 $M/M^m/1/B$ Queue

In an $M/M^m/1$ queue at any time step, at most one customer could arrive and at most $m$ customers could leave. We shall encounter this type of queue when we study network switches where FIFO queues exist at each input port. For each queue, one packet arrives at the input line to the storage buffer but a maximum of $m$ packets can leave the queue destined to the different switch outputs. Therefore, the queue size can increase by one, but can decrease by more than one. The probability that a packet arrives is $a$ and $b = 1 - a$ is the probability that a packet does not arrive at a time step. Define $c_{i,j}$ as the probability that $j$ customers leave the queue when there are $i$ customers in the queue.

$$c_{i,j} = \binom{i}{j} c^j \; d^{i-j} \tag{7.93}$$

[1] Another useful technique for triangularizing a matrix is to use Householder transformation. However, we prefer Givens rotation due to its numerical stability. Alston Householder once commented that he would never fly in an airplane that was designed with the help of a computer using floating-point arithmetic [4].

where $c$ is the probability that a packet departs and $d = 1 - c$. The state transition matrix is an upper $(B + 1) \times (B + 1)$ Hessenberg matrix in which all the elements of subdiagonals 2, 3, $\cdots$ are zero—i.e. $p_{ij} = 0$ for $i > j + 1$. The matrix has only $m$ superdiagonals. We assume an arriving packet is served at the same time step and we also assume that when the queue is full any arriving packets are discarded. Nonzero element $p_{i,j}$ of the matrix is expressed in general as

$$p_{i,j} = \begin{cases} b \sum_{k=m}^{j} c_{j,k} & j - i = m, \quad j < B \\ a \sum_{k=m}^{j+1} c_{j,k} + b c_{j,m-1} & j - i + 1 = m, \quad j < B \\ a c_{j+1,j-i+1} + b c_{j,j-i} & j < B \\ c_{m,B-i} & B - m < i \leq B, \quad j = B \\ \sum_{k=m}^{B} c_{B,B-i} & B - m < i \leq B \quad j = B \end{cases} \tag{7.94}$$

For the case when $B = 6$ and $m = 3$ the transition matrix $P$ will have the form

$$\mathbf{P} = \begin{bmatrix} q_0 & r_1 & s_2 & t_3 & 0 & 0 & 0 \\ p_0 & q_1 & r_2 & s_3 & t_4 & 0 & 0 \\ & p_1 & q_2 & r_3 & s_4 & t_5 & 0 \\ 0 & 0 & p_2 & q_3 & r_4 & s_5 & y \\ 0 & 0 & 0 & p_3 & q_4 & r_5 & x_2 \\ 0 & 0 & 0 & 0 & p_4 & q_5 & x_1 \\ 0 & 0 & 0 & 0 & 0 & p_5 & x_0 \end{bmatrix} \tag{7.95}$$

where the matrix elements are given by the general expressions

$$p_i = a\, c_{i+1,0} \tag{7.96}$$

$$q_i = a\, c_{i+1,1} + b\, c_{i,0} \tag{7.97}$$

$$r_i = a\, c_{i+1,2} + b\, c_{i,1} \tag{7.98}$$

$$s_i = a \sum_{j=m}^{i+1} c_{i+1,j} + b\, c_{i,2} \tag{7.99}$$

$$t_i = b \sum_{j=m}^{i} c_{i,j} \tag{7.100}$$

$$x_i = c_{6,i} \tag{7.101}$$

$$y = \sum_{j=m}^{B} c_{B,j} \tag{7.102}$$

The condition for the stability of the queue is when average traffic at the queue input is smaller than the average traffic at the queue output.

$$a < m\, c \tag{7.103}$$

### 7.8.1 $M/M^m/1/B$ Queue Performance

The average input traffic for the $M/M^m/1/B$ queue is obtained simply as

$$N_a(in) = a \tag{7.104}$$

Traffic is lost in the $M/M^m/1/B$ queue when it becomes full and a packet arrives while no packets leave. The average lost traffic $N_a(lost)$ is expressed simply as

$$N_a(lost) = a\ s_B\ c_{B,0} \tag{7.105}$$

The packet loss probability $L$ is the ratio of lost traffic relative to the input traffic

$$L = \frac{N_a(lost)}{N_a(in)} = s_B\ c_{B,0} \tag{7.106}$$

The throughput of the queue is obtained using the traffic conservation principle

$$\begin{aligned} Th &= N_a(in) - N_a(lost) \\ &= a\,(1 - s_B c_{B,0}) \end{aligned} \tag{7.107}$$

The efficiency of the $M/M^m/1/B$ queue is given by

$$\begin{aligned} \eta &= 1 - L \\ &= 1 - s_B\ c_{B,0} \end{aligned} \tag{7.108}$$

The average queue size is given by the equation

$$Q_a = \sum_{i=0}^{B} i\ s_i \tag{7.109}$$

We can invoke Little's result to estimate the *wait time*, which is the average number of time steps a packet spends in the queue before it is routed.

$$Q_a = W \times Th \tag{7.110}$$

where $W$ is the average number of time steps that a packet spends in the queue. Thus $W$ is given by

$$W = \frac{Q_a}{Th} \tag{7.111}$$

The wait time is measured in units of time steps. The wait time in units of seconds is given by the unnormalized version of Little's result.

$$W' = \frac{Q_a}{Th'} \tag{7.112}$$

*Example 7.9.* Consider the $M/M^m/1/B$ queue with the following parameters $a = 0.1$, $c = 0.07$, $m = 2$, and $B = 5$. Check its stability condition and find the equilibrium distribution vector and the queue performance.

Using (7.95), the transition matrix is

$$\mathbf{P} = \begin{bmatrix} 0.9070 & 0.0635 & 0.0044 & 0 & 0 & 0 \\ 0.0930 & 0.8500 & 0.1186 & 0.0126 & 0 & 0 \\ 0 & 0.0865 & 0.7966 & 0.1661 & 0.0241 & 0 \\ 0 & 0 & 0.0804 & 0.7464 & 0.2069 & 0.0425 \\ 0 & 0 & 0 & 0.0748 & 0.6994 & 0.2618 \\ 0 & 0 & 0 & 0 & 0.0696 & 0.6957 \end{bmatrix}$$

The transition matrix has two superdiagonals because $m = 2$.

The steady-state distribution vector is the eigenvector of $\mathbf{P}$ that corresponds to unity eigenvalue. We have

$$\mathbf{s} = \begin{bmatrix} 0.2561 & 0.3584 & 0.2414 & 0.1043 & 0.0324 & 0.0074 \end{bmatrix}^t$$

We see that the most probable state for the queue is state $s_1$ which occurs with probability 35.8 %.

The queue performance is as follows:

$$\begin{aligned} N_a(lost) &= 6.4058 \times 10^{-4} && \text{packets/time step} \\ Th &= 0.0994 && \text{packets/time step} \\ L &= 0.0064 \times 10^{-2} && \\ \eta &= 0.9936 && \\ Q_a &= 1.3205 && \text{packets} \\ W &= 13.2906 && \text{time steps} \end{aligned}$$

■

*Example 7.10.* Find the performance of the queue in the previous example when the queue size becomes $B = 10$.

The queue performance is as follows:

$$\begin{aligned} N_a(lost) &= 2.6263 \times 10^{-8} && \text{packets/time step} \\ Th &= 0.1 && \text{packets/time step} \\ L &= 2.6263 \times 10^{-7} && \\ \eta &= 1 && \\ Q_a &= 1.3286 && \text{packets} \\ W &= 13.2862 && \text{time steps} \end{aligned}$$

We see that increasing the queue size exponentially decreases the loss probability. The throughput is not changed by much. ■

*Example 7.11.* Plot the $M/M^m/1/B$ performance when $m = 2$ and the input traffic varies between $0 \leq a \leq 1$ for $B = 5$ and $c = 0.05$.

Figure 7.7 shows the throughput, efficiency, loss probability, and delay to plot these quantities versus input traffic.

The important things to note from this example are:

1. The throughput of the queue increases with increasing input traffic but shows slight decrease when the input traffic approaches the value $m\,c$.
2. The efficiency of the queue is very close to 100 % until the input traffic approaches the maximum output traffic $m\,c$.
3. Packet loss probability is always present but starts to increase when the input traffic approaches the value $m\,c$.
4. Packet decreases when the input traffic approaches the value $m\,c$. This is due to the queue size becomes constant while the throughput keeps increasing. ■

### 7.8.2 Performance Bounds on $M/M^m/1/B$ Queue

The previous examples help us get some rough estimates for the performance bounds of the $M/M/^m 1/B$ queue.

Under full load conditions, the $M/M^m/1/B$ become full and we can assume

$$a \to 1 \tag{7.113}$$

$$b \to 0 \tag{7.114}$$

$$s_0 \to 0 \tag{7.115}$$

$$s_B \to 1 \tag{7.116}$$

$$Q_a \to B \tag{7.117}$$

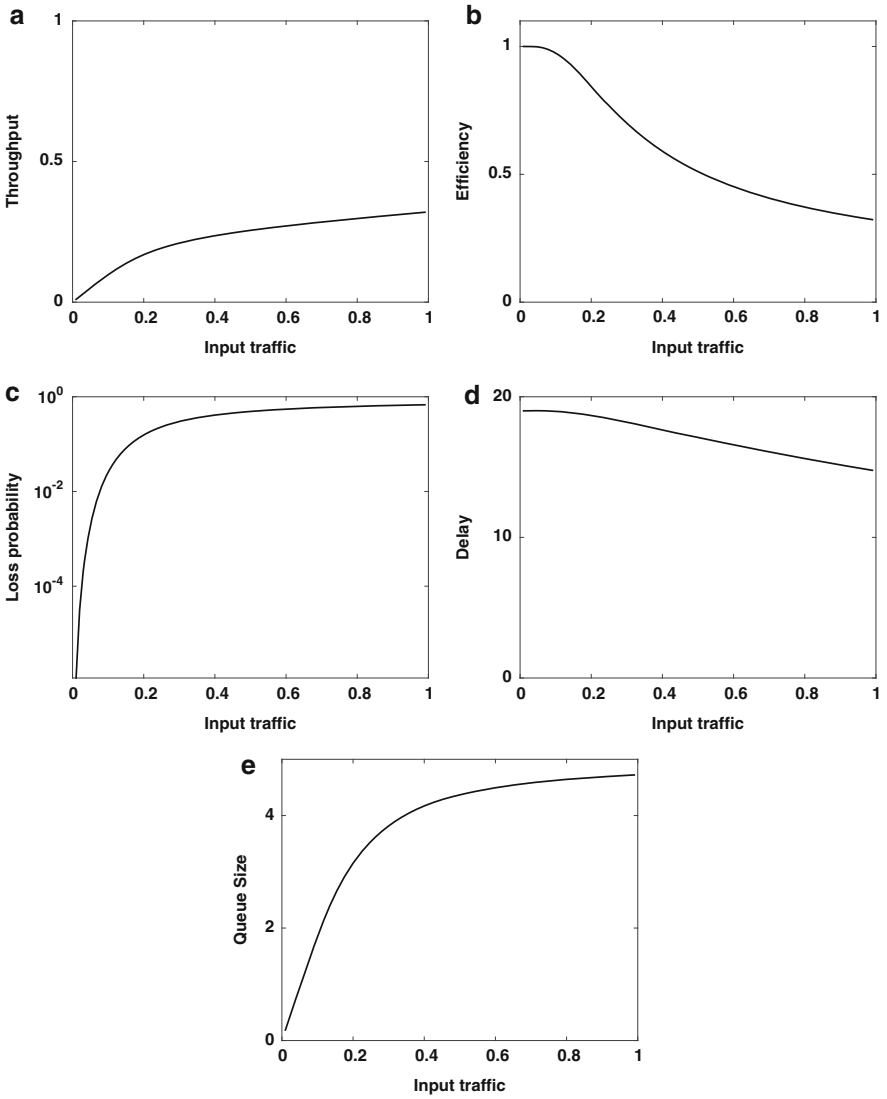



**Fig. 7.7** $M/M^m/1/B$ throughput, efficiency, loss probability, and delay versus input traffic when $m = 2$, $B = 5$, and $c = 0.05$

The maximum lost traffic is given from (7.105) by

$$\begin{aligned} N_a(lost)_{\max} &= c_{B,0} \\ &= (1-c)^m \\ &\leq d^m \end{aligned} \tag{7.118}$$

where $n = \min(B, m)$. The reason for the inequality sign is that $s_B$ seldom approaches 1.

The maximum packet loss probability is given from (7.106) by

$$\begin{aligned} L(\max) &= c_{B,0} \\ &= (1-c)^n \\ &\leq d^n \end{aligned} \tag{7.119}$$

The maximum throughput is given from (7.107) by

$$\begin{aligned} Th(\max) &= 1 - c_{B,0} \\ &\geq 1 - (1-c)^m \\ &= 1 - d^m \end{aligned} \tag{7.120}$$

The minimum efficiency of the $M/M^m/1/B$ queue is given from (7.108) by

$$\eta(\min) \geq 1 - d^m \tag{7.121}$$

The maximum delay is given by the approximate formula

$$\begin{aligned} W(\max) &\leq \frac{B}{Th(\max)} \\ &\leq \frac{B}{1-d^m} \\ &\leq \frac{B}{1-d^m} \end{aligned} \tag{7.122}$$

### 7.8.3 Alternative Solution Method

When $B$ is large, it is better to use numerical techniques such as forward- or backward substitution using Givens rotations.

At steady state we can write

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{7.123}$$

We can use the technique explained in Sect. 4.10 on page 146 to construct a system of linear equations that can be solved using any of the specialized software designed to solve large systems of linear equations.

## 7.9 The $D/M/1/B$ Queue

In the $D/M/1/B$ queue packets arrive at a fixed rate but leave the queue in a random fashion. Let us assume that the time step is chosen such that exactly one packet arrives at the $n$-th time step. Assume also that $c$ is the probability that a packet leaves the queue during one time step. We also assume that at most one packet leaves the queue in one time step. $d = 1 - c$ has the usual meaning. Figure 7.8 shows the state transition diagram for such queue for the case when $n = 4$ and $B = 4$ also. The number of rows corresponds to the number of time steps between packet arrivals. The last row corresponds to the states when a packet arrives. The number of columns corresponds to the size of the queue such that each column corresponds to a particular state of queue occupancy. For example, the leftmost column corresponds to the case when the queue is empty. The rightmost column corresponds to the case when the queue is full. The state of occupancy of the queue is indicated in Table 7.1.

In that case we know that a packet arrives when the queue is in one of the bottom states $s_{3,0}$, $s_{3,1}$, $s_{3,2}$, $s_{3,3}$, or $s_{3,4}$.

The state vector $\mathbf{s}$ can be grouped into $B$ subvectors

$$\mathbf{s} = \left[ \mathbf{s}_0 \; \mathbf{s}_1 \; \cdots \; \mathbf{s}_B \right]^t \tag{7.124}$$

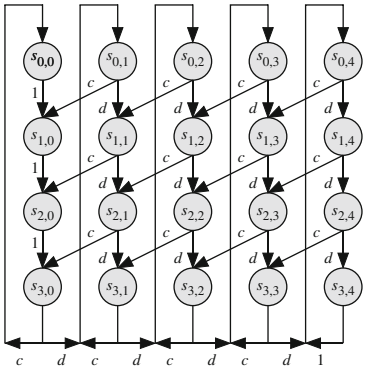



**Fig. 7.8** State transition diagram for the discrete-time $D/M/1/B$ queue

**Table 7.1** Relation of the queue states and the $D/M/1/B$ queue occupancy

| States | Queue occupancy |
|---|---|
| $s_{0,0}$–$s_{3,0}$ | Queue empty |
| $s_{0,1}$–$s_{3,1}$ | One customer in queue |
| $s_{0,2}$–$s_{3,2}$ | Two customers in queue |
| ⋮ | ⋮ |
| $s_{0,j}$–$s_{3,j}$ | $j$ customers in queue |
| ⋮ | ⋮ |

where the subvector $\mathbf{s}_j$ corresponds to the $j$-th column in Fig. 7.8 and is given by

$$\mathbf{s}_j = \begin{bmatrix} s_{0,j} & s_{1,j} & \cdots & s_{n-1,j} \end{bmatrix}^t \tag{7.125}$$

corresponds to the case when there are $j$ customers in the queue. The state transition matrix $\mathbf{P}$ corresponding to the state vector $\mathbf{s}$ will be of dimension $n(B+1)\times n(B+1)$. We describe the transition matrix $\mathbf{P}$ as a composite matrix of size $(B+1)\times(B+1)$ as follows.

$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{C} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B} & \mathbf{D} & \mathbf{C} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{D} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{D} & \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{B} & \mathbf{D} & \mathbf{C} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{B} & \mathbf{E} \end{bmatrix} \tag{7.126}$$

where all the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$, and $\mathbf{E}$ are of dimension $n \times n$. For the case $n = 4$ we can write

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & c \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & d \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & c & 0 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 & c \\ d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & d & 0 \end{bmatrix}, \qquad \mathbf{E} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & d & 0 \end{bmatrix}$$

Having found the state transition matrix, we are now able to find the steady state value for the distribution vector of the $D/M/1/B$ queue. At steady state the distribution vector $\mathbf{s}$ is derived from the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{7.127}$$

$$\mathbf{1}\,\mathbf{s} = 1 \tag{7.128}$$

where $\mathbf{1}$ is a row vector whose components are all 1's.

We can find the vector $\mathbf{s}$ by iterations as follows. We start by assuming a value for element $s_{0,0} = 1$. As a consequence, all the elements of the vector $\mathbf{s}_0$ can be found as follows

$$s_{1,0} = s_{0,0} = 1 \tag{7.129}$$

$$s_{2,0} = s_{1,0} = 1 \tag{7.130}$$

$$s_{3,0} = s_{2,0} = 1 \tag{7.131}$$

$$\vdots \tag{7.132}$$

Thus we know that $\mathbf{s}_0$ is assumed to be a vector whose components are all ones. To find $\mathbf{s}_1$ we use the equation

$$\mathbf{s}_0 = \mathbf{A}\mathbf{s}_0 + \mathbf{C}\mathbf{s}_1 \tag{7.133}$$

or

$$\mathbf{s}_1 = \mathbf{C}^{-1}(\mathbf{I} - \mathbf{A})\mathbf{s}_0 \tag{7.134}$$

Since we have an initial assumed value for $\mathbf{s}_0$, we now know $\mathbf{s}_1$. In general we can write the iterative expressions

$$\mathbf{s}_i = \mathbf{C}^{-1}(\mathbf{I} - \mathbf{A})\mathbf{s}_{i-1} \quad 1 \leq i \leq B \tag{7.135}$$

Having found all the vectors $\mathbf{s}_i$, we obtained the normalized distribution vector $\mathbf{s}'$ as

$$\mathbf{s}' = s / \sum_{i=0}^{n-1} \sum_{j=0}^{B} s_{i,j} \tag{7.136}$$

### *7.9.1 Performance of the $D/M/1/B$ Queue*

The average input traffic $N_a(in)$ is needed to estimate the efficiency of the queue and queue delay. Since we get one packet every $n$ time steps, $N_a(in)$ is given by

$$N_a(in) = \frac{1}{n} \tag{7.137}$$

The throughput of the queue is given by

$$\begin{aligned} Th &= c s_{n-1,0} + c \sum_{i=0}^{n-1} \sum_{j=1}^{B} s_{i,j} \\ &= c \left( 1 - \sum_{i=0}^{n-2} s_{i,0} \right) \end{aligned} \tag{7.138}$$

The efficiency of the $D/M/1/B$ queue is given by

$$\eta = \frac{Th}{N_a(in)} = cn\left(1 - \sum_{i=0}^{n-2} s_{i,0}\right) \tag{7.139}$$

Packets are lost in the $D/M/1B$ queue when the queue is full and a packet arrives but does not leave. The average lost traffic is given by

$$N_a(lost) = ds_{n-1,B} \tag{7.140}$$

The packet loss probability $L$ is given by

$$L = \frac{N_a(lost)}{N_a(in)} = dns_{n-1,B} \tag{7.141}$$

The average queue size is given by

$$Q_a = \sum_{i=0}^{n-1} \sum_{j=0}^{B} i\ s_{i,j} \tag{7.142}$$

## 7.10 The $M/D/1/B$ Queue

In the $M/D/1/B$ queue packets arrive in a random fashion but leave the queue at a fixed rate. Let us assume that the time step is chosen such that a packet leaves at the $n$-th time step. Assume also that $a$ is the probability that it arrives at queue during one time step. $b = 1 - a$ has the usual meaning. We also assume that at most one packet arrives in the queue in one time step. Figure 7.9 shows the state transition

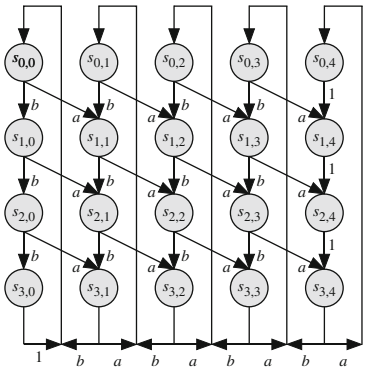



**Fig. 7.9** State transition diagram for the discrete-time $M/D/1/B$ queue

**Table 7.2** Relation of the queue states and the $M/D/1/B$ queue occupancy

| States | Queue occupancy |
|---|---|
| $s_{0,0}$–$s_{3,0}$ | Queue empty |
| $s_{0,1}$–$s_{3,1}$ | One customer in queue |
| $s_{0,2}$–$s_{3,2}$ | Two customers in queue |
| $\vdots$ | $\vdots$ |
| $s_{0,j}$–$s_{3,j}$ | $j$ customers in queue |
| $\vdots$ | $\vdots$ |

diagram for such queue for the case when $n = 4$ and $B = 4$ also. The number of rows corresponds to the number of time steps between packet departures. The last row corresponds to the states when a packet leaves. The number of columns corresponds to the size of the queue such that each column corresponds to a particular state of queue occupancy. For example, the leftmost column corresponds to the case when the queue is empty. The rightmost column corresponds to the case when the queue is full. The state of occupancy of the queue is indicated in Table 7.2 In that case we know that a packet leaves when the queue is in one of the bottom states $s_{3,1}$, $s_{3,2}$, $s_{3,3}$, or $s_{3,4}$.

The state vector $\mathbf{s}$ can be grouped into $B$ subvectors

$$\mathbf{s} = \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_B \end{bmatrix}^t \tag{7.143}$$

where the subvector $\mathbf{s}_j$ corresponds to the $j$-th column in Fig. 7.8 and is given by

$$\mathbf{s}_j = \begin{bmatrix} s_{0,j} & s_{1,j} & \cdots & s_{n-1,j} \end{bmatrix}^t \tag{7.144}$$

corresponds to the case when there are $j$ customers in the queue.

The state transition matrix $\mathbf{P}$ corresponding to the state vector $\mathbf{s}$ will be of dimension $n(B+1) \times n(B+1)$. We describe the transition matrix $\mathbf{P}$ as a composite matrix of size $(B+1) \times (B+1)$ as follows.

$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{C} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{B} & \mathbf{D} & \mathbf{C} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{D} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{D} & \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{B} & \mathbf{D} & \mathbf{C} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{B} & \mathbf{E} \end{bmatrix} \tag{7.145}$$

where all the matrices **A**, **B**, **C**, **D**, and **E** are of dimension $n \times n$. For the case $n = 4$ we can write

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ b & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & b & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & b \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 & a \\ b & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & b & 0 \end{bmatrix}, \qquad \mathbf{E} = \begin{bmatrix} 0 & 0 & 0 & a \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Having found the state transition matrix, we are now able to find the steady state value for the distribution vector of the $M/D/1/B$ queue. At steady state the distribution vector **s** is derived from the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{7.146}$$

$$\mathbf{1}\,\mathbf{s} = 1 \tag{7.147}$$

where **1** is a row vector whose components are all 1's.

We can find the vector **s** by iterations as follows. We start by assuming a value for element $s_{0,0} = 1$. As a consequence, all the elements of the vector $\mathbf{s}_0$ can be found as follows

$$s_{1,0} = bs_{0,0} = b \tag{7.148}$$

$$s_{2,0} = bs_{1,0} = b^2 \tag{7.149}$$

$$s_{3,0} = bs_{2,0} = b^3 \tag{7.150}$$

$$\vdots \tag{7.151}$$

Thus we know $\mathbf{s}_0$. To find $\mathbf{s}_1$ we use the equation

$$\mathbf{s}_0 = \mathbf{A}\mathbf{s}_0 + \mathbf{C}\mathbf{s}_1 \tag{7.152}$$

or

$$\mathbf{s}_1 = \mathbf{C}^{-1}(\mathbf{I} - \mathbf{A})\mathbf{s}_0 \tag{7.153}$$

Since we have an initial assumed value for $\mathbf{s}_0$, we now know $\mathbf{s}_1$. In general we can write the iterative expressions

$$\mathbf{s}_i = \mathbf{C}^{-1}(\mathbf{I} - \mathbf{A})\mathbf{s}_{i-1} \quad 1 \leq i \leq B \tag{7.154}$$

Having found all the vectors $\mathbf{s}_i$, we obtained the normalized distribution vector $\mathbf{s}'$ as

$$\mathbf{s}' = s/\sum_{i=0}^{n-1}\sum_{j=0}^{B} s_{i,j} \tag{7.155}$$

### 7.10.1 Performance of the $M/D/1/B$ Queue

The average input traffic $N_a(in)$ is needed to estimate the efficiency of the queue and queue delay. $N_a(in)$ is given by

$$N_a(in) = N\ a \tag{7.156}$$

The throughput of the queue is given by

$$Th = a\sum_{j=0}^{B} s_{n-1,j} \tag{7.157}$$

The efficiency of the $D/M/1/B$ queue is given by

$$\eta = \frac{Th}{N_a(in)} = \frac{1}{N} \times \sum_{j=0}^{B} s_{n-1,j} \tag{7.158}$$

Packets are lost in the $M/D/1B$ queue when the queue is full and a packet arrives but does not leave. The average lost traffic is given by

$$N_a(lost) = a\sum_{i=0}^{n-2} s_{i,B} \tag{7.159}$$

The packet loss probability $L$ is given by

$$L = \frac{N_a(lost)}{N_a(in)} = \frac{1}{N}\sum_{i=0}^{n-2} s_{i,B} \tag{7.160}$$

The average queue size is given by

$$Q_a = \sum_{i=0}^{n-1}\sum_{j=0}^{B} i\ s_{i,j} \tag{7.161}$$

## 7.11 Systems of Communicating Markov Chains

In the previous sections we investigated the behavior of single Markov chains or single queues. More often than not communication systems are composed of interconnected or interdependent Markov chains or queues. As an example, let us consider Fig. 7.10. This system represents an entity that is connected to a bus or a communication channel in general. This could be the Ethernet network interface card (NIC) on a computer connected to a local area network. The system to be analyzed consists of:

1. traffic source
2. packet or transmit buffer
3. medium access module

Each one of these components can be modeled individually using Markov chain analysis. For example, the traffic source can be modeled using the techniques in Chap. 15. The transmit buffer is of course modeled using any of the queuing models discussed in this chapter. The medium access module could be modeled using the techniques discussed in Chap. 10.

Let us assume that the Markov models for the three components are characterized by $s_1$ states for the traffic source, $s_2$ states for the queue, and $s_3$ states for the medium access module. We could develop a unified Markov model for our system and this would probably require $s_1 \times s_2 \times s_3$ states. This is the *state explosion problem* associated with most Markov chain models. If we, on the other hand, treat the problem as a system of dependent Markov chains, then we are in effect dealing with the individual components and the total number of states is $s_1 + s_2 + s_3$. Of course, we cannot solve each system separately since the state of each component depends on the other components communicating with it.

We can generalize the problem by considering a communicating Markov chain system composed of $n$ modules. Module $i$ is characterized by four quantities:

1. State transition matrix $\mathbf{P}_i$
2. Steady state distribution vector $\mathbf{s}_i$
3. Probabilities $x_i$ corresponding to the module inputs
4. Probabilities $y_i$ corresponding to the module outputs

The parameters of each transition matrix $\mathbf{P}_i$ $(0 \leq i < n)$ depend in general on the values of $\mathbf{s}_j$, $x_j$ and $y_j$ of all the other modules or systems. Furthermore, the module inputs and outputs $x_i$ and $y_i$ might also depend on the queue parameters such as $\mathbf{s}_j$. In this analysis, we assume that $x_i$ and $y_i$ are independent variables for simplicity. Figure 7.11 shows the system of $n$ interdependent Markov chains.

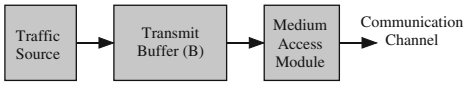



**Fig. 7.10** A system of several Markov chains or queues

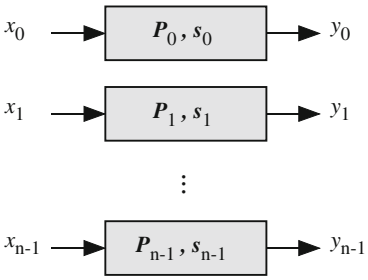



**Fig. 7.11** Model of a network of interdependent Markov chains

If we attempt to model the system in Fig. 7.11 as a single Markov chain, the number of states we have to deal with would be given by

$$\text{Merged number of states } = \Pi_{i=0}^{n-1} K_i$$

where $K_i$ is the number of states of module $i$. If all the modules had the same number of states ($K$), then the total number of states would have been

$$\text{Merged number of states } = K^n$$

If we deal with the system as a system of communicating and dependent Markov chains, then the total number of states we have to deal with would be given by

$$\text{Number of states } = \sum_{i=0}^{n-1} K_i$$

where $K_i$ is the number of states of module $i$. If all the modules had the same number of states ($K$), then the total number of states would have been

$$\text{Number of states } = nK$$

For typical systems, the quantity $nK$ is much smaller than $K^n$. As an example, assume a modest system where each module has $K = 10$ states and we have $n$ modules where $n = 5$. The number of states using the two approaches yield

$$\text{Number of states } = K^n = 10^5 = 100{,}000$$

$$\text{Number of states } = nK = 50$$

The situation is even worse had $K = 100$ where our states would have been ten billion using the former approach compared to 500 using the latter.

This is one reason that researchers and engineers alike attempt to use Petri Nets to describe typical systems [6, 7]. However, Petri Nets are basically graphical tools. Our approach in this section could be very loosely considered as approaching Petri Nets through establishing the communicating channels and dependencies between the different modules of the system.

For each module in the system we can write the steady-state equation

$$\mathbf{P}_i(\mathbf{s}_j, x_j, y_j) \times \mathbf{s}_i = \mathbf{s}_i \quad 0 \leq i, j < n \tag{7.162}$$

where the notation $\mathbf{P}_i(\mathbf{s}_j, x_j, y_j)$ indicates that $\mathbf{P}_i$ is a function of $\mathbf{s}_j$, $x_j$ and $y_j$.

*Example 7.12.* Assume a system of two interdependent queues such that the first queue is a simple $M/M/1/B$ queue with parameters: arrival probability $a$, departure probability $c$, and size $B = 2$. The second queue depends on the first queue as indicated by the state transition matrix

$$\mathbf{Y} = \begin{bmatrix} \alpha & 1 - x_0 \\ 1 - \alpha & x_0 \end{bmatrix}$$

where $\alpha < 0$ and $x_0$ is the probability that the first queue is empty. Explain how the steady state distribution vectors $\mathbf{x}$ and $\mathbf{y}$ of the two queues could be obtained.

We have for the first queue

$$\mathbf{X} = \begin{bmatrix} 1 - ad & bc & 0 \\ ad & f & bc \\ 0 & ad & 1 - bc \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_0 & x_1 & x_2 \end{bmatrix}^t$$

where $f = ac + bd$. Since the first queue is not dependent on the second queue, the steady state distribution vector is found using Eq. (7.17)

$$\mathbf{x} = \frac{1 - \rho}{1 - \rho^3} \times \begin{bmatrix} 1 & \rho & \rho^2 \end{bmatrix}^t$$

where $\rho = ad/bc$.

For the second queue we have

$$\mathbf{Y} = \begin{bmatrix} \alpha & 1 - x_0 \\ 1 - \alpha & x_0 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_0 & y_1 \end{bmatrix}^t$$

Matrix $\mathbf{Y}$ is determined since we know the value of $x_0$. Thus we can find the value of $\mathbf{y}$ using any of the techniques that we studied in Chap. 4. ■

### 7.11.1 A General Solution for Communicating Markov Chains

In the general case, a simple solution for the system is not possible due to the complexity of the transition matrices. The general steps we recommend to employ can be summarized as follows:

**Step 0: Initialization of $\mathbf{s}_i$**

Set initial nonzero values for all the distribution vectors $\mathbf{s}_i(0)$ for all $0 \leq i < n$. Of course each vector must be normalized in the sense that the sum of its components must always be 1. In general, at iteration $k$ the estimated value of $\mathbf{s}_i(k)$ is known. The notation $\mathbf{s}_i(k)$ indicates the value of $\mathbf{s}_i$ at iteration $k$.

**Step 1: Estimating $\mathbf{P}_i$**

The elements of the transition matrices $\mathbf{P}_i(k)$ at iteration $k$ can now be estimated since $\mathbf{s}_i(k)$, $x_i$ and $y_i$ are known. Initially, $k = 0$ to start our iterations and we are then able to calculate the initial $\mathbf{P}_i(0)$.

**Step 2: Calculating $\mathbf{s}_i$**

Knowing $\mathbf{P}_i(k)$ we can now calculate the values $\mathbf{s}_{i,c}(k)$, where the notation $\mathbf{s}_{i,c}(k)$ indicates the calculated value of $\mathbf{s}_i(k)$ which will most probably be different from the assumed $\mathbf{s}_i(k)$. We calculate $\mathbf{s}_{i,c}(k)$ using the equation

$$\mathbf{s}_{i,c}(k) = \mathbf{P}_i(k)\, \mathbf{s}_i(k) \tag{7.163}$$

Again, initially $k = 0$ at the start of iterations.

**Step 3: Calculating the Estimation Error $\mathbf{e}_i(k)$**

The calculated value $\mathbf{s}_{i,c}(k)$ will not be equal to the values $\mathbf{s}_i(k)$. The estimation error vector is calculated as

$$\mathbf{e}_i(k) = \mathbf{s}_{i,c}(k) - \mathbf{s}_i(k) \tag{7.164}$$

The magnitude of the estimation error for $\mathbf{s}_i(k)$ is given by

$$\epsilon_i(k) = \sqrt{\mathbf{e}_i^t(k)\, \mathbf{e}_i(k)} \tag{7.165}$$

**Step 3: Updating Value of $\mathbf{s}_i(k)$**

We are now able to update our guess of the state vectors and obtain new and better values $\mathbf{s}_i(k+1)$ according to the update equations

$$\mathbf{s}_i(k+1) = \mathbf{s}_i(k) + \alpha \mathbf{e}_i(k) \qquad 0 \leq i < n \tag{7.166}$$

where $\alpha \ll 1$ is a small correction factor to ensure smooth convergence.

**Step 4: Improving the Estimated Values of $\mathbf{s}_i(k)$**

We repeat steps 1–3 with the new values $\mathbf{s}_i(k+1)$ until the total error measure $\epsilon_t$ is below an acceptable level

$$\epsilon_t = \sum_{i=0}^{n} \epsilon_i \leq \gamma \tag{7.167}$$

where $\gamma$ is the acceptable error threshold.

We are "confident" that convergence will take place since at each iteration the matrices $\mathbf{P}_i(k)$ are all column stochastic and their eigenvalues satisfy the inequality $\lambda \leq 1$.

## 7.12 Problems

### *Throughput and Efficiency*

**7.1.** Consider a switch that generates its own traffic, $N_a(internal)$, in addition to the traffic arriving at its input, $N_a(in)$, according to the discussion in Sect. 7.3. Define the throughput and the efficiency for this system in terms of $N_a(in)$, $N_a(internal)$, and $N_a(out)$, such that $\eta$ never exceeds unity.

### *M/M/1 Queue*

**7.2.** Prove that (7.13) on page 237 is true when the $M/M/1$ queue is stable.

**7.3.** Consider an $M/M/1$ with a distribution vector $\rho$ very close to unity such that

$$\rho = 1 - \epsilon$$

where $\epsilon \ll 1$. Find the equilibrium distribution vector.

**7.4.** Consider an $M/M/1$ queue with arrival probability $a = 0.5$ and departure probability $c = 0.6$.

(a) Construct the first six rows and columns of the transition matrix $\mathbf{P}$.
(b) Find the values of the first 10 components of the equilibrium distribution vector.
(c) Calculate the queue performance.

**7.5.** Repeat Problem 7.4 when the departure probability becomes almost equal to the arrival probability (e.g. $c = 0.55$).

**7.6.** Consider an $M/M/1$ queue with arrival probability $a = 0.1$ and departure probability $c = 0.5$.

(a) Construct the first six rows and columns of the transition matrix $\mathbf{P}$.
(b) Find the values of the first ten components of the equilibrium distribution vector.
(c) Calculate the queue performance.

**7.7.** Repeat Problem 7.6 when the departure probability becomes almost equal to the arrival probability (e.g., $c = 0.11$).

**7.8.** In an $M/M/1$ queue it was found out that the average queue size $Q_a = 5$ packets and the average waiting time is $W = 20$ time steps. Calculate the queue arrival and departure probabilities and find the first ten entries of the distribution vector.

**7.9.** In an $M/M/1$ queue it was found out that the average queue size $Q_a = 2$ packets and the average waiting time is $W = 100$ time steps. Calculate the queue arrival and departure probabilities and find the first ten entries of the distribution vector.

**7.10.** Equation (7.7) describes the $M/M/1$ queue when a packet could be served in the same time step at which it arrives. Suppose that an arriving packet cannot be served until the next time step. What will be expression for the state matrix? Compare your result to (7.7).

**7.11.** Derive the performance for the $M/M/1$ queue described in Problem 7.10.

**7.12.** In the $M/M/1$ queue in Problem 7.10 it was found out that the average queue size $Q_a = 2$ packets and the average waiting time is $W = 100$ time steps. Calculate the queue arrival and departure probabilities and find the first ten entries of the distribution vector.

## *M/M/1/B Queue*

**7.13.** Prove the average queue size formula for the $M/M/1/B$ queue is given in (7.47) on page 245.

**7.14.** Consider an $M/M/1/B$ queue with arrival probability $a = 0.5$, departure probability $c = 0.6$, and maximum queue size $B = 4$.

(a) Construct the transition matrix **P**.
(b) Find the values of the components of the equilibrium distribution vector.
(c) Calculate the queue performance.
(d) Compare your results with those of the $M/M/1$ queue in Problem 7.4 having the same arrival and departure probabilities.

**7.15.** Repeat Problem 7.14 when the departure probability becomes almost equal to the arrival probability (e.g., $c = 0.55$). Compare your results with those of the $M/M/1$ queue in Problem 7.5 having the same arrival and departure probabilities.

**7.16.** Repeat Problem 7.14 when the departure probability actually exceeds the arrival probability (e.g., $c = 0.8$). Compare your results with those of the $M/M/1$ queue in Problem 7.5 having the same arrival and departure probabilities.

**7.17.** Consider an $M/M/1/B$ queue with arrival probability $a = 0.1$, departure probability $c = 0.5$, and maximum queue size $B = 5$.

(a) Construct the transition matrix **P**.
(b) Find the values of the equilibrium distribution vector.
(c) Calculate the queue performance.
(d) Compare your results with those of the $M/M/1$ queue in Problem 7.6 having the same arrival and departure probabilities.

**7.18.** Repeat Problem 7.17 when the departure probability becomes almost equal to the arrival probability (i.e., $c = 0.11$). Compare your results with those of the $M/M/1$ queue in Problem 7.7 having the same arrival and departure probabilities.

**7.19.** Equation (7.30), on page 242, describes the $M/M/1/B$ queue when a packet could be served in the same time step at which it arrives. Suppose that an arriving packet cannot be served until the next time step. What will be expression for the state matrix? Compare your result to (7.30).

**7.20.** Derive the performance for the $M/M/1/B$ queue described in Problem 7.19.

**7.21.** Consider an $M/M/1/B$ queue with arrival probability $a = 0.4$, departure probability $c = 0.39$, and maximum queue size $B = 5$. The queue is not stable since the arrival probability is larger than the departure probability.

(a) Construct the transition matrix **P**.
(b) Find the values of the equilibrium distribution vector.
(c) Calculate the queue performance.

**7.22.** In the $M/M/1$ queue in Problem 7.20 it was found out that the average queue size $Q_a = 2$ packets and the average waiting time is $W = 100$ time steps. Calculate the queue arrival and departure probabilities and find the first ten entries of the distribution vector.

## $M^m/M/1/B$ Queue

**7.23.** Write down the entries for the transition matrix of an $M^m/M/1/B$ queue in terms of the arrival and departure statistics $a_k$ and $c$, respectively. Consider the case when $B = 4$.

**7.24.** Equation (7.63), on page 250, describes the $M^m/M/1/B$ queue when up to $m$ packets could arrive at one time step. Prove that for the special case when $m = 1$, (7.63) becomes identical to (7.30).

**7.25.** Equation (7.63) describes the $M^m/M/1/B$ queue when a packet could be served in the same time step at which it arrives. Suppose that an arriving packet cannot be served until the next time step. What will be expression for the state matrix? Compare your result to (7.63).

**7.26.** Consider an $M^m/M/1/B$ queue with arrival probability $a = 0.2$, $m = 2$, and departure probability $c = 0.39$ and maximum queue size $B = 5$. The queue is not stable since the average number of arrivals is larger than the average number of departures.

(a) Construct the transition matrix **P**.
(b) Find the values of the equilibrium distribution vector.
(c) Calculate the queue performance.

## $M/M^m/1/B$ Queue

**7.27.** Write down the entries for the transition matrix of an $M/M^m/1/B$ queue in terms of the arrival and departure statistics $a$ and $c_k$, respectively. Consider the case when $B = 4$.

**7.28.** Equation (7.95), on page 257, describes the $M/M^m/1/B$ queue when up to $m$ packets could leave at one time step. Prove that for the special case when $m = 1$, (7.95) becomes identical to (7.30).

**7.29.** Equation (7.95) describes the $M^m/M/1/B$ queue when a packet could be served in the same time step at which it arrives. Suppose that an arriving packet cannot be served until the next time step. What will be expression for the state matrix? Compare your result to (7.95).

**7.30.** Consider an $M/M^m/1/B$ queue with arrival probability $a = 0.5$, departure probability $c = 0.2$, $m = 2$, and maximum queue size $B = 5$. The queue is not stable since the average number of arrivals is larger than the average number of departures.

(a) Construct the transition matrix **P**.
(b) Find the values of the equilibrium distribution vector.
(c) Calculate the queue performance.

**7.31.** In the analysis of the $M/M^m/1$ queue it was assumed that when a packet arrives, it can serviced at the same time step. Suppose the arriving packet is serviced in the next time step. Draw the state transition diagram and the write down the corresponding transition matrix. Derive the main performance equations of such a queue.

## References

1. F. Elguibaly, Analysis and design of arbitration protocols. IEEE Trans. Comput. **38**(2), 1168–1175 (1989)
2. D.G. Kendall, Stochastic processes occurring in the theory of queues and their analysis by means of the embedded Markov chain. Ann. Math. Stat. **24**, 338–354 (1953)
3. M.E. Woodward, *Communication and Computer Networks* (IEEE Computer Society Press, Los Alamitos, 1994)
4. I. Peterson, *Fatal Defect: Chasing Killer Computer Bugs* (Random House, New York, 1995)
5. W.J. Stewart, *Introduction to Numerical Solutions of Markov Chains* (Princeton University Press, Princeton, 1994)
6. J. Billington, M. Diaz, G. Rozenberg (eds.), *Application of Petri Nets to Communication Networks: Advances in Petri Nets* (Springer, New York, 1999)
7. K. Jensen, *Coloured Petri Nets: Basic Concepts, Analysis Methods, and Practical Use* (Springer, New York, 1997)

# Chapter 8
# Modeling Traffic Flow Control Protocols

## 8.1 Introduction

In this chapter we illustrate how to develop queuing models for three protocols for traffic management:

1. The leaky bucket algorithm
2. The token bucket algorithm
3. Virtual scheduling algorithm (VS)

Modeling a protocol or a system is just like designing a digital system, or any system for that mater. There are many ways to model a protocol based on the assumptions that one makes. My motivation here is simplicity and not taking a guided tour through the maze of protocol modeling. My recommendation to the reader is to read the discussion on each protocol then lay down the outline of a model that describes the protocol. The model or models developed here should then be compared with the one attempted by the reader.

## 8.2 The Leaky Bucket Algorithm

Computer traffic is seldom uniform and is characterized by periods of burstiness. Traffic bursts tax the network resources such as switch buffers and lead to network congestion and data loss. Because it is impossible for the network to accept only uniform traffic, mechanisms have been proposed to regulate or smooth out these bursts.

Thus traffic shaping, also known as traffic policing, aims at regulating the average rate of traffic flow even in the presence of occasional bursts [1]. This helps manage the congestion problem at the switches.

When a user accesses the network, the important parameter to describe the traffic is the average data rate ($\lambda_a$). This is estimated by observing the number of packets



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_8

($N$) sent over a long time interval $t$ and finding the average data rate as $\lambda_a = N/t$. This rate is compared to a maximum rate ($\lambda_b$) that is specified by the leaky bucket algorithm. As long as $\lambda_a < \lambda_b$, the user is classified as conforming and data is accepted. Users that obey this traffic contract are termed *conforming users*, while users that violate this contract are termed *violating* or *nonconforming users*. Traffic policing to ensure that each user is conforming is done at the points where users access the network (ingress points).

Leaky bucket is a *rate-based* algorithm for controlling the *maximum rate* of traffic arriving from a source. If the input data rate is less than the maximum rate specified by the algorithm, leaky bucket accepts the data. If the input data rate exceeds the maximum rate, leaky bucket passes the data at the maximum rate and excess data is buffered. If the buffer is full, then excess data is discarded. In our modeling of the leaky bucket algorithm we are interested only in the state of the data buffer since the state of that buffer dictates the actions to be done on the incoming traffic. Our aim then is to simply model the buffer state so that we are able to predict the performance of the algorithm

Figure 8.1a shows the leaky bucket buffer. The buffer accepts incoming data and releases the stored packets at a data rate that does not exceed $\lambda_b$. Depending on the data arrival rate $\lambda_a$ and the state of occupancy of the packet buffer, the following scenarios could take place.

1. $\lambda_a < \lambda_b$: Data arrive at a rate ($\lambda_a$) lower than the maximum rate ($\lambda_b$) specified by the leaky bucket algorithm. In that case data will be accepted as shown by the arrival of packets 1 and 2 in Fig. 8.1b. The long time interarrival time between packets 1 and 2 indicates a low data arrival rate. We assumed in the figure that the maximum departure rate $\lambda_b$ is equivalent to three time steps between packets. As soon as these packets arrive, they are passed through by the leaky bucket algorithm.
2. $\lambda_a > \lambda_b$: Data arrive at a rate higher than $\lambda_b$ and the data buffer is not full. In that case data will be buffered so it can be issued at the maximum rate $\lambda_b$. This

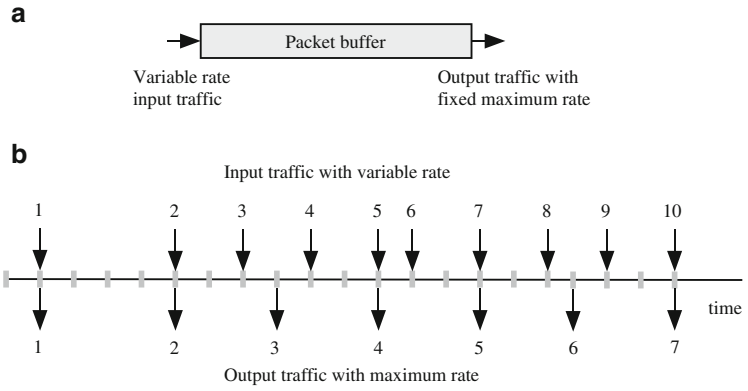



**Fig. 8.1** The leaky bucket algorithm smoothes input variable rate traffic by buffering it and regulating the maximum buffer output traffic rate. (**a**) Diagram of packet buffer. (**b**) Packet arrival and departure

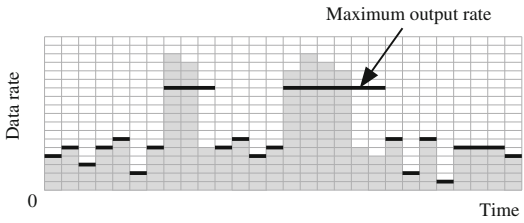



**Fig. 8.2** Control of data rate by the leaky bucket algorithm. Data rate at the input is indicated by the *grey areas* and data rate at the output is indicated by the *black lines*

is shown by arrival of packets 3, 4, and 5 in Fig. 8.1b. Note that packets 3, 4, and 5 are arrived close together in time at the input indicating a high input data rate. The interarrival time is 2 time steps which indicates higher data rate than $\lambda_b$ which is equivalent to two time steps. At the output, the spacing between these packets is equivalent to data transmitted at the rate $\lambda_b$.

3. $\lambda_a > \lambda_b$: Data arrive at a rate higher than $\lambda_b$ and the data buffer is full. In that case data will be discarded or labeled as nonconforming. This is shown by arrival of packets 6 and 7 in Fig. 8.1b.

Figure 8.2 shows the variation of output data rate in relation to the input data rate. Grey areas indicate input data rate and solid black lines indicate maximum data rate $\lambda_b$. Output data rate does not exceed $\lambda_b$ and any excess input traffic is buffered or lost. The figure shows two occasions when the input data rate exceeds $\lambda_b$. When this situation happens excess data is buffered and then released when the output data rate becomes low again.

### *8.2.1 Modeling the Leaky Bucket Algorithm*

In this section we perform Markov chain analysis of the leaky bucket algorithm. The states of the Markov chain represent the number of packets stored in the leaky bucket buffer.

Packets arrive at the input of the buffer at a rate $\lambda_{in}$ which varies with time because of the burstiness of the source. To model the source burstiness in a simple manner, we assume the data source has the following parameters:

$\lambda_a$ Average data rate of source.
$\sigma$ Source burst rate when it is nonconforming.

$\lambda_a$ and $\sigma$ typically satisfy the relations

$$\lambda_a < \lambda_b \tag{8.1}$$

$$\sigma > \lambda_b \tag{8.2}$$

where $\lambda_b$ is the maximum data departure rate as determined by the leaky bucket algorithm.

On the other hand, packets leave the buffer with an output rate $\lambda_{out}$ given by

$$\lambda_{out} = \begin{cases} \min(\lambda_{in},\ \lambda_b) & \text{When packet buffer is empty} \\ \lambda_b & \text{When packet buffer is not empty} \end{cases} \tag{8.3}$$

Notice that the output data rate is governed by the state of the data buffer and not by the input data rate.

The leaky bucket algorithm can be modeled using two different types of queues depending on our choice of the time step. These two approaches are explained in the following two sections.

### 8.2.2 Single Arrival/Single Departure Model ($M/M/1/B$)

In this approach to modeling the leaky bucket algorithm we take the time step equal to the inverse of the maximum data rate on the line.

$$T = \frac{1}{\lambda_l} \tag{8.4}$$

where the time step value is measured in units of seconds and $\lambda_l$ is the *maximum* input line rate (in units of packets/s) such that

$$\lambda_b < \lambda_l$$

The above inequality is true since the line is shared by many users. The time $T$ is the time between packet arrivals at the maximum allowable rate on the input line. When $\lambda_l$ is specified in units of bits per second (bps), $T$ is obtained as

$$T = \frac{A}{\lambda_l} \tag{8.5}$$

where $A$ is the average packet length.

Figure 8.3 shows the events of packet arrival and departure and also the time step value as indicated by the spacing between the grey tick marks.

At a given time step a maximum of one packet could arrive at or leave the buffer. The packet arrival probability ($a$) is given by studying the number of arriving packets in a time $t$. The average number of packets arriving in this time frame is

$$N(in) = \lambda_a\, t \tag{8.6}$$

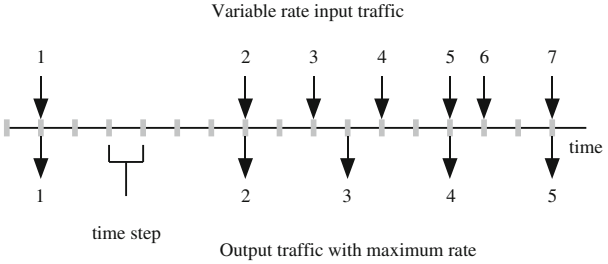



**Fig. 8.3** The leaky bucket algorithm where the time step is chosen equal to the inverse of the maximum line rate

But during this time period, we have $N$ time steps with $N = t/T$. The average number of arriving packets is estimated also using the binomial distribution as

$$N(in) = a\ N = a\ \frac{t}{T} \tag{8.7}$$

From the above two equations we get

$$a = \lambda_a\ T = \frac{\lambda_a}{\lambda_l} \tag{8.8}$$

Of course, when the source is conforming, the departure probability is $c = 1$. Using a similar argument, the minimum packet departure probability ($c$) is given by

$$c = \frac{\lambda_{out}}{\lambda_l} \tag{8.9}$$

Therefore, we have a single-input, single-output data buffer whose size is assumed $B$. The queue we are studying becomes $M/M/1/B$ queue and the transition matrix is $(B+1) \times (B+1)$ and is given by

$$\mathbf{P} = \begin{bmatrix} f_0 & bc & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ ad & f & bc & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & ad & f & bc & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & ad & f & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & f & bc & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & ad & f & bc & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & ad & f & bc \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & ad & 1-bc \end{bmatrix} \tag{8.10}$$

where $b = 1 - a$, $d = 1 - c$, $f_0 = 1 - ad$, and $f = 1 - ad - bc$.

### 8.2.3 Leaky Bucket Performance ($M/M/1/B$ Case)

Having obtained the transition matrix, we are able to calculate the performance of the leaky bucket protocol.

The throughput of the leaky bucket algorithm when the $M/M/1/B$ model is used is given from Sect. 7.6 on page 241 by

$$Th = c\ (1 - b\ s_0) \tag{8.11}$$

The throughput is measured in units of packets/time step. The throughput in units of packets/second is expressed as

$$Th' = \frac{Th}{T} = Th \times \frac{\lambda_l}{A} \tag{8.12}$$

where we assumed $\lambda_l$ was given in units of bits/s.

The average number of lost or tagged packets per time step is obtained using the results of the $M/M/1/B$ queue

$$N_a(lost) = s_B\ a\ d \tag{8.13}$$

The lost traffic is measured in units of packets/time step. And the number of packets lost per second is

$$\begin{aligned} N'_a(lost) &= \frac{N_a(lost)}{T} \\ &= N_a(lost) \times \frac{\lambda_l}{A} \end{aligned} \tag{8.14}$$

where we assumed $\lambda_l$ was given in units of bits/s. The packet loss probability is given by

$$L = \frac{N_a(lost)}{N_a(in)} = \frac{s_B\ a\ d}{\lambda_a} \tag{8.15}$$

The average queue size is given by

$$Q_a = \sum_{i=0}^{B} i\ s_i \tag{8.16}$$

where $s_i$ is the probability that the data buffer has $i$ packets.

Using Little's result, the average wait time in the buffer is

$$W = \frac{Q_a}{Th} \tag{8.17}$$

The wait time is measured in units of time steps. The wait time in units of seconds is given by

$$W' = \frac{Q_a}{Th'} \tag{8.18}$$

*Example 8.1.* A leaky bucket traffic shaper has the following parameters.

$$\begin{array}{ll} \lambda_a = 1 \text{ Mbps} & \sigma = 4 \text{ Mbps} \\ \lambda_b = 1.5 \text{ Mbps} & \lambda_l = 50 \text{ Mbps} \\ A = 400 \text{ bits} & B = 5 \text{ packets} \end{array}$$

Derive the performance of this protocol using the $M/M/1/B$ modeling approach.

The arrival probability is

$$a = \frac{\lambda_a}{\lambda_l} = 0.02$$

The minimum departure probability is

$$c = \frac{\lambda_b}{\lambda_l} = 0.03$$

We see that under the assumed traffic conditions the arrival probability is larger than the departure probability and we expect the packet buffer to be filled.

The transition matrix will be

$$\mathbf{P} = \begin{bmatrix} 0.9399 & 0.0281 & 0 & 0 & 0 & 0 \\ 0.0601 & 0.9117 & 0.0281 & 0 & 0 & 0 \\ 0 & 0.0601 & 0.9117 & 0.0281 & 0 & 0 \\ 0 & 0 & 0.0601 & 0.9117 & 0.0281 & 0 \\ 0 & 0 & 0 & 0.0601 & 0.9117 & 0.0281 \\ 0 & 0 & 0 & & 0.0601 & 0.9719 \end{bmatrix}$$

The equilibrium distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0.0121 & 0.0258 & 0.0551 & 0.1177 & 0.2516 & 0.5377 \end{bmatrix}^t$$

Since $s_5 = 0.5377$, we conclude that 53.77 % of the time the packet data buffer is full. The other performance parameters are:

$$\begin{array}{ll} Th = 0.0297 & \text{packets/time step} \\ Th' = 3.7076 \times 10^3 & \text{packets/s} \end{array}$$

$$
\begin{array}{lll}
N_a(lost) & = 0.0323 & \text{packets/timestep} \\
N'_a(lost) & = 4.0424 \times 10^3 & \text{packets/s} \\
L & = 1.0432 \times 10^{-8} & \\
Q_a & = 4.1843 & \text{packets} \\
W & = 141.0713 & \text{times steps} \\
W' & = 94.048 & \mu\text{s}
\end{array}
$$

We note that the leaky bucket is tagging or dropping 76.02 % of the incoming packets. ■

### 8.2.4 Multiple Arrival/Single Departure Model ($M^m/M/1/B$)

In this approach to modeling the leaky bucket algorithm we take the time step equal to the inverse of the maximum data rate as dictated by the leaky bucket algorithm for that particular source.

$$T = \frac{1}{\lambda_b} \tag{8.19}$$

where the time step is measured in units of seconds and the leaky bucket rate $\lambda_b$ is in units of packets/s. Usually $\lambda_b$ is specified in units of bits/s. In that case $T$ is obtained as

$$T = \frac{A}{\lambda_b} \tag{8.20}$$

where $A$ is the average packet length.

Figure 8.4 shows the events of packet arrival and departure and also the time step value as indicated by the spacing between the successive output packets.

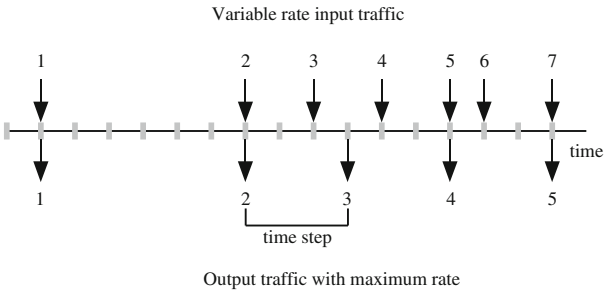



**Fig. 8.4** Events of packet arrival and departure for the leaky bucket algorithm. The time step value is equal to the time between two adjacent output packets

At a given time step one or more packets could arrive at the buffer and only one packet can leave if the buffer is not empty. Therefore, we have an $M^m/M/1/B$ queue to describe the state of the packet buffer.

The average data rate of the source as seen by the leaky bucket algorithm is given by $\lambda_a$. The maximum number of packets that could arrive at the queue input in one time step is determined by the maximum burst rate $\sigma$

$$N = \lceil \sigma \times T \rceil = \lceil \frac{\sigma}{\lambda_b} \rceil \tag{8.21}$$

where $\lceil x \rceil$ is ceiling function which produces the smallest integer that is larger than or equal to $x$.

The probability of $k$ packets arriving in one time step is given by

$$a_k = \binom{N}{k} a^k \, b^{N-k} \qquad k = 0, 1, 2, \ldots, N \tag{8.22}$$

where $a$ is the probability that a packet arrives and $b = 1 - a$.

The average number of packets arriving in one time step is estimated as

$$N_a(in) = \lambda_a \, T \tag{8.23}$$

The average input packets is estimated also using the binomial distribution as

$$N_a(in) = a \, N = a \, \lceil \sigma \, T \rceil \tag{8.24}$$

From the above two equations we get

$$a = \frac{\lambda_a \, T}{\lceil \sigma \, T \rceil} \leq \frac{\lambda_a}{\sigma} \tag{8.25}$$

Because of our choice for the time step size, the queue size can only decrease by one at most at any instant with probability $c = 1$. Assuming the packet buffer size is $B$, the transition matrix will be $(B + 1) \times (B + 1)$ and will be slightly modified from the form given by (7.63) on page 250:

$$\mathbf{P} = \begin{bmatrix} x & a_0 & 0 & 0 & 0 & \cdots & 0 \\ a_2 & a_1 & a_0 & 0 & 0 & \cdots & 0 \\ a_3 & a_2 & a_1 & a_0 & 0 & \cdots & 0 \\ a_4 & a_3 & a_2 & a_1 & a_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ a_B & a_{B-1} & a_{B-2} & a_{B-3} & a_{B-4} & \cdots & a_0 \\ z_B & z_{B-1} & z_{B-2} & z_{B-3} & z_{B-4} & \cdots & z_0 \end{bmatrix} \tag{8.26}$$

where $x = a_0 + a_1$ and

$$z_i = 1 - \sum_{j=0}^{i} a_j \tag{8.27}$$

### 8.2.5 Leaky Bucket Performance ($M^m/M/1/B$ Case)

Having obtained the transition matrix, we are able to calculate the performance of the leaky bucket protocol.

The throughput of the leaky bucket algorithm when the $M^m/M/1/B$ model is used is given from Sect. 7.7 on page 249 with a departure probability $c = 1$

$$Th = 1 - a_0\, s_0 \tag{8.28}$$

The throughput is measured in units of packets/time step and the throughput in units of packets/s is

$$Th' = Th \times \lambda_b \qquad \text{packets/s} \tag{8.29}$$

The lost or tagged packets are given by

$$\begin{aligned} N_a(lost) &= N_a(in) - N_a(out) \\ &= N\ a - (1 - a_0\, s_0) \end{aligned} \tag{8.30}$$

The lost traffic is measured in units of packets/time step. The average lost traffic per second is given by

$$\begin{aligned} N'_a(lost) &= \frac{N_a(lost)}{T} \\ &= [N\ a - (1 - a_0\, s_0)]\, \lambda_b \end{aligned} \tag{8.31}$$

The packet loss probability $l$ is the ratio of lost traffic relative to the input traffic

$$\begin{aligned} L &= \frac{N_a(lost)}{N_a(in)} \\ &= 1 - \frac{1 - a_0\, s_0}{N\ a} \end{aligned} \tag{8.32}$$

The average queue size is given by

$$Q_a = \sum_{i=0}^{B} i\ s_i \tag{8.33}$$

where $s_i$ is the probability that the data buffer has $i$ packets.

Using Little's result, the average wait time in the buffer is

$$W = \frac{Q_a}{Th} \tag{8.34}$$

The wait time is measured in units of time steps. The wait time in units of seconds is given by

$$W' = \frac{Q_a}{Th'} \tag{8.35}$$

*Example 8.2.* Repeat Example 8.1 using the $M^m/M/1/B$ modeling approach. Assume the probability the source is conforming is 0.3.

The average input data rate is

$$\lambda_a = 1 \times 0.3 + 4 \times 0.7 = 3.1 \qquad \text{Mbps}$$

$N$ is found as

$$N = \lceil \frac{\sigma}{\lambda_b} \rceil = 3$$

The packet arrival probability is

$$a = \lambda_a \times T = 0.6889$$

The probability that $k$ packets arrive in one time step is

$$a_k = \binom{3}{k} a^k b^{3-k} \qquad K = 0, 1, 2, 3$$

The transition matrix will be

$$\mathbf{P} = \begin{bmatrix} 0.2301 & 0.0301 & 0 & 0 & 0 & 0 \\ 0.4429 & 0.2000 & 0.0301 & 0 & 0 & 0 \\ 0.3269 & 0.4429 & 0.2000 & 0.0301 & 0 & 0 \\ 0 & 0.3269 & 0.4429 & 0.2000 & 0.0301 & 0 \\ 0 & 0 & 0.3269 & 0.4429 & 0.2000 & 0.0301 \\ 0 & 0 & 0 & 0.3269 & 0.7699 & 0.9699 \end{bmatrix}$$

The equilibrium distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0 & 0 & 0.0001 & 0.0014 & 0.0370 & 0.9615 \end{bmatrix}^t$$

We see that 96 % of the time the packet buffer is full. The other performance parameters are:

$$\begin{array}{lll} Th & = 1 & \text{packets/time step} \\ Th' & = 3.7500 \times 10^3 & \text{packets/s} \\ N_a(lost) & = 1.0667 & \text{packets/time step} \\ L & = 0.5161 & \\ Q_a & = 4.9600 & \text{packets} \\ W & = 4.96 & \text{times steps} \\ W' & = 1.3 & \text{ms} \end{array}$$

Comparing these performance figures with those obtained for the same system using the $M/M/1/B$ queue, we see that the packet throughput and delay in the system are approximately similar. One possible reason for the small variation is the use of the ceiling function in the $M^m/M/1/B$ analysis. ■

## 8.3 The Token Bucket Algorithm

Token bucket is a *credit-based* algorithm for controlling the *volume* of traffic arriving from a source. Tokens are issued at a constant rate and arriving packets can leave the system only if there are tokens available in the token buffer. Figure 8.5 shows the token bucket algorithm as it applies to a packet buffer. Figure 8.5a shows the input data buffer and token buffer. Input data arrive with a variable rate while the tokens arrive with a constant rate. There is mutual coupling or feedback between the two buffers such that both the tokens and the packets depart at the same rate which varies depending on the states of both queues.

Figure 8.5b shows the events of packet arrival and departure as well as token arrival (indicated by the grey circles). Depending on the data arrival rate and the state of occupancy of the token buffer, the following scenarios could take place.

1. No data arrive and tokens are stored in the buffer and count as credit for the source. This is similar to the situation before the arrival of packet 1 in Fig. 8.5b.
2. Data arrive at a rate lower than the token issue rate. In that case data will be accepted and the excess tokens will be stored in the token buffer as credit since the source is conforming. This is similar to the arrival of packets 1, 2, and 3 in Fig. 8.5b. More token arrive than data and token buffer starts filling up.
3. Data arrive at a rate higher than the token issue rate. In that case data will be accepted as long as there are tokens in the buffer. This is similar to the arrival of packets 4 and 5 in Fig. 8.5b. Packets 4 and 5 go through although no tokens have arrived because the token buffer was not empty. When the token buffer becomes empty, data will be treated as in the following step.
4. Data arrive but no tokens are present. Only a portion of the data will be accepted at a rate equal to the token issue rate. The excess data can be discarded or tagged

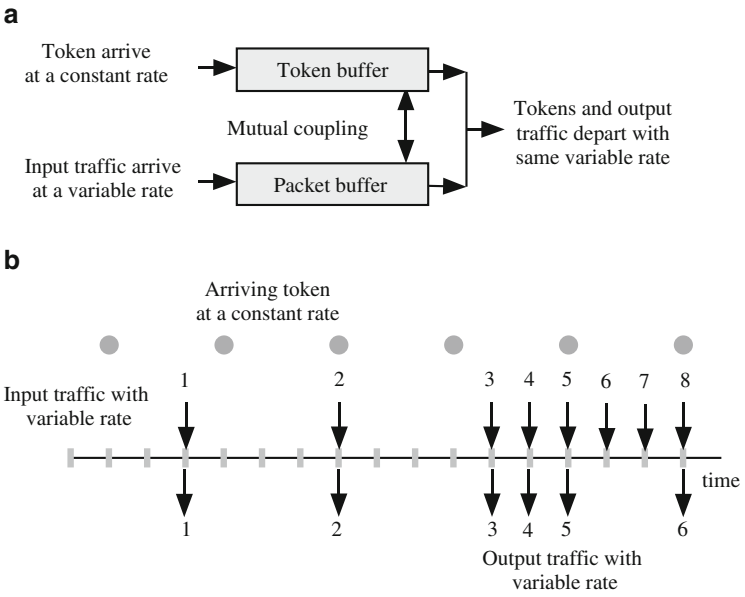



**Fig. 8.5** The token bucket algorithm smoothes input variable rate traffic by buffering it and regulating the maximum buffer output traffic rate. (**a**) Diagram of packet buffer. (**b**) Packet arrival and departure

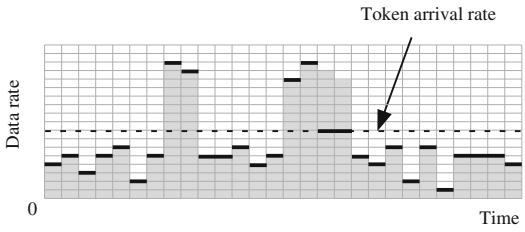



**Fig. 8.6** Control of data rate by the token bucket algorithm. Data rate at the input is indicated by the *grey areas*, data rate at the output is indicated by the *black lines* and the *dashed line* is the token issue rate

as nonconforming. This is similar to the arrival of packets 6, 7, and 8 in Fig. 8.5b. Packets 6 and 7 arrive when the token buffer is empty and do not go through. The packets get stored in the packet buffer and when a token arrives packet 6 goes through.

Figure 8.6 shows the variation of output data rate in relation to the input data rate. Grey areas indicate input data rate and solid black lines indicate output data rate. The dashed line in the middle of the figure indicates the token arrival rate. We see that output data rate can temporarily exceed the token rate but only for a short time. The duration of this burst depends on the amount of tokens stored in the token buffer. Bigger token buffer size allows for longer bursts from the source. However, a bursty source will not allow the token buffer enough time to fill up.

On the other hand, the packet buffer allows for temporary storage of arriving packets when there are no tokens in the token buffer.

### 8.3.1 Modeling the Token Bucket Algorithm

In this section we perform Markov chain analysis of the token bucket algorithm. The states of the Markov chain represent the number of tokens stored in the token buffer or bucket and the number of packets stored in the packet buffer.

We cannot model the token buffer separately from the packet buffer since the departure from one buffer depends on the state of occupancy of the other buffer. In a sense we have two mutually coupled queues as was shown in Fig. 8.5a.

The token bucket algorithm can be modeled using two different types of queues depending on our choice of the time step. These two approaches are explained in the following two sections.

### 8.3.2 Single Arrival/Single Departure Model ($M/M/1/B$)

In this approach to modeling the token bucket algorithm we take the time step equal to the inverse of the maximum data rate on the line

$$T = \frac{1}{\lambda_l} \tag{8.36}$$

where $T$ is measured in units of seconds and $\lambda_l$ is the *maximum* input line rate in units of packets/s. Usually $\lambda_l$ is specified in units of bps. In that case $T$ is obtained as

$$T = \frac{A}{\lambda_l} \tag{8.37}$$

where $A$ is the average packet length.

Figure 8.7 shows the events of packet arrival and departure and also the time step value as indicated by the spacing between the grey tick marks.

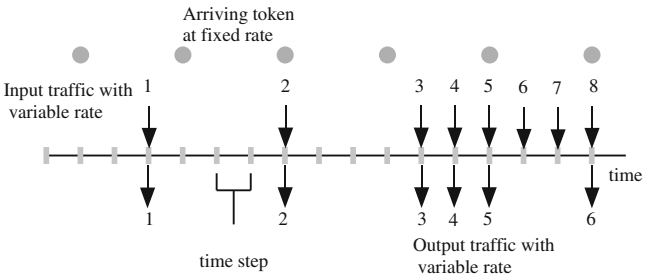



**Fig. 8.7** The token bucket algorithm where the time step is chosen equal to the inverse of the maximum line rate

Packets arrive at the input of the buffer at a rate $\lambda_a$ which varies with time because of the burstiness of the source. To model the source burstiness in a simple manner, we assume the data source has the following parameters:

$\lambda_a$ Source average data rate.
$\sigma$ Source burst rate.

$\lambda_a$ and $\sigma$ typically satisfy the relations

$$\lambda_a < \lambda_t \tag{8.38}$$

$$\sigma > \lambda_t \tag{8.39}$$

where $\lambda_t$ is the token arrival rate.

The overall average data rate of the source as seen by the token bucket algorithm is given by $\lambda_a$. At a given time step a maximum of one token could arrive at or leave the token buffer. Thus the packet buffer is described by an $M/M/1/B$ queue. The token arrival probability ($a$) is given by

$$a = \frac{\lambda_t}{\lambda_l} \tag{8.40}$$

We also define $b = 1 - a$ as the probability that a token does not arrive.

At a given time step a maximum of one packet could arrive or leave the packet buffer. Since a token leaves the token buffer each time a packet arrives, the token departure probability $c$ is given by

$$c = \frac{\lambda_a}{\lambda_l} \tag{8.41}$$

We also define $d = 1 - c$ as the probability that a token does not leave the token buffer.

The state of occupancy of the *token buffer* depends on the statistics of token and packet arrivals as follows:

1. The token buffer will stay at the same state with probability $ac + bd$; i.e., when a token arrives and a packet arrives or when no token arrives and no packet arrives too.
2. The token buffer will increase in size by one with probability $ad$; i.e., if a token arrives but no packets arrive.
3. The token buffer will decrease in size by one with probability $bc$; i.e., if no token arrives but one packet arrives.

The state of occupancy of the *packet buffer* depends on the statistics of the token and packet arrivals as follows:

1. The packet buffer will stay at the same state with probability $ac + bd$, i.e. when a token arrives and a packet arrives or when no token arrives and no packet arrives too.

2. The packet buffer will increase in size by one with probability $bc$; i.e., if no token arrives but one packet arrives.
3. The packet buffer will decrease in size by one with probability $ad$; i.e., if a token arrives but no packets arrive.

Based on the above discussion, we have two single-input, single-output buffers. The size of the token buffer is assumed $B_t$ and the size of the packet buffer is assumed $B_p$.

Figure 8.8 shows the Markov chain transition diagram for the system comprising the token and packet buffers. The upper row represents the states of the token buffer and the lower row represents the states of the packet buffer. The transition probabilities are dictated by the token and packet arrival probabilities and the states of occupancy of the token and packet buffers. The figure shows a token buffer whose size is $B_t = 4$ and a packet buffer whose size is $B_p = 3$.

The numbering of the states is completely arbitrary and does not necessarily represent the number of tokens or packets in a buffer. The meaning of each state is shown in Table 8.1.

The transition matrix of the composite system is a $\left(B_t + B_p + 1\right) \times \left(B_t + B_p + 1\right)$ tridiagonal matrix. For the case when $B_t = 4$ and $B_p = 3$ the matrix is given by

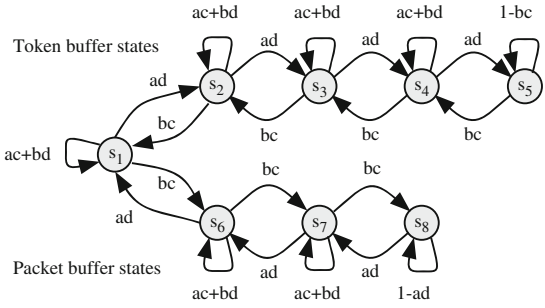



**Fig. 8.8** Markov chain transition diagram for the system comprising the token and packet buffers. Token buffer size is $B_t = 4$ and packet buffer size is $B_p = 3$

**Table 8.1** Defining the binomial coefficient

| State | Token buffer occupancy | Packet buffer occupancy |
|---|---|---|
| $s_1$ | 0 | 0 |
| $s_2$ | 1 | 0 |
| $s_3$ | 2 | 0 |
| $s_4$ | 3 | 0 |
| $s_5$ | 4 | 0 |
| $s_6$ | 0 | 1 |
| $s_7$ | 0 | 2 |
| $s_8$ | 0 | 3 |

$$\mathbf{P} = \begin{bmatrix} f & bc & 0 & 0 & 0 & ad & 0 & 0 \\ ad & f & bc & 0 & 0 & 0 & 0 & 0 \\ 0 & ad & f & bc & 0 & 0 & 0 & 0 \\ 0 & 0 & ad & f & bc & 0 & 0 & 0 \\ 0 & 0 & 0 & ad & 1-bc & 0 & 0 & 0 \\ bc & 0 & 0 & 0 & 0 & f & ad & 0 \\ 0 & 0 & 0 & 0 & 0 & bc & f & ad \\ 0 & 0 & 0 & 0 & 0 & 0 & bc & 1-ad \end{bmatrix} \tag{8.42}$$

where $b = 1 - a$, $d = 1 - c$ and $f = ac + bd = 1 - ad - bc$.

### 8.3.3 Token Bucket Performance ($M/M/1/B$ Case)

Having obtained the transition matrix, we are able to calculate the performance of the token bucket protocol.

The throughput of the token bucket algorithm is the average number of packets per time step that are produced without being tagged or lost. To find the throughput we must study all the states of the combined system. It is much easier to obtain the throughput using the traffic conservation principle after we find the lost traffic.

Packets are lost or tagged for future discard if they arrive when the packet buffer is full and no token arrives at that time step. The average number of lost or tagged packets per time step is

$$N_a(lost) = b\ c\ \ s_{B_t+B_p+1} \tag{8.43}$$

The lost traffic is measured in units of packets/time step. And the number of packets lost in units of packets/second is

$$N'_a(lost) = \frac{N_a(lost)}{T} = N_a(lost)\ \lambda_l \tag{8.44}$$

The average number of packets arriving per time step is given by

$$N_a(in) = c \tag{8.45}$$

The packet loss probability is

$$\begin{aligned} L &= \frac{N_a(lost)}{N_a(in)} \\ &= \frac{b\ c\ \ s_{B_t+B_p+1}}{c} \\ &= b\ s_{B_t+B_p+1} \end{aligned} \tag{8.46}$$

The throughput of the token bucket algorithm is obtained using the traffic conservation principle

$$\begin{aligned} Th &= N_a(in) - N_a(lost) \\ &= c - b\ c\ \ s_{B_t+B_p+1} \\ &= c\left(1 - b\ s_{B_t+B_p+1}\right) \end{aligned} \tag{8.47}$$

The throughput is measured in units of packets/time step. The throughput in units of packets/second is expressed as

$$Th' = \frac{Th}{T} = Th \times \frac{\lambda_l}{A} \tag{8.48}$$

where we assumed $\lambda_l$ was given in units of bits/s.

The packet acceptance probability $p_a$ or $\eta$ of the token bucket algorithm is

$$\begin{aligned} p_a &= \frac{Th}{N_a(in)} \\ &= 1 - L \\ &= 1 - b\ s_{B_t+B_p+1} \end{aligned} \tag{8.49}$$

We remind the reader that packet acceptance probability is just another name for the efficiency of the token bucket algorithm. It merely indicates the percentage of packets that make it through that traffic regulator without getting lost or tagged.

The average queue size for the tokens is given by

$$Q_t = \sum_{i=1}^{B_t} i\ s_{i+1} \tag{8.50}$$

Notice the range of values of the state index in the above equation.

Similarly, the average queue size for the packets is given by

$$Q_p = \sum_{i=1}^{B_p} i\ s_{i+B_t+1} \tag{8.51}$$

Notice the range of values of the state index in the above equation.

Using Little's result, the average wait time or delay for the packets in the packet buffer is

$$W = \frac{Q_p}{Th} \tag{8.52}$$

The wait time is measured in units of time steps. The wait time to in second is given by

$$W' = \frac{Q_p}{Th'} \tag{8.53}$$

*Example 8.3.* Find the performance of a token bucket traffic shaper that has the following parameters, where $A$ is the average packet length. Assume the probability the source is conforming is 0.6.

$$\begin{array}{ll} \lambda_a = 10 \text{ Mbps} & \sigma = 50 \text{ Mbps} \\ \lambda_t = 15 \text{ Mbps} & \lambda_l = 100 \text{ Mbps} \\ A = 400 \text{ bits} & B_t = 2 \text{ packets} \\ B_p = 3 \text{ packets} & \end{array}$$

The token arrival probability is

$$a = \frac{\lambda_t}{\lambda_l} = 0.15$$

The average data rate at the input is

$$\lambda_a = p\,\lambda_a + (1-p)\,\sigma = 26 \qquad \text{Mbps}$$

The token departure probability is

$$c = \frac{\lambda_a}{\lambda_l} = 0.26$$

The transition matrix will be

$$\mathbf{P} = \begin{bmatrix} 0.6680 & 0.2210 & 0 & 0.1110 & 0 & 0 \\ 0.1110 & 0.6680 & 0.2210 & 0 & 0 & 0 \\ 0 & 0.1110 & 0.7790 & 0 & 0 & 0 \\ 0.2210 & 0 & 0 & 0.6680 & 0.1110 & 0 \\ 0 & 0 & 0 & 0.2210 & 0.6680 & 0.1110 \\ 0 & 0 & 0 & 0 & 0.2210 & 0.8890 \end{bmatrix}$$

The equilibrium distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0.0641 & 0.0322 & 0.0162 & 0.1276 & 0.2541 & 0.5059 \end{bmatrix}^t$$

We see that 50.59 % of the time the packet buffer is full indicating that the source is misbehaving. The other performance parameters are:

$$
\begin{array}{lll}
N_a(lost) & = 0.1118 & \text{packets/time step} \\
N'_a(lost) & = 2.7949 \times 10^4 & \text{packets/s} \\
L & = 0.7453 & \\
Th & = 0.1482 & \text{packets/time step} \\
Th' & = 3.7051 \times 10^4 & \text{packets/s} \\
p_a & = 0.57 & \\
Q_a & = 2.1533 & \text{packets} \\
W & = 14.5294 & \text{times steps} \\
W' & = 5.8118 \times 10^{-5} & \text{s}
\end{array}
$$

■

*Example 8.4.* Investigate the effect of doubling the token buffer or the packet buffer on the performance of the token bucket algorithm in the above example.

Doubling the token buffer to $B_t = 4$ or doubling the packet buffer results in the following parameters:

| Parameter | $B_t = 2, B_p = 3$ | $B_t = 4, B_p = 3$ | $B_t = 2, B_p = 6$ |
|---|---|---|---|
| $N_a(lost)$ | 0.1118 | 0.1104 | 0.1102 |
| $L$ | 0.4300 | 0.4248 | 0.4239 |
| $Th$ | 0.1482 | 0.1496 | 0.1498 |
| $p_a$ | 0.57 | 0.5752 | 0.5761 |
| $Q_a$ | 2.1533 | 2.1274 | 5.0173 |
| $W$ | 14.5294 | 14.2250 | 33.4988 |

We note that increasing the buffer size improves the system performance. However doubling the packet buffer size doubles the delay without too much improvement in throughput compared to doubling the token buffer size. ■

### 8.3.4 Multiple Arrivals/Single Departure Model ($M^m/M/1/B$)

In this approach to modeling the token bucket algorithm we take the time step equal to the inverse of the fixed token arrival rate.

$$T = \frac{1}{\lambda_t} \tag{8.54}$$

where $T$ is measured in units of seconds and $\lambda_t$ is the token arrival rate in units of packets/s. Usually $\lambda_l$ is specified in units of bps. In that case $T$ is obtained as

$$T = \frac{A}{\lambda_t} \tag{8.55}$$

where $A$ is the average packet length.

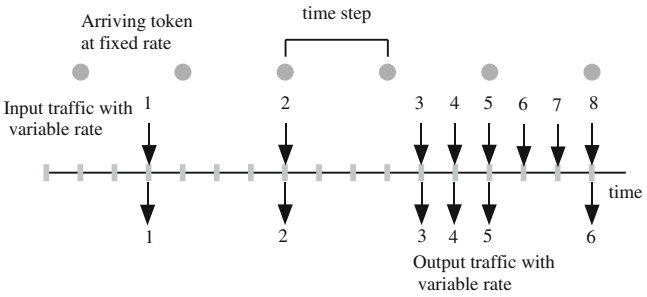



**Fig. 8.9** Events of packet arrival and departure for the token bucket algorithm. The time step value is equal to the time between two adjacent arriving tokens

Figure 8.9 shows the events of packet arrival and departure and also the time step value as indicated by the spacing between the successive tokens.

Thus at a given time step only one token arrives at the buffer and one or more tokens can leave if the buffer is not empty and data arrives. The token arrival probability per time step is given by

$$a = 1 \tag{8.56}$$

The average number of arriving packets per time step is given by

$$N_a(in) = \frac{\lambda_a}{\lambda_t} \tag{8.57}$$

where the average input data rate ($\lambda_a$) is given as before by

$$\lambda_a = p\lambda_a + (1-p)\sigma \tag{8.58}$$

where $p$ is the probability that the source is producing data at the rate $\lambda_a$ and $1-p$ is the probability that the source is producing data at the burst rate $\sigma$. $\lambda_a$ could be smaller or larger than $\lambda_t$ depending on the probability $p$.

The maximum number of packets ($N$) that could arrive at the queue input as determined by the maximum burst rate $\sigma$

$$N = \lceil \frac{\sigma}{\lambda_t} \rceil \tag{8.59}$$

with $\lceil x \rceil$ is the smallest integer that is larger than or equal to $x$. The average number of packets that arrive per time step is given from (8.57) and (8.59) according to the binomial distribution as

$$N\ x = N_a(in) \tag{8.60}$$

where $x$ is the probability of a packet arriving. The above equation gives $x$ as

$$x = \frac{N_a(in)}{N} \tag{8.61}$$

Thus we are now able to determine the packet arrival probabilities as

$$c_i = \binom{N}{i} x^i (1-x)^{L-i} \qquad i = 0, 1, 2, \ldots, N \tag{8.62}$$

The state of occupancy of the token buffer depends on the statistics of token and packet arrivals as follows:

1. The token buffer will stay at the same state with probability $c_1$; i.e., when one packet arrives.
2. The token buffer will increase in size by one with probability $c_0$; i.e., when no packets arrive.
3. The token buffer will decrease in size by one with probability $c_2$; i.e., when two packets arrive.
4. The token buffer will decrease in size by $i$ with probability $c_{i+1}$; i.e., when $i+1$ packets arrive with $i < N$

The state of occupancy of the packet buffer depends on the statistics of token and packet arrivals as follows:

1. The packet buffer will stay at the same state with probability $c_1$; i.e., when one packet arrives.
2. The packet buffer will decrease in size by one with probability $c_0$; i.e., when no packets arrive.
3. The packet buffer will increase in size by one with probability $c_2$; i.e., when two packets arrive.
4. The packet buffer will increase in size by $i$ with probability $c_{i+1}$; i.e., when $i+1$ packets arrive with $i < N$

Based on the above discussion, we have two buffers to hold the tokens and the packets. The token buffer is single-input multiple output while the packet queue is multiple input, single-output. The size of the token buffer is assumed $B_t$ and the size of the packet buffer is assumed $B_p$.

Figure 8.10 shows the Markov chain transition diagram for the system comprising the token and packet buffers. The upper row represents the states of the token buffer and the lower row represents the states of the packet buffer. The transition probabilities are dictated by the packet arrival probabilities and the states of occupancy of the token and packet buffers. The figure shows a token buffer whose size is $B_t = 4$ and a packet buffer whose size is $B_p = 3$. The maximum number of packets that could arrive in one time step is assumed $N = 3$. The figure shows the $c_0$, $c_1$, and $c_2$ transitions. The $c_3$ transitions are only shown out of states $s_4$ and $s_4$ in order to reduce the clutter.

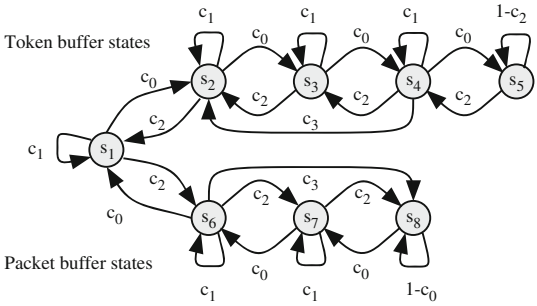



**Fig. 8.10** Markov chain transition diagram for the system comprising the token and packet buffers. Token buffer size is $B_t = 4$, packet buffer size is $B_p = 3$ and $N = 3$. The $c_3$ transitions are only shown out of states $s_4$ and $s_4$ in order to reduce the clutter

**Table 8.2** Defining the binomial coefficient

| State | Token buffer occupancy | Packet buffer occupancy |
|---|---|---|
| $s_1$ | 0 | 0 |
| $s_2$ | 1 | 0 |
| $s_3$ | 2 | 0 |
| $s_4$ | 3 | 0 |
| $s_5$ | 4 | 0 |
| $s_6$ | 0 | 1 |
| $s_7$ | 0 | 2 |
| $s_8$ | 0 | 3 |

The token and packet buffer occupancies associate with each state are shown in Table 8.2.

The transition matrix of the composite system is a $\left(B_t + B_p + 1\right) \times \left(B_t + B_p + 1\right)$ tridiagonal matrix. For the case when $B_t = 4$, $B_p = 3$, and $N = 3$ the matrix is given by

$$\mathbf{P} = \begin{bmatrix} c_1 & c_2 & c_3 & 0 & 0 & c_0 & 0 & 0 \\ c_0 & c_1 & c_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & c_0 & c_1 & c_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_0 & c_1 & c_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_0 & 1-c_2 & 0 & 0 & 0 \\ c_2 & c_3 & 0 & 0 & 0 & c_1 & c_0 & 0 \\ c_3 & 0 & 0 & 0 & 0 & c_2 & c_1 & c_0 \\ 0 & 0 & 0 & 0 & 0 & c_3 & c_2 + c_3 & 1-c_0 \end{bmatrix} \tag{8.63}$$

### 8.3.5 *Token Bucket Performance (Multiple Arrival/Departure Case)*

Having obtained the transition matrix, we are able to calculate the performance figures of the token bucket protocol.

The throughput of the token bucket algorithm is the average number of packets per time step that are produced without being tagged or lost. To find the throughput we must study all the states of the combined system. It is much easier to obtain the throughput using the traffic conservation principle after we find the lost traffic.

Packets are lost or tagged for future discard if more than one packet arrive when the packet buffer cannot accommodate all of them.

When $N < B_p$, the average number of lost or tagged packets per time step is given by

$$N_a(lost) = \sum_{i=1}^{B_p-1} s_{B_t+B_p+2-i} \sum_{j=i+1}^{N} i\ (j-1)c_j \tag{8.64}$$

The lost traffic is measured in units of packets/time step. And the number of packets lost per second is

$$N'_a(lost) = \frac{N_a(lost)}{T} = N_a(lost)\ \lambda_l \tag{8.65}$$

The average number of packets arriving per time step is given by

$$N_a(in) = \sum_{i=0}^{N} i\ c_i = N\ x \tag{8.66}$$

The packet loss probability is

$$\begin{aligned} L &= \frac{N_a(lost)}{N_a(in)} \\ &= \frac{N_a(lost)}{N\ c} \end{aligned} \tag{8.67}$$

The throughput of the token bucket algorithm is obtained using the traffic conservation principle

$$\begin{aligned} Th &= N_a(in) - N_a(lost) \\ &= N\ c - N_a(lost) \end{aligned} \tag{8.68}$$

The throughput is measured in units of packets/time step. The throughput in units of packets/second is expressed as

$$Th' = \frac{Th}{T} = Th \times \lambda_l \tag{8.69}$$

The packet acceptance probability of the token bucket algorithm is

$$\begin{aligned} p_a &= \frac{Th}{N_a(in)} \\ &= 1 - \frac{N_a(lost)}{N\ c} \end{aligned} \tag{8.70}$$

We remind the reader that packet acceptance probability is just another name for the efficiency of the token bucket algorithm. It merely indicates the percentage of packets that make it through that traffic regulator without getting lost or tagged.

The average queue size for the tokens is given by

$$Q_t = \sum_{i=1}^{B_t} i\ s_{i+1} \tag{8.71}$$

Notice the value of the state index in the above equation.

The average queue size for the packets is given by

$$Q_p = \sum_{i=1}^{B_p} i\ s_{i+B_t+1} \tag{8.72}$$

Notice the value of the state index in the above equation.

Using Little's result, the average wait time or delay for the packets in the packet buffer is

$$W = \frac{Q_p}{Th} \tag{8.73}$$

The wait time is measured in units of time steps. The wait time to in second is given by

$$W' = \frac{Q_p}{Th'} \tag{8.74}$$

*Example 8.5.* Repeat Example 8.3 using the multiple arrival/departure modeling approach.

The maximum number of packets that could arrive in one time step $m$ is found as

$$N = \lceil \frac{\sigma}{\lambda_{out}} \rceil = 4$$

The average input data rate is

$$\lambda_a = 10 \times 0.6 + 50 \times 0.4 = 26 \qquad \text{Mbps}$$

The packet arrival probability is

$$x = \frac{26}{100} = 0.4333$$

The probability that $k$ packets arrive in one time step is

$$c_k = \binom{4}{k} a^k b^{4-k} \qquad k = 0, 1, 2$$

The transition matrix will be $6 \times 6$ and is given by

$$\mathbf{P} = \begin{bmatrix} 0.3154 & 0.3618 & 0.1844 & 0.1031 & 0 & 0 \\ 0.1031 & 0.3154 & 0.3618 & 0 & 0 & 0 \\ 0 & 0.1031 & 0.4184 & 0 & 0 & 0 \\ 0.3618 & 0.1844 & 0.0353 & 0.3154 & 0.1031 & 0 \\ 0.1844 & 0.0353 & 0 & 0.3618 & 0.3154 & 0.1031 \\ 0.0353 & 0 & 0 & 0.2197 & 0.5815 & 0.8965 \end{bmatrix}$$

The equilibrium distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0.0039 & 0.0006 & 0.0001 & 0.0231 & 0.1388 & 0.8334 \end{bmatrix}^t$$

State $s_2$ indicates that 0.06 % of the time the token buffer has one token. However, state $s_8$ indicates that 80.34 % of the time the packet buffer is full indicating the source is misbehaving.

The throughput of the queue is given by

$$\begin{array}{lll} N_a(lost) & = 0.3279 & \text{packets/time step} \\ N_a'(lost) & = 1.2298 \times 10^4 & \text{packets/s} \\ L & = 0.1892 & \\ Th & = 1.4054 & \text{packets/time step} \\ Th' & = 5.2702 \times 10^4 & \text{packets/s} \\ p_a & = 0.8108 & \\ Q_a & = 2.8 & \text{packets} \\ W & = 1.9931 & \text{times steps} \\ W' & = 5.3149 \times 10^{-5} & \text{s} \end{array}$$

■

*Example 8.6.* Investigate the effect of doubling the token buffer or the packet buffer on the performance of the token bucket algorithm in the above example.

Doubling the token buffer to $B_t = 4$ or doubling the packet buffer results in the following parameters:

| Parameter | $B_t = 2, B_p = 3$ | $B_t = 4, B_p = 3$ | $B_t = 2, B_p = 6$ |
|---|---|---|---|
| $N_a(lost)$ | 0.3279 | 0.3279 | 0.3279 |
| $L$ | 0.1892 | 0.1892 | 0.1892 |
| $Th$ | 1.4054 | 1.4054 | 1.4054 |
| $p_a$ | 0.8108 | 0.8108 | 0.8108 |
| $Q_a$ | 2.8011 | 2.8011 | 5.8002 |
| $W$ | 1.9931 | 1.9931 | 4.1271 |

Because the token buffer was nearly empty in the original system, doubling the size of the token buffer has no impact on the system performance as can be seen by comparing the second and third columns of the above table. Doubling the packet buffer size doubles the delay without noticeable improvement in throughput. ■

## 8.4 Virtual Scheduling Algorithm

The virtual scheduling (VS) algorithm manages the ATM network traffic by closely monitoring the cell arrival rate. When a cell arrives, the algorithm calculates the theoretical arrival time ($TAT$) of the next cell according to the formula

$$TAT = \frac{1}{\lambda_a} \tag{8.75}$$

where $\lambda_a$ is the expected average data rate (units of cells/s). $TAT$ is measured by finding the difference between the arrival times of the headers of two consecutive ATM cells as explained in Fig. 8.11. This is *not* the time between the last bit of one cell and the first bit of the other.

If the cell arrival rate is in units of bits/s, then TAT is written as

$$TAT = \frac{A}{\lambda_a} \tag{8.76}$$

where $A$ is the size of an ATM cell which is 424 bits.

Assuming the time difference between the current cell and the next cell is $t$, then the cell is treated as conforming if $t$ satisfies the following inequality

$$t \geq TAT - \Delta \tag{8.77}$$

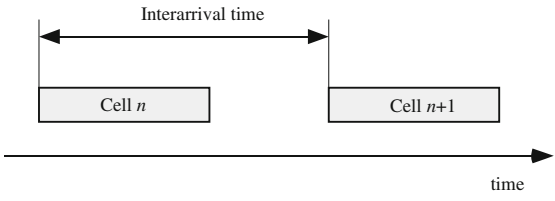



**Fig. 8.11** Measuring the interarrival time between two consecutive ATM cells

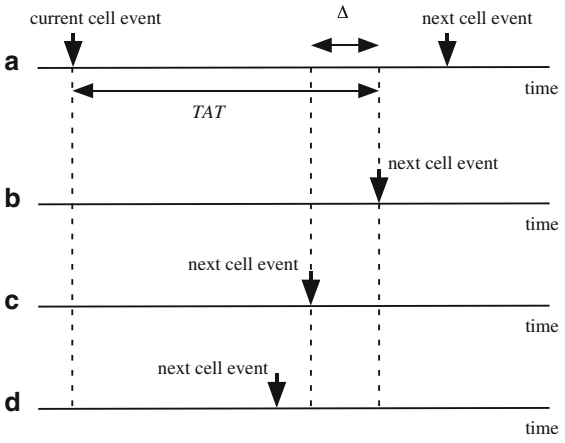



**Fig. 8.12** Different cases for cell arrival in the VS algorithm. (**a**) $t > TAT$ and cell is conforming. (**b**) $t = TAT$ and cell is conforming. (**c**) $t = TAT - \Delta$ and cell is conforming. (**d**) $t < TAT - \Delta$ and cell is nonconforming

where $\Delta$ is a small time value to allow for the slight variations in the data rate. The cell is treated as misbehaving, or nonconforming, when the cell arrival time satisfies the inequality

$$t < TAT - \Delta \tag{8.78}$$

The problem with the above two equations is that a source could keep sending data at a rate slightly higher than $\lambda_a$ and still be conforming if every cell arrives within the bound of (8.77).

Figure 8.12 shows the different cases for cell arrival in VS. Figure 8.12a shows a conforming cell because the cell arrival time satisfies (8.77). Figure 8.12b shows another conforming cell because the arrival time still satisfies (8.77). Figure 8.12c also shows a conforming cell because the arrival time satisfies the equality part of (8.77). Figure 8.12d shows a nonconforming cell because the arrival time does not satisfy (8.77).

### *8.4.1 Modeling the VS Algorithm*

In this section we perform Markov chain analysis of the virtual scheduling algorithm. We make the following assumptions for our analysis of the virtual scheduling algorithm.

1. The states of the Markov chain represent how many times the arriving cells from a certain flow have been nonconforming. In other words, state $s_i$ of the penalty queue indicates that the source has been nonconforming $i$ times.
2. The number of states ($B$) of the queue will dictate the maximum burst size tolerated, which is equal to the maximum number of penalties allowed for that source.

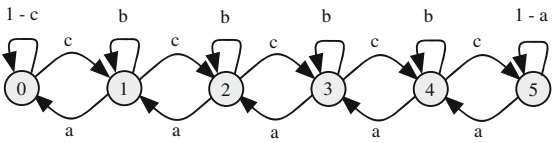



**Fig. 8.13** State transition diagram for the VS queue for the case $B = 5$

3. The queue changes states upon arrival of each cell.
4. Credit is given to the source each time it is conforming.
5. A penalty is given to the source each time it is nonconforming.
6. $a$ is the probability that the arriving cell satisfies the inequality $t \geq TAT$. In that case credit is issued to the source.
7. $b$ is the probability that the arriving cell satisfies the following inequality

$$TAT - \Delta \leq t \leq TAT$$

   In that case no credit or penalty is issued.
8. $c$ is the probability that the arriving cell satisfies the inequality $t < TAT$. In that case a penalty is issued to the source.

Of course $c = 1 - a - b$ since the source cannot be in any other state.

Based on the above assumptions, we have a single arrival, single departure $M/M/1/B$ queue with $B + 1$ states. Figure 8.13 shows the state transitions for the VS queue.

It is interesting to note that the state transition diagram of the virtual scheduling algorithm in Fig. 8.13 is a special case of the state transition diagram for the token bucket algorithm in Fig. 8.8 when the token bucket buffer size is $B_t = 1$.

The corresponding transition matrix $\mathbf{P}$ will be $(B + 1) \times (B + 1)$ and will have the following entries for the case $B = 5$.

$$\mathbf{P} = \begin{bmatrix} 1-c & a & 0 & 0 & 0 & 0 \\ c & b & a & 0 & 0 & 0 \\ 0 & c & b & a & 0 & 0 \\ 0 & 0 & c & b & a & 0 \\ 0 & 0 & 0 & c & b & a \\ 0 & 0 & 0 & 0 & c & 1-a \end{bmatrix} \tag{8.79}$$

Notice that the transition matrix is tridiagonal because of the single arrival, single departure feature of the queue.

### *8.4.2 VS Protocol Performance*

Having obtained the transition matrix, we are able to calculate the performance of the VS protocol.

The probability that an arriving cell is marked for discard is considered equal to the cell loss probability. This happens when the source exceeds the maximum number of penalties allowed. Thus cell loss probability is given by

$$L = c\ s_B \tag{8.80}$$

where $c$ is the probability that a cell arrived while the source is nonconforming and $s_B$ is the probability that the penalty queue is full.

The average number of cells that are dropped per time step is given by

$$N_a(lost) = L\ \lambda_a \tag{8.81}$$

where $\lambda_a$ is the average input data rate (units cells/time step).

The efficiency or access probability $p_a$ is the probability that an arriving cell is not dropped or marked for future discard. $p_a$ is given by

$$p_a = 1 - L = 1 - c\ s_B \tag{8.82}$$

The average number of packets that are accepted without being dropped per time step is the system throughput and is given by

$$\begin{aligned} Th &= p_a\ \lambda_a \\ &= (1 - c\ s_B)\ \lambda_a \end{aligned} \tag{8.83}$$

The maximum bust size allowed from the packet source is determined by the size of the queue.

$$\text{Max. burst size } = B \qquad \text{cells} \tag{8.84}$$

*Example 8.7.* Estimate the performance of the VS algorithm for a source having the following properties:

$$\begin{array}{ll} a = 0.2 & b = 0.5 \\ c = 0.3 & A = 424 \text{ bits} \\ \lambda = 150 \text{ Mbps} & B = 5 \end{array}$$

The transition matrix for the VS protocol is given by

$$\mathbf{P} = \begin{bmatrix} 0.7 & 0.2 & 0 & 0 & 0 & 0 \\ 0.3 & 0.5 & 0.2 & 0 & 0 & 0 \\ 0 & 0.3 & 0.5 & 0.2 & 0 & 0 \\ 0 & 0 & 0.3 & 0.5 & 0.2 & 0 \\ 0 & 0 & 0 & 0.3 & 0.5 & 0.2 \\ 0 & 0 & 0 & 0 & 0.3 & 0.8 \end{bmatrix}$$

The distribution vector for the system is

$$\mathbf{s} = \begin{bmatrix} 0.0481 & 0.0722 & 0.1083 & 0.1624 & 0.2436 & 0.3654 \end{bmatrix}^t$$

The performance figures of the protocol are as follows:

$$\begin{array}{lcl} L & = & 0.1096 \\ N_a(lost) & = & 16.4436 \times 10^6 \text{ packets/s} \\ p_a & = & 0.8904 \\ Th & = & 133.5564 \times 10^6 \text{ packets/s} \end{array}$$ ■

*Example 8.8.* What would happen in the above example if the VS algorithm uses a buffer whose size is $B = 2$?

The following table illustrates the effect of reducing the buffer size.

| Parameter | $B = 8$ | $B = 2$ |
|---|---|---|
| $L$ | 0.1096 | 0.1421 |
| $N_a(lost)$ (M packets/s) | 16.4436 | 21.3158 |
| $p_a$ | 0.8904 | 0.8579 |
| $Th$ (M packets/s) | 133.5564 | 128.6842 |

As expected, the reduced penalty buffer results in decreased performance such as higher cell loss probability and lower throughput. ■

## 8.5 Problems

### *Leaky Bucket Algorithm*

**8.1.** In a leaky bucket traffic shaper the packet arrival rate for a certain user is on the average 5 Mbps with a maximum burst rate of 30 Mbps. The output rate is dictated by the algorithm to be 10 Mbps. Derive the performance of this protocol using the $M/M/1/B$ modeling approach assuming packet buffer size to be $B = 5$ and the maximum line rate is 200 Mbps. Assume different values for the probability that the source is conforming.

**8.2.** Repeat Problem 8.1 using the $M^m/M/1/B$ modeling approach.

**8.3.** An alternative to modeling the leaky bucket algorithm using the $M^m/M/1/B$ queue is to assume the Bernoulli probability of packet arrival to be given as

$$a = \frac{\lambda_a}{\lambda_l}$$

which is similar to the packet arrival statistics for the $M/M/1/B$ queue. Study this situation using the data given in Example 8.1 and comment on your results.

## *Token Bucket Algorithm*

**8.4.** Consider the single arrival/departure model for the token bucket algorithm. Assume that arriving data are buffered in a packet buffer. Now we have two buffers to consider: the token buffer and the data buffer. Model the data buffer based on the results obtained for the token buffer

**8.5.** Estimate the maximum burst size in the token bucket protocol.

**8.6.** In a token bucket traffic shaper the packet arrival rate for a certain user is on the average 15 Mbps with a maximum burst rate of 30 Mbps. Assume the source is conforming 30 % of the time and the token arrival rate is dictated by the algorithm to be 20 Mbps. Study the state of the token buffer using the $M/M/1/B$ modeling approach assuming its size to be $B = 5$ and the maximum line rate is 100 Mbps.

**8.7.** Repeat Problem 8.6 using the multiple arrival/departure modeling approach.

**8.8.** Draw the Markov state transition diagram for the multiple arrivals/departures model when $B_t = 4$, $B_p = 3$, and $m = 4$.

**8.9.** Write down the transition matrix for the above problem and compare with the same system that had $m = 3$ in (8.63) on page 301.

**8.10.** Write down the transition matrix for the multiple arrivals/departures model when $B_t = 4$, $B_p = 6$, and $m = 8$.

## *Virtual Scheduling Algorithm*

**8.11.** Analyze the virtual scheduling algorithm in which an arriving cell is conforming if $t \geq TAT - \Delta$ and is nonconforming if $t < TAT - \Delta$.

**8.12.** Compare the performance of the VS algorithm treated in the text to the VS algorithm analyzed in Problem 8.11.

**8.13.** Analyze the virtual scheduling algorithm in which an arriving cell is issued a credit or penalty according to the criteria:

$$\begin{aligned} \text{credit:} \quad & t \geq TAT + \Delta \\ \text{no action:} \quad & TAT \leq t < TAT + \Delta \\ \text{no action:} \quad & ATA - \Delta \leq t < TAT + \Delta \\ \text{credit:} \quad & t \geq TAT - \Delta \end{aligned}$$

## Reference

1. A.S. Tanenbaum, *Computer Networks* (Prentice Hall, Upper Saddle River, 1996)

# Chapter 9
# Modeling Error Control Protocols

## 9.1 Introduction

Modeling a protocol or a system is just like designing a digital system, or any system for that mater. There are many ways to model a protocol based on the assumptions that one makes. My motivation here is simplicity and not taking a guided tour through the maze of protocol modeling. My recommendation to the reader is to read the discussion on each protocol then lay down the outline of a model that describes the protocol. The model or models developed here should then be compared with the one attempted by the reader.

Automatic-Repeat-reQuest (ARQ) techniques are used to control transmission errors caused by channel noise [1]. All ARQ techniques employ some kind or error coding of the transmitted data so that the receiver has the ability to detect the presence of errors. When an error is detected, the receiver requests a retransmission of the faulty data. ARQ techniques are simple to implement in hardware and they are especially effective when there is a reliable feedback channel connecting the receiver to the transmitter such that the round-trip delay is small.

There are three main types of ARQ techniques:

- Stop-and-Wait ARQ (SW ARQ).
- Go-Back-$N$ ARQ (GBN ARQ).
- Selective-Repeat ARQ (SR ARQ).

We discuss and model each of these techniques in the following sections.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_9

## 9.2 Stop-and-Wait ARQ Protocol

Stop-and-Wait ARQ (SW ARQ) protocol is a simple protocol for handling frame transmission errors when the round-trip time ($2\tau_p$) for frame propagation and reception of acknowledgment is smaller than frame transmission time ($\tau_t$). The propagation delay $\tau_p$ is given by

$$\tau_p = \frac{d}{c}$$

where $c$ is speed of light and $d$ is the distance between transmitter and receiver. The transmission delay $\tau_t$ is given by

$$\tau_t = \frac{L}{\lambda}$$

where $L$ is the number of bits in a frame and $\lambda$ is the transmission rate in bits per second.

Thus ARQ protocols are efficient and useful when we have

$$2\tau_p \ll \tau_t \tag{9.1}$$

If the above inequality is not true, then Forward Error Correction (FEC) techniques should be used [1].

When the sender transmits a frame on the forward channel, the receiver checks it for errors. If there are no errors, the receiver acknowledges the correct transmission by sending an Acknowledge (ACK) signal through the feedback channel. In that case the transmitter proceeds to send the next frame. If there were errors in the received frame, the receiver sends a Negative Acknowledgment signal (NAK) and the sender sends the same frame again. If the receiver does not receive ACK or NAK signals due to some problem in the feedback channel, the receiver waits for a certain timeout period and sends the frame again.

Based on the above discussion, we conclude that the time between transmitted frames is equal to $2\tau_p$ where $\tau_p$ is the one-way propagation delay.

Figure 9.1 shows an example of transmitting several frames using SW ARQ. Frame 1 was correctly received as indicated by the ACK signal and the sender starts sending frame 2.

Frame 2 was received in error as indicated by NAK and the grey line. The transmitter sends frame 2 one more time. For some reason no acknowledgment signals were received (indicated by short grey line) and the sender sends frame 2 for the third time after waiting for the proper timeout period. Frame 2 was received correctly as indicated by the ACK signal and the sender starts sending frame 3.

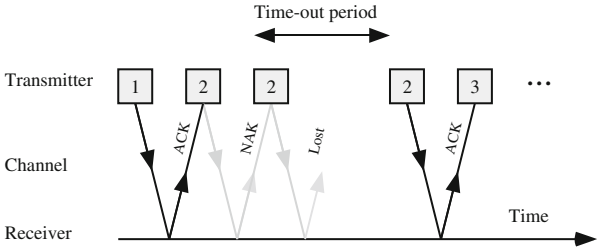
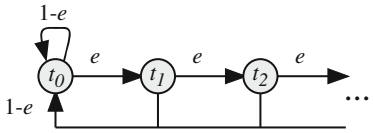



**Fig. 9.1** SW ARQ protocol

**Fig. 9.2** State transition diagram for the a sending station using the SW ARQ error control protocol




### 9.2.1 Modeling Stop-and-Wait ARQ

In this section we perform Markov chain analysis of the stop-and-wait algorithm. We make the following assumptions for our analysis of the SW ARQ

1. The average length of a frame is $n$ bits.
2. The forward channel has random noise and the probability that a bit will be received in error is $\epsilon$. Another name for $\epsilon$ is Bit Error Rate (BER) .
3. The feedback channel is assumed noise free so that acknowledgment signals from the receiving station will always be transmitted to the sending station.
4. The sender will keep sending a frame until it is correctly received. The effect of limiting the number of retransmissions is discussed in Problem 9.2.

The state of the sender while attempting to transmit a frame depends only on the outcome of the frame just sent. Hence we can represent the state of the sender as a Markov chain having the following properties:

1. State $i$ of the Markov chain indicates that the sender is retransmitting the frame for the $i$-th time. State zero indicates error-free transmission.
2. The number of states is infinite since no upper bound is placed on the number of retransmissions.
3. The time step is taken equal to the sum of transmission delay and round-trip delay $T = \tau_t + 2\tau_p$.

The state transition diagram for the SW ARQ protocol is shown in Fig. 9.2. In the figure, $e$ represents the probability that the transmitted frame contained an error. $e$ is given by the expression

$$e = 1 - (1 - \epsilon)^n \tag{9.2}$$

For a noise-free channel $\epsilon = 0$ and so $e = 0$ also. When the average number of errors in a frame is very small (i.e., $\epsilon\, n \ll 1$), we can write

$$e \approx \epsilon\, n \tag{9.3}$$

The quantity $\epsilon\, n$ is an approximation of the average number of bits in error in a frame (see Problems 9.4 and 9.5). Naturally we would like the number of errors to be small so as not to waste the bandwidth in retransmissions. Thus we must have

$$e = \epsilon\, n \ll 1 \tag{9.4}$$

Equation (9.2) assumed no FEC coding is implemented. Problem 9.4 requires you to derive an expression for $e$ when FEC is employed.

We organize the state distribution vector as follows.

$$\mathbf{s} = \begin{bmatrix} s_0 & s_1 & s_2 & \cdots \end{bmatrix}^t \tag{9.5}$$

where $s_i$ corresponds to retransmitting the frame for the $i$-th time. $s_0$ corresponds to transmitting the frame once with zero retransmissions. This is the case when the frame was correctly received without having to retransmit it.

The corresponding transition matrix of the channel is given by

$$\mathbf{P} = \begin{bmatrix} 1-e & 1-e & 1-e & 1-e & \cdots \\ e & 0 & 0 & 0 & \cdots \\ 0 & e & 0 & 0 & \cdots \\ 0 & 0 & e & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{9.6}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{9.7}$$

$$\sum \mathbf{s} = 1 \tag{9.8}$$

The solution to the above two equations is simple:

$$\mathbf{s} = (1-e) \times \begin{bmatrix} 1 & e & e^2 & \cdots \end{bmatrix}^t \tag{9.9}$$

### 9.2.2 SW ARQ Performance

The average number of retransmissions for a frame is given by

$$\begin{aligned} N_t &= (1-e) \times (s_1 + 2s_2 + 3s_3 + \cdots) \\ &= (1-e)^2 \times \sum_{i=0}^{\infty} i\ e^i \\ &= e \qquad \text{transmissions/frame} \end{aligned} \tag{9.10}$$

For a noise-free channel $e = 0$ and the average number of retransmissions is 0 also. This indicates that a frame is sent once for a successful transmission.

For a typical channel $e \approx \epsilon\, n \ll 1$ and the average number of transmissions can be approximated as

$$N_t \approx \epsilon\, n \tag{9.11}$$

We define the efficiency of the SW ARQ protocol as the inverse of the total number of transmissions which includes the first transmission plus the average number of retransmissions. In that case $\eta$ is given by

$$\eta = \frac{1}{1+N_t} = \frac{1}{1+e} \tag{9.12}$$

For an error-free channel $N_t = 0$ and $\eta = 100\,\%$. For a typical channel $e \approx \epsilon\, n \ll 1$ and the efficiency is given by

$$\eta \approx 1 - \epsilon\, n \tag{9.13}$$

This indicates that the efficiency decreases with increase in BER or frame size. Thus we see that the system performance will degrade gradually with any increase in the number of bits in the frame or any increase in the frame error probability.

The throughput of the transmitter can be expressed as

$$Th = \eta = 1 - e \qquad \text{frames/time step} \tag{9.14}$$

Thus for an error-free channel, $\eta = 1$ and arriving frames are guaranteed to be transmitted on the first try. We could have obtained the above expression for the throughput by estimating the number of frames that are successfully transmitted in each transmitter state:

$$\begin{aligned} Th &= (1-e) \sum_{i=0}^{\infty} s_i \\ &= 1 - e \end{aligned} \tag{9.15}$$

When errors are present in the channel, then $\eta < 1$ and so is the system throughput.

*Example 9.1.* Assume an SW ARQ protocol in which the frame size is $n = 1{,}000$ and the BER is $\epsilon = 10^{-4}$. Find the performance of the SW ARQ protocol for this channel. Repeat the example when the BER increases by a factor of ten.

According to (9.10) the average number of transmissions for a window is

$$N_t = 0.1052$$

and the efficiency is

$$\eta = 90.48\,\%$$

Notice that because the BER is low, we need just about one transmission to correctly receive a frame.

Now we increase the BER to $e = 10^{-3}$ and get the following results.

$$N_t = 1.7196$$

and the efficiency is

$$\eta = 36.77$$

Notice that when the BER is increased by one order of magnitude, the average number of frame retransmission is increased by a factor of 16.35. ■

## 9.3 Go-Back-$N$ (GBN ARQ) Protocol

In the go-back-$N$ protocol the transmitter keeps sending frames but keeps a copy in a buffer, which is called the transmission window. The number of frames in the buffer, or the window, is $N$ which equals the number of frames sent during one round-trip time.

When the sender transmits the frames on the forward channel, the receiver checks them for errors. If there are no errors, the receiver acknowledges each frame by sending ACK signals through the feedback channel. Upon reception of an ACK for a certain frame, the receiver drops it from the head of its buffer. If a received frame is in error, the receiver sends a NAK for that particular frame. When the transmitter receives the NAK signal, it resends all $N$ frames in its buffer starting with the frame in error.

Figure 9.3 shows an example of transmitting several frames using go-back-$N$ where the buffer size is $N = 3$. Solid arrows indicate ACK signals and grey arrows indicate NAK signals. Frame 1 was correctly received while frame 2 was received in error. We see that the transmitter starts to send frame 2 and the 3 subsequent frames that were in its buffer.

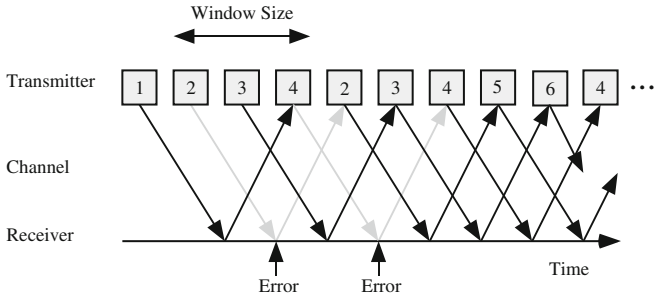



**Fig. 9.3** Go-back-$N$ ARQ protocol with buffer size $N = 3$. *Solid arrows* indicate ACK signals and *grey arrows* indicate NAK signals

### *9.3.1 Modeling the GBN ARQ Protocol*

In this section we perform Markov chain analysis of the GBN ARQ protocol. We make the following assumptions for our analysis of the go-back-$N$ protocol.

1. Each window contains $N$ frames.
2. The average length of a frame is $n$ bits.
3. The forward channel has random noise and the probability that a bit will be received in error is $\epsilon$. Another name for $\epsilon$ is BER
4. The feedback channel is assumed noise free so that acknowledgment signals from the receiving station will always be transmitted to the sending station.
5. The maximum number of retransmissions is $k_m$ after which the sender will declare the channel to be not functioning.

The state of the sender while attempting to transmit a frame depends only on outcome of the frame just sent. Hence we can represent the state of the sender as a Markov chain having the following properties:

1. The states of the Markov chain are grouped into the sets $\mathcal{T}$, $\mathcal{R}_1$, $\mathcal{R}_2$, etc. These sets are explained below.
2. The number of states is infinite since no upper bound is placed on the number of retransmissions.
3. The time step is taken equal to the sum of transmission delay of one frame $T = \tau_t$. Thus a window that contains $N$ frames will require $N$ time steps to be transmitted.

The set $\mathcal{T}$ represents the states of the sender while it is transmitting a window for freshly arrived $N$ frames.

$$\mathcal{T} = \{t_1\ t_2 \cdots t_N\} \tag{9.16}$$

The set $\mathcal{R}_1$ represents the states of the sender while it is retransmitting frames for the first time due to a damaged or lost frame. This set is the union of several subsets

$$\mathcal{R}_1 = R_{1,1} \cup R_{1,2} \cup \cdots \cup R_{1,N} \tag{9.17}$$

where subset $R_{1,i}$ is the subset of $\mathcal{R}_1$ that contain $i$ states corresponding to retransmitting $i$ frames for the first time. In other words, the first $N - i$ frames have been correctly received. With the help of (9.23) below we can verify that all states in subset $R_{1,i}$ are equal so that we can write the $i$ states associated with $R_{1,i}$ as

$$R_{1,i} = \left\{ r_{1,i}\ r_{1,i} \cdots r_{1,i} \right\} \tag{9.18}$$

Thus $\mathcal{R}_1$ has $N$ *unique* subsets such that subset $R_{1,i}$ has $i$ equal states $r_{1,i}$:

$r_{1,1}$ corresponding to last frame in error

$r_{1,2}$ corresponding to frame before last in error

$\vdots$

$r_{1,N}$ correspondingto first frame in error

Similarly the set $\mathcal{R}_2$ represents the states of the sender while it is retransmitting frames for the second time. This set is the union of several subsets

$$\mathcal{R}_2 = R_{2,1} \cup R_{2,2} \cup \cdots \cup R_{1,N} \tag{9.19}$$

where subset $R_{2,i}$ is the subset of $\mathcal{R}_2$ that contain $i$ states corresponding to retransmitting $i$ frames for the second time. With the help of (9.23) we can verify that all states in subset $R_{2,i}$ are equal so that we can write the $i$ states associated with $R_{2,i}$ as

$$R_{2,i} = \left\{ r_{2,i}\ r_{2,i} \cdots r_{2,i} \right\} \tag{9.20}$$

Thus $\mathcal{R}_2$ has $N$ *unique* subsets such that subset $R_{2,i}$ has $i$ equal states $r_{2,i}$:

$r_{2,1}$ corresponding to last frame in error

$r_{2,2}$ corresponding to frame before last in error

$\vdots$

$r_{2,N}$ correspondingto first frame in error

The state transition diagram for the GBN ARQ protocol is shown in Fig. 9.4. The sets of states $\mathcal{T}$, $\mathcal{R}_1$ and $\mathcal{R}_2$, are shown. To reduce clutter, only transitions in and out of $\mathcal{R}_1$ are indicated. The thick lines indicate multiple transitions lumped together. However, (9.23) shows all the transitions between states.

In the figure, all states in each column are equal due to the fact that the transition probabilities between them is one.

We organize the state distribution vector in the following order.

$$\mathbf{s} = \left[ \mathcal{T}\ \mathcal{R}_1\ \mathcal{R}_2 \cdots \right]^t \tag{9.21}$$

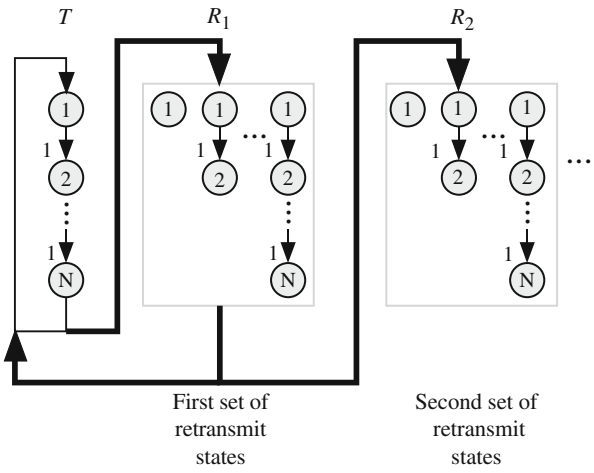



**Fig. 9.4** State transition diagram for a sending station using the GBN ARQ error control protocol. Only transitions in and out of $\mathcal{R}_1$ are indicated

For the case when $N = 2$ the distribution vector can be written as

$$\mathbf{s} = \left[ t \;\; t \;\; | \; r_{1,1} \;\; r_{1,2} \; r_{1,2} \;\; | \; r_{2,1} \;\; r_{2,2} \; r_{2,2} \;\; | \cdots \right]^t \tag{9.22}$$

The corresponding transition matrix of the sender for the case when $N = 2$ is given by

$$\mathbf{P} = \begin{bmatrix} 0 & p_{2,0} & p_{1,0} & 0 & p_{2,0} & p_{1,0} & 0 & p_{2,0} & \cdots \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ & & & & & & & & \\ 0 & p_{2,1} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ & & & & & & & & \\ 0 & p_{2,2} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots \\ & & & & & & & & \\ 0 & 0 & p_{1,1} & 0 & p_{2,1} & 0 & 0 & 0 & \cdots \\ & & & & & & & & \\ 0 & 0 & 0 & 0 & p_{2,2} & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{9.23}$$

In the above matrix, the transition probability $p_{i,j}$ is the probability that the last $j$ frames need to be retransmitted given that a frame of $i$ frames was sent. For instance, $p_{5,2}$ indicates the probability that the last two frames have to be retransmitted given that five frames were sent and the first three frames were received without error. $p_{i,j}$ is given by the expression

$$p_{i,j} = (1-e)^{i-j}\, e \tag{9.24}$$

where $e$ is the probability that a frame contained one or more errors.

$$e = 1 - (1 - \epsilon)^n \tag{9.25}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{9.26}$$

$$\sum \mathbf{s} = 1 \tag{9.27}$$

When $k_m$ is the maximum number of retransmissions, the dimension of $\mathbf{s}$ would be given by

$$\dim(\mathbf{s}) = N + \frac{K_m N(N+1)}{2} \tag{9.28}$$

As an example, for a window size $N = 32$ frames and the maximum number of retransmissions is $K_m = 16$, the size of $\mathbf{s}$ would be 33,344 and the state transition matrix would be of size $33,344 \times 33,344$. We can use MATLAB to find the distribution vector $\mathbf{s}$.

### 9.3.2 *Using Iterations to Find* s

An alternative approach is to find $\mathbf{s}$ using iterations. From the structure of the matrix $\mathbf{P}$, we can easily prove that all transmit states in the set $\mathcal{T}$ are equal.

$$t = t_1 = t_2 = \cdots = t_N \tag{9.29}$$

In fact, all states in any column of the matrix are equal. For example, we can write the $N$ *unique states* of $\mathcal{R}_1$ as

$$r_{1,N} = t\ p_{N,N} \tag{9.30}$$

$$r_{1,N-1} = t\ p_{N,N-1} \tag{9.31}$$

$$r_{1,N-2} = t\ p_{N,N-2} \tag{9.32}$$

$$\vdots$$

$$r_{1,1} = t\ p_{N,1} \tag{9.33}$$

In general we have

$$r_{1,j} = t\ p_{N,j} \qquad j = 1,\ 2,\ \cdots N \tag{9.34}$$

In that case state $r_{1,1}$ will be repeated once. State $r_{1,2}$ will be repeated twice, and finally $r_{1,N}$ will be repeated $N$ times.

For the rest of the retransmission states, we use iterative expressions as follows. The $N$ unique states of $\mathcal{R}_2$ are expressed in terms of the unique states of $\mathcal{R}_1$ as

$$r_{2,N} = r_{1,N}\ p_{N,N} \tag{9.35}$$

$$r_{2,N-1} = \sum_{i=N-1}^{N} r_{1,i}\ p_{i,N-1} \tag{9.36}$$

$$r_{2,N-2} = \sum_{i=N-2}^{N} r_{1,i}\ p_{i,N-2} \tag{9.37}$$

$$\vdots$$

$$r_{2,1} = \sum_{i=1}^{N} r_{1,i}\ p_{i,1} \tag{9.38}$$

In general we have

$$r_{2,j} = \sum_{i=j}^{N} r_{1,i}\ p_{i,j} \qquad j = 1,\ 2,\ \cdots N \tag{9.39}$$

The states associated with the $k$-th retransmission $\mathcal{R}_k$ are given by the iterative expression

$$r_{k,j} = \sum_{i=j}^{N} r_{k-1,i}\ p_{i,j} \qquad \begin{matrix} k = 2,\ 3,\ \cdots \\ j = 1,\ 2,\ \cdots N \end{matrix} \tag{9.40}$$

with the initial condition

$$r_{1,j} = t\ p_{N,j} \qquad j = 1,\ 2,\ \cdots N \tag{9.41}$$

### *9.3.3 Algorithm for Finding s by Iterations*

The above iterations express all the retransmission states in terms of the transmit state $t$. In order to find the distribution vector $\mathbf{s}$, we follow this algorithm:

1. Assign to each transmit state some value, for example

$$t = 1 \tag{9.42}$$

2. Estimate the retransmit states for $\mathcal{R}_1$ using the iterative expression (9.34).
3. Estimate the values of the other retransmit states $\mathcal{R}_k$ using the iterative expression (9.40).
4. Find the sum of all states

$$\mathcal{S} = N\ t + \sum_{k=1}^{K_m} \sum_{j=1}^{N} j\ r_{k,j} \tag{9.43}$$

5. The normalized value of the distribution vector is given by the following equation.

$$\mathbf{s} = \frac{1}{\mathcal{S}} \left[ t\ t\ \mid r_{1,1}\ \ r_{1,2}\ r_{1,2}\ \mid r_{2,1}\ \ r_{2,2}\ r_{2,2}\ \mid \cdots \right]^t \tag{9.44}$$

### 9.3.4 GBN ARQ Performance

As long as the sender keeps retransmitting frames that were received in error, the next frame cannot be sent. Therefore, we are interested in estimating the average number of retransmissions for a given frame ($R_a$), the average number of frames sent in each retransmission attempt ($N_a$), and the average delay a frame takes to be transmitted when errors are present ($T_a$).

#### *Estimating Average Number of Retransmissions $R_a$*

The probability that the source is in the $k$-th retransmission state is given by

$$\alpha_k = \sum_{j=1}^{N} j\ r_{k,j} \tag{9.45}$$

$\alpha_k$ is also equal to the average number of frames sent at the $k$-th retransmission ($n_k$).

The average number of retransmissions by the source for a given frame is given by

$$R_a = \sum_{k=1}^{k_m} k\ \alpha_k \tag{9.46}$$

### *Estimating Average Delay $T_a$*

The delay associated with the $k$-th retransmission is given by the accumulation of all the frames that were previously sent by earlier retransmissions. Thus we can write

$$t_k = \sum_{j=1}^{k} \alpha_j \tag{9.47}$$

The average delay for transmitting a given frame is given by

$$T_a = \sum_{k=1}^{k_m} t_k \, \alpha_k \tag{9.48}$$

### *Estimating the Average Number of Frames Sent $N_a$*

The average number of frames sent due to all retransmissions is given by

$$N_a = \sum_{k=1}^{k_m} n_k \, \alpha_k \tag{9.49}$$

### *GBN ARQ Efficiency $\eta$ and Throughput ($Th$)*

The efficiency of the GBN ARQ protocol is the ratio of frame size to the total number of frames transmitted:

$$\eta = \frac{N}{N + N_a} \tag{9.50}$$

When there are no errors $N_a = 0$ and we get 100 % efficiency.

The throughput of the transmitter can be expressed as

$$Th = \eta \qquad \text{frames/time step} \tag{9.51}$$

Thus for an error-free channel, $\eta = 1$ and arriving frames are guaranteed to be transmitted on the first try.

When errors are present in the channel, then $\eta < 1$ and so is the system throughput.

*Example 9.2.* Assume a GBN ARQ protocol with the following parameters.

$$\begin{aligned} n &= 500 \quad \text{bits} \\ N &= 8 \quad \text{frames} \\ \epsilon &= 10^{-4} \\ K_m &= 16 \end{aligned}$$

Find the performance of the GBN ARQ protocol for this channel.

Using the technique in Sect. 9.3.2, the average number of retransmissions for a given frame is

$$R_a = 0.2195$$

the average delay for a given frame is

$$T_a = 0.0306$$

the average number of retransmitted frames for a given frame is

$$N_a = 0.026$$

and the efficiency is

$$\eta = 99.68\,\%$$

■

## 9.4 Selective-Repeat (SR ARQ) Protocol

The selective-repeat protocol is a general strategy for handling frame transmission errors when the round-trip time for frame transmission and reception of the acknowledgment is comparable to frame transmission time. SR ARQ is used by the TCP transport protocol. In this protocol the transmitter groups the frames into windows, so that each window contains $N$ frames. When the sender sends frames within a window, the receiver stores the frames of the current window and checks for errors. After a complete window has been received, or after the proper timeout period, the receiver instructs the transmitter to resend only the frames that contained errors. That results in a more efficient protocol compared to GBN ARQ that resends frames in error as well as error-free frames.

Figure 9.5 shows an example of transmitting several frames using selective-repeat protocol where the buffer size is $N = 3$. Solid arrows indicate ACK signals and grey arrows indicate NAK signals. Frame 1 was correctly received while frame

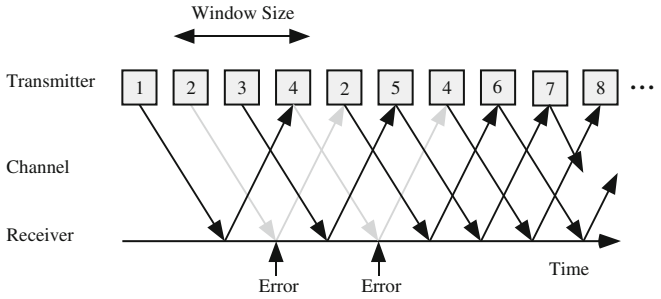



**Fig. 9.5** Selective-repeat ARQ protocol with buffer size $N = 3$. *Solid arrows* indicate ACK signals and *grey arrows* indicate NAK signals

2 was received in error. We see that the transmitter starts to send frame 2 as soon as the corresponding NAK is received. Frame 4 was also received in error and we can see that it is retransmitted as soon as its NAK signal was received.

### *9.4.1 Modeling the SR ARQ Protocol*

In this section we perform Markov chain analysis of the SR ARQ protocol. We make the following assumptions for our analysis of the selective-repeat protocol.

1. Each window contains $N$ frames.
2. The average length of a frame is $n$ bits.
3. The forward channel has random noise and the probability that a bit will be received in error is $\epsilon$. Another name for $\epsilon$ is BER.
4. The feedback channel is assumed noise free so that acknowledgment signals from the receiving station will always be transmitted to the sending station.
5. The maximum number of retransmissions is $k_m$ after which the sender will declare the channel to be not functioning.

The state of the sender while attempting to transmit a frame depends only on the outcome of the frame just sent. Hence we can represent the state of the sender as a Markov chain having the following properties:

1. The states of the Markov chain are grouped into the sets $\mathcal{T}$, $\mathcal{R}_1$, $\mathcal{R}_2$, etc. These sets are explained below.
2. The number of states is infinite since no upper bound is placed on the number of retransmissions.
3. The time step is taken equal to the sum of transmission delay of one frame $T = \tau_t$. Thus a window that contains $N$ frames will require $N$ time steps to be transmitted.

The set $\mathcal{T}$ represents the states of the sender while it is transmitting a window for freshly arrived $N$ frames.

$$\mathcal{T} = \{t_1\ t_2 \cdots t_N\} \tag{9.52}$$

The set $\mathcal{R}_1$ represents the states of the sender while it is retransmitting frames for the first time. This set is the union of several subsets

$$\mathcal{R}_1 = R_{1,1} \cup R_{1,2} \cup \cdots \cup R_{1,N} \tag{9.53}$$

where subset $R_{1,i}$ is the subset of $\mathcal{R}_1$ that contain $i$ states corresponding to retransmitting $i$ frames for the first time. With the help of (9.59) we can verify that all states in subset $R_{1,i}$ are equal so that we can write the $i$ states associated with $R_{1,i}$ as

$$R_{1,i} = \{r_{1,i}\ r_{1,i} \cdots r_{1,i}\} \tag{9.54}$$

Thus $\mathcal{R}_1$ has $N$ *unique* states:

$r_{1,1}$ repeated once,
$r_{1,2}$ repeated twice,
$\vdots$
$r_{1,N}$ repeated N times.

Similarly the set $\mathcal{R}_2$ represents the states of the sender while it is retransmitting frames for the second time. This set is the union of several subsets

$$\mathcal{R}_2 = R_{2,1} \cup R_{2,2} \cup \cdots \cup R_{1,N} \tag{9.55}$$

where subset $R_{2,i}$ is the subset of $\mathcal{R}_2$ that contain $i$ states corresponding to retransmitting $i$ frames for the second time. With the help of (9.59) we can verify that all states in subset $R_{2,i}$ are equal so that we can write the $i$ states associated with $R_{2,i}$ as

$$R_{2,i} = \{r_{2,i}\ r_{2,i} \cdots r_{2,i}\} \tag{9.56}$$

Thus $\mathcal{R}_2$ has $N$ *unique* states:

$r_{2,1}$ repeated once,
$r_{2,2}$ repeated twice,
$\vdots$
$r_{2,N}$ repeated N times.

The state transition diagram for the SR ARQ protocol is shown in Fig. 9.6. The sets of states $\mathcal{T}$, $\mathcal{R}_1$, and $\mathcal{R}_2$ are shown. To reduce clutter, only transitions in and out of $\mathcal{R}_1$ are indicated. The thick lines indicate multiple transitions lumped together. However, (9.59) shows all the transitions between states.

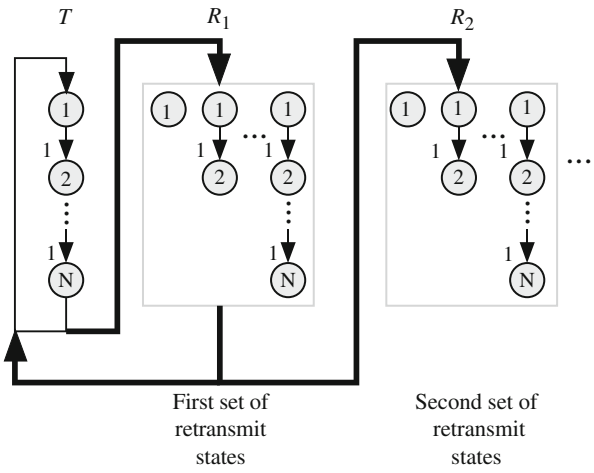



**Fig. 9.6** State transition diagram for a sending station using the SR ARQ error control protocol. Only transitions in and out of $\mathcal{R}_1$ are indicated

In the figure, all states in each column are equal due to the fact that the transition probabilities between them is one.

We organize the state distribution vector in the following order.

$$\mathbf{s} = \left[\mathcal{T}\ \mathcal{R}_1\ \mathcal{R}_2\ \cdots\right]^t \tag{9.57}$$

For the case when $N = 2$ the distribution vector can be written as

$$\mathbf{s} = \left[t\ t\ \mid r_{1,1}\ \ r_{1,2}\ r_{1,2}\ \mid r_{2,1}\ \ r_{2,2}\ r_{2,2}\ \mid \cdots\right]^t \tag{9.58}$$

The corresponding transition matrix of the sender for the case when $N = 2$ is given by

$$\mathbf{P} = \begin{bmatrix}
0 & p_{2,0} & p_{1,0} & 0 & p_{2,0} & p_{1,0} & 0 & p_{2,0} & \cdots \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & p_{2,1} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & p_{2,2} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & 0 & p_{1,1} & 0 & p_{2,1} & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & p_{2,2} & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix} \tag{9.59}$$

In the above matrix, the transition probability $p_{i,j}$ is the probability that $j$ frames need to be retransmitted given that a frame of $i$ frames was sent. For instance, $p_{5,2}$ indicates the probability that two frames have to be retransmitted given that five frames were sent. $p_{i,j}$ is given by the expression

$$p_{i,j} = \binom{i}{j} (1-e)^{i-j}\, e^{j} \tag{9.60}$$

where $e$ is the probability that a frame contained one or more errors.

$$e = 1 - (1-\epsilon)^{n} \tag{9.61}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{9.62}$$

$$\sum \mathbf{s} = 1 \tag{9.63}$$

When $k_m$ is the maximum number of retransmissions, the dimension of $\mathbf{s}$ would be given by

$$\dim(\mathbf{s}) = N + \frac{K_m N(N+1)}{2} \tag{9.64}$$

As an example, for a window size $N = 32$ frames and the maximum number of retransmissions is $K_m = 16$, the size of $\mathbf{s}$ would be 33,344 and the state transition matrix would be of size $33,344 \times 33,344$. We can use MATLAB to find the distribution vector $\mathbf{s}$.

### *9.4.2 SR ARQ Performance*

As long as the sender keeps retransmitting frames that were received in error, the next frame cannot be sent. Therefore, we are interested in estimating the average number of retransmissions for a given frame ($R_a$), the average number of frames sent in each retransmission attempt ($N_a$), and the average delay a frame takes to be transmitted when errors are present ($T_a$).

### *Estimating Average Number of Retransmissions $R_a$*

The probability that the source is in the $k$-th retransmission state is given by

$$\alpha_k = \sum_{j=1}^{N} r_{k,j} \tag{9.65}$$

The average number of retransmissions by the source for a given frame is given by

$$R_a = \sum_{k=1}^{k_m} k\ \alpha_k \tag{9.66}$$

### *Estimating Average Delay $T_a$*

The average number of frames sent at the $k$-th retransmissions is given by

$$n_k = \sum_{j=1}^{N} j\ r_{k,j} \tag{9.67}$$

The delay associated with the $k$-th retransmission is given by the accumulation of all the frames that were previously sent by earlier retransmissions. Thus we can write

$$t_k = \sum_{j=1}^{k} n_j \tag{9.68}$$

The average delay for transmitting a given frame is given by

$$T_a = \sum_{k=1}^{k_m} t_k\ \alpha_k \tag{9.69}$$

### *Estimating the Average Number of Frames Sent $N_a$*

The average number of frames sent due to all retransmissions is given by

$$N_a = \sum_{k=1}^{k_m} n_k\ \alpha_k \tag{9.70}$$

### *SR ARQ Efficiency η and Throughput (Th)*

The efficiency of the SR ARQ protocol is the ratio of frame size to the total number of frames transmitted:

$$\eta = \frac{N}{N + N_a} \tag{9.71}$$

When there are no errors $N_a = 0$ and we get 100 % efficiency.

The throughput of the transmitter can be expressed as

$$Th = \eta \qquad \text{frames/time step} \tag{9.72}$$

Thus for an error-free channel, $\eta = 1$ and arriving frames are guaranteed to be transmitted on the first try.

When errors are present in the channel, then $\eta < 1$ and so is the system throughput.

*Example 9.3.* Assume an SR ARQ protocol with the following parameters.

$$\begin{aligned} n &= 500 \qquad \text{bits} \\ N &= 8 \qquad \text{frames} \\ e &= 10^{-4} \\ K_m &= 16 \end{aligned}$$

Find the performance of the SR ARQ protocol for this channel.

Using the technique in Sect. 9.3.2, the average number of retransmissions for a given frame is

$$R_a = 0.37$$

the average delay for a given frame is

$$T_t = 0.14$$

the average number of retransmitted frames for a given frame is

$$N_a = 0.13$$

and the efficiency is

$$\eta = 98.41\,\%$$

Comparing these results with those of GBN ARQ we note that SR ARQ performs better for the same parameters. ■

## 9.5 Problems

### *SW ARQ Protocol*

**9.1.** Prove that the probability $e$ that a frame in the SW ARQ protocol is in error is $e \approx \epsilon\, n$ when $\epsilon \ll 1$.

**9.2.** One of the assumptions in Sect. 9.2.1 of the SW ARQ protocol was that the sender will keep retransmitting the frame until it is correctly received. Assume the maximum number of retransmissions is limited to $k_m$. How will this impact the state transition diagram, state transition matrix, the distribution vector, and the system performance?

**9.3.** Assume an SW ARQ protocol in which the frame size is 100 bits and the probability that a received frame is in error is $\epsilon = 10^{-5}$. Find the performance of the protocol.

**9.4.** Assume a Forward Error Control (FEC) coding is used such that the receiving station can correctly decode a received frame if the number of errors does not exceed $k$ errors. Obtain an expression for the frame error probability $e$ under this scheme and compare the expression you get to Eq. (9.2).

**9.5.** Assume an SW ARQ protocol in which the frame size is $n$ bits but FEC is used to improve the performance. The FEC code employed can correct only up to $k = 3$ bits in error.

1. Draw the transition diagram for such a protocol and compare with the standard SW ARQ protocol discussed in the text.
2. Derive the transition matrix and compare with the standard SW ARQ protocol.
3. Estimate the performance of this protocol and compare with the standard SW ARQ protocol.

**9.6.** Equations (9.11) and (9.13) indicate that SW ARQ performance will not change if we scale $n$ to $\alpha n$, where $\alpha > 1$ and decrease $\epsilon$ by the same scale factor $\epsilon/\alpha$ when $\epsilon \ll 1$. Verify these assertions using an SW ARQ parameters of $n = 100$, $\epsilon = 10^{-4}$.

**9.7.** Assume SW ARQ where the sender has a transmit buffer of size $B$. Study the sender transmit buffer.

### *GBN ARQ Protocol*

**9.8.** Obtain the transition matrix for the GBN ARQ protocol having the following parameters (chosen to make the problem manageable)

$$N = 3 \quad k_m = 1$$

Assume two cases of the channel: a very noisy channel ($\epsilon = 0.01$) and for a less noisy channel ($\epsilon = 10^{-5}$). Compare the two matrices and state your conclusions.

**9.9.** Given a GBN ARQ protocol in which the window size is $N = 20$ frames and the probability that a received frame is in error is $e = 10^{-4}$. Find the performance of such a protocol for this channel assuming $k_m = 8$.

**9.10.** Given a GBN ARQ protocol in which the window size is $N = 20$ frames and the probability that a received frame is in error is $e = 5\times10^{-4}$. Find the performance of such a protocol for this channel.

**9.11.** Assume the GBN ARQ protocol is now modified such that if the received window contained one or more frames in error, then the whole window is discarded and a request is issued to retransmit the entire window again. This is repeated for a maximum of $k_m$ times until an error-free window is received.

1. Identify the states of this system.
2. Write down the transition matrix.
3. The transition matrix that results will be reducible. Derive the steady-state distribution vector.

## *SR ARQ Protocol*

**9.12.** Obtain the transition matrices for the SR ARQ protocol having the following parameters (chosen to make the problem manageable)

$$N = 3 \quad k_m = 1$$

Assume two cases of the channel: a very noisy channel ($e = 0.01$) and for a less noisy channel ($e = 10^{-5}$). Compare the two matrices and state your conclusions.

**9.13.** Given an SR ARQ protocol in which the window size is $N = 20$ frames and the probability that a received frame is in error is $e = 10^{-4}$. Find the performance of such a protocol for this channel assuming $k_m = 8$.

**9.14.** Given an SR ARQ protocol in which the window size is $N = 20$ frames and the probability that a received frame is in error is $e = 5\times10^{-4}$. Find the performance of such a protocol for this channel.

**9.15.** Consider an SR ARQ protocol where FEC coding is employed. In that scheme, the sender adds extra correction bits to each frame or frame. The receiver is thus able to correct frames that have up to two errors per window. Draw the transition diagram for this system and compare with the SR ARQ protocol discussed in this chapter. Derive the relevant performance for this modified protocol.

**9.16.** In the SR ARQ protocol discussed in the test, when a window is received with $i$ errors, the $i$ frames are retransmitted until all of them are received without

any errors. Now consider the case when the receiver only request to retransmit $j$ frames out of the $i$ frames that contained $i$ errors originally. Do you expect this protocol to perform better than the SR ARQ protocol discussed in text?

## Reference

1. S. Lin, D.J. Costello, Jr., M.J. Miller, Automatic-repeat-request error-control schemes. IEEE Commun. Mag. **22**(12), 5–17 (1984)

# Chapter 10
# Modeling Medium Access Control Protocols

## 10.1 Introduction

In this chapter we illustrate how to develop models for several medium access control (MAC) protocols that are commonly used in computer communications. We will model the following medium access protocols in this chapter and in the next chapters as well:

1. IEEE Standard 802.1p: The Static Priority Scheduling Algorithm
2. Pure ALOHA
3. Slotted ALOHA
4. IEEE 802.3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
5. Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)
6. IEEE 802.11: for ad hoc wireless local area networks (LANs) using the basic distributed coordination function (DCF)
7. IEEE 802.11: for ad hoc wireless LANs using the request to send/clear to send (RTS/CTS) protocol.
8. IEEE 802.11e: for ad hoc wireless LANs using Enhanced Distributed Channel Access (EDCA)
9. IEEE 802.11e: for infrastructure wireless LANs using Hybrid Coordination function Control Channel Access (HCCA)
10. IEEE 802.16 (WiMAX): for infrastructure metropolitan area networks (MAN).

The static, or fixed, priority scheduling algorithm is lumped with media access algorithms since static priority is also used as a medium access protocol. Because MAC belongs to the data link layer, our unit of data transfer is the frame since packets are the business of the higher layer like the network layer and above.

Modeling a protocol is just like designing a digital system, or any system for that mater. There are many ways to model a protocol based on the assumptions that one makes. Our motivation here is simplicity and not taking a guided tour through the maze of protocol modeling. Our recommendation to the reader is to read the



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_10

discussion on each protocol then lay down the outline of a model that describes the protocol. The model or models developed here should then be compared with the one attempted by the reader.

## 10.2 IEEE Standard 802.1p: Static-Priority Protocol

The IEEE 802.1p is based on the static priority scheduling algorithm. IEEE 802.1p standard has a priority scheme such that frames queued in a lower priority queue are not sent if there are frames queued in higher priority queues. The frames are sent only when all higher priority queues are empty. In that sense, the static-priority protocol is a scheduling protocol to provide access to an outgoing link among several competing queues.

The IEEE 802.1p enables creating priority classes for network traffic. This enables quality of service (QoS) support. The analysis given here provides insight into how to use Markov chains to derive important performance figures.

### *10.2.1 Modeling the IEEE 802.1p: Static-Priority Protocol*

In this section we assume there are $N$ priority classes and each class has its own queue to store incoming traffic for that class. The state of each queue depends only on its immediate past history and we can model the queues using Markov chain analysis. To start our analysis, we employ the following assumptions.

1. The states of the Markov chain represent the occupancy of the priority queues.
2. The time step is taken equal to the transmission delay of a frame.
3. There are $N$ priority classes; with class 1 having the highest priority and so on till class $N$ which has the lowest priority.
4. The size of the queue in priority class $i$ is equal to $B_i$.
5. $a_i$ is the frame arrival probability for queue $i$.
6. $c_i$ is the frame departure probability for queue $i$.
7. Arrivals are processed at the same time step.
8. All frames have equal lengths.

Figure 10.1 illustrates the flows into and out of each queue. The downward arrows represent lost flows. Notice that some data is lost by each queue when the arrival rate exceeds the departure rate. The highest priority queue does not suffer any data loss since its data is guaranteed service. The highest priority queue does not need a buffer to store incoming data when preemptive static priority is employed. If a nonpreemptive scheme is employed, then the highest priority queue will require a buffer of size one to store incoming data until the frame being sent is finished.

From the above assumptions, we can write for queue 1 the following frame arrival and departure probabilities:

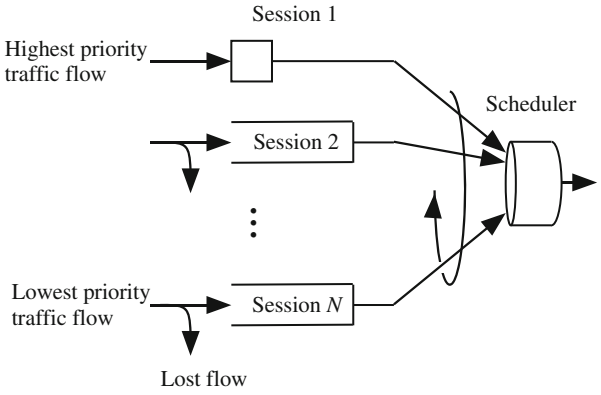



**Fig. 10.1** The flows into and out of each priority queue in the static priority scheduling protocol. The *downward arrows* represent lost flows

$$u_1(\text{arrival}) = a_1 \tag{10.1}$$

$$u_1(\text{departure}) = c_1 = 1 \tag{10.2}$$

For queue 2 we can write the following frame arrival and departure probabilities:

$$u_2(\text{arrival}) = a_2 \tag{10.3}$$

$$u_2(\text{departure}) = c_2 = e_1 \tag{10.4}$$

where $e_1 = b_1 = 1 - a_1$ is the probability that queue 1 is empty since no frames arrive. Thus queue 2 can access the output channel only when queue 1 is empty. The probability that queue 2 is empty is given by the expression for state $s_0$ of the $M/M/1/B$ queue:

$$e_2 = \frac{1 - \rho_2}{1 - \rho_2^{B_2+1}} \tag{10.5}$$

where $B_2$ is the size of the queue 2 buffer and $\rho_2$ is the distribution index for queue 2.

$$\rho_2 = \frac{a_2\, d_2}{b_2\, c_2} \tag{10.6}$$

with $b_2 = 1 - a_2$ and $d_2 = 1 - c_2$.

For queue 3 we can write the following frame arrival and departure probabilities:

$$p_3(\text{arrival}) = a_3 \tag{10.7}$$

$$p_3(\text{departure}) = c_3 = e_2 \times e_1 \tag{10.8}$$

$$e_3 = \frac{1-\rho_3}{1-\rho_3^{B_3+1}} \tag{10.9}$$

In general we can write the following iterative expression for queue $i$, where $1 < i \leq N$,

$$p_i\,(\text{arrival}) = a_i \tag{10.10}$$

$$p_i\,(\text{departure}) = c_i = e_{i-1}\, e_{i-2} \cdots e_1 \tag{10.11}$$

$$e_i = \frac{1-\rho_i}{1-\rho_i^{B_i+1}} \tag{10.12}$$

where $\rho_i$ is the distribution index for queue $i$ and is given by

$$\rho_i = \frac{a_i\ d_i}{b_i\ c_i} \tag{10.13}$$

with $b_i = 1 - a_i$ and $d_i = 1 - c_i$.

After the arrival and departure probabilities of each queue are found, we can estimate the queue parameters according to the $M/M/1/B$ analysis in Sect. 7.6.

The performance parameters for queue 1 are a bit unique due to its high priority

$$\begin{aligned} Th_1 &= 1 \text{ frames/time step} \\ p_{a,1} &= 1 \\ N_{a,1}(lost) &= 0 \text{ frames/time step} \\ L_1 &= 0 \\ Q_{a,1} &= 0 \text{ frames} \\ W_1 &= 0 \text{ time steps} \end{aligned}$$

For queue $i$ with $1 < i \leq N$ we can write

$$Th_i = c_i\ (1 - b_i\ e_i) \text{ frames/time step}$$

$$p_{a,i} = \frac{c_i(1-b_i\ e_i)}{a_i}$$

$$N_{a,i}(lost) = e_i\ a_i\ d_i \quad \text{frames/time step}$$

$$L_i = e_i\ d_i$$

$$Q_{a,i} = \frac{\left[\rho_i - (B_i+1)\rho_i^{B_i} + B_i \rho_i^{B_i+1}\right]}{(1-\rho_i)\times\left(1-\rho_i^{B_i+1}\right)} \text{ frames}$$

$$W_i = \frac{Q_{a,i}}{Th_i} \quad \text{time steps}$$

*Example 10.1.* Consider a static-priority protocol serving four users where the frame arrival probabilities for all users are equal (i.e., $a_i = a$ for all $1 \leq i \leq 4$) and all users have the same buffer size (i.e., $B_i = B$ for all $1 \leq i \leq 4$). Estimate the performance of each user.

The following table shows the performance parameters for each user starting with the highest priority user.

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $Th$ | 1 | 0.3965 | 0.3167 | 0.0376 |
| $p_a$ | 1 | 0.9912 | 0.7916 | 0.0940 |
| $N_a(lost)$ | 0 | 0.0035 | 0.0833 | 0.3624 |
| $L$ | 0 | 0.0088 | 0.2084 | 0.9060 |
| $Q_a$ | 0 | 0.6941 | 3.8790 | 4.9377 |
| $W$ | 0 | 1.7507 | 12.2502 | 131.2926 |

As expected, the least priority queue has the worst performance. ■

## 10.3 ALOHA

ALOHA was developed by N. Abramson in the 1970s at the University of Hawaii to allow several computers spread over a wide geographical area to communicate using a broadcast wireless channel. The technique chosen was simple and applies to any system where several users attempt to access a shared resource without the help of a central controller. ALOHA did not require global time synchronization and simplified its implementation but impacted its performance.

In ALOHA network, a user that has a frame to transmit does so without waiting to see if the channel is busy or not. When two users transmit at the same time both colliding frames will be received in error due to interference. The collision phenomenon that occurs in a shared medium is also known as *contention*. The sender knows that contention has occurred by *listening* to the channel to check if the frame it just sent is in error or not. If errors are detected, the sender retransmits the frame after waiting for a random amount of time. Another way for the sender to sense collision is to wait for an acknowledgment from the receiver.

Figure 10.2 illustrates the ALOHA contention problem. The figure assumes all transmitted frames have equal lengths and each frame has a duration $T$. The time $T$ is equal to the maximum propagation delay between any pair of stations in the

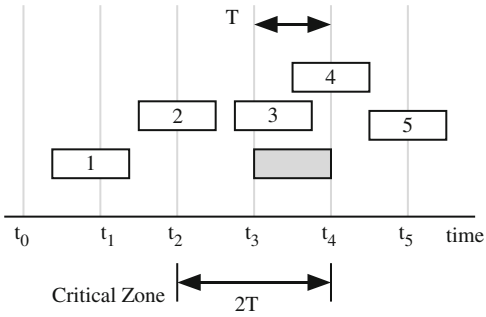



**Fig. 10.2** ALOHA contention problem. Illustrating all possible conflict situations encountered by the transmitted frame (*shaded rectangle*)

**Table 10.1** ALOHA contention problem. Illustrating all possible conflict situations encountered by the transmitted frame (shaded rectangle in Fig. 10.2)

| Frame | Start time $t$ | Contention with shaded frame |
|---|---|---|
| 1 | $t_0 < t < t_1$ | No |
| 2 | $t_1 < t < t_2$ | No |
| 3 | $t_2 < t < t_3$ | Yes |
| 4 | $t_3 < t < t_4$ | Yes |
| 5 | $t_4 < t < t_5$ | No |

network. Because of the "free for all" situation, anything can happen. Let us see how a conflict might arise while attempting to transmit one frame, shown as the shaded block in Fig. 10.2. Table 10.1 summarizes potential conflict situations.

We make the following conclusions based on Fig. 10.2 and Table 10.1. Any frame transmitted in the period $T$ before our shaded frame will cause contention (e.g., frame 3). Any frame transmitted during the period $T$ when our shaded frame is being sent will cause contention (e.g., frame 4). The critical zone during transmission of the shaded frame is shown at the bottom of Fig. 10.2. Thus for a successful transmission at a given time step, the channel must be quiet and all users must be idle for previous time step.

### *10.3.1 Modeling the ALOHA Network*

In this section we perform Markov chain analysis of the ALOHA network. We make the following assumptions for our analysis of ALOHA

1. The states of the Markov chain represent the status of the wireless channel: *idle*, *transmitting*, and *collided*.
2. The propagation delay between any pair of users is less than the frame time $T$.
3. The time step value $T$ is taken equal to the frame transmission delay.
4. There are $N$ users in the system.
5. Users can transmit any time they want.
6. The probability that a user transmits a frame in one time step is $a$.
7. All frames have equal lengths and the duration of each frame is $T$.

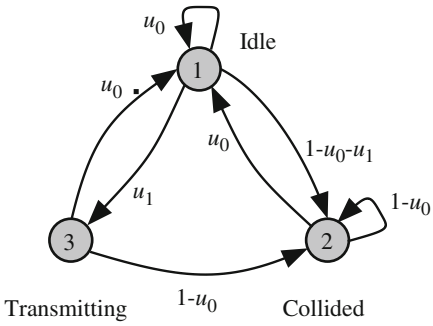



**Fig. 10.3** State transition diagram for the ALOHA channel

8. Contention occurs if a frame is sent at time $t$ and there are transmissions during the time period $T - t$ to $t + T$.
9. A user retransmits a corrupted frame after waiting a random amount of time.

Based on the above assumptions the wireless channel can be in one of three states: *idle, collided*, or *transmitting*. Figure 10.3 shows the transition diagram for our Markov chain. The following observations help explain the figure.

**Idle state:** We remain in $s_1$ as long as all users are idle (probability $u_0$). We move to transmitting state if exactly one user requests access (probability $u_1$) and we move to collided state if two or more users request access (probability $1 - u_0 - u_1$).

**Transmitting state:** We move to idle state if all users are idle and if one or more users request access, we move to collided state since there will be no period of calm before the next transmission.

**Collided state:** We move to idle state if all users are idle. Any arriving requests keep the system in collided state.

Note in the figure that the channel can only move to the transmitting state in the next time step only if it is presently idle. If the channel is presently in the collided state, it cannot move to the transmitting state in the next time step since this would violate condition 8 above. The channel must first become idle and quiet since any user that attempts to transmit will only succeed in jamming the channel again. When the channel is transmitting, a period of calm has to be maintained for one step time and the channel then moves to the transmitting state. Otherwise, the channel will become collided again.

The probability that $k$ users request access during a time step is given by binomial statistics

$$u_k = \binom{N}{k} a^k (1-a)^{N-k} \tag{10.14}$$

The transition matrix of the system is

$$\mathbf{P} = \begin{bmatrix} u_0 & u_0 & u_0 \\ 1 - u_0 - u_1 & 1 - u_0 & 1 - u_0 \\ u_1 & 0 & 0 \end{bmatrix} \tag{10.15}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{10.16}$$

$$s_1 + s_2 + s_3 = 1 \tag{10.17}$$

The solution to the above two equations is simple:

$$\begin{aligned} s_1 &= u_0 \\ s_2 &= 1 - u_0 - u_0\,u_1 \\ s_3 &= u_0\,u_1 \end{aligned} \tag{10.18}$$

### *10.3.2 ALOHA Performance*

We are interested in the throughput of ALOHA which is given by

$$Th = s_3 = u_0\,u_1 \tag{10.19}$$

Substituting the values of $u_0$ and $u_1$ from (10.14), we get

$$Th = N\,a\,(1-a)^{2N-1} \tag{10.20}$$

For large $N$, the throughput can be expressed in terms of the input traffic as

$$Th = Nae^{-2Na} \tag{10.21}$$

The throughput is measured in units of frames/time step. The throughput in units of frames/s is

$$Th' = \frac{Th}{T} \tag{10.22}$$

The dotted line in Fig. 10.4 shows the variation of the throughput against the average number of users transmitting frames per time step for the case when $N = 10$ users. The solid line is for the slotted ALOHA network which is discussed in the next section.

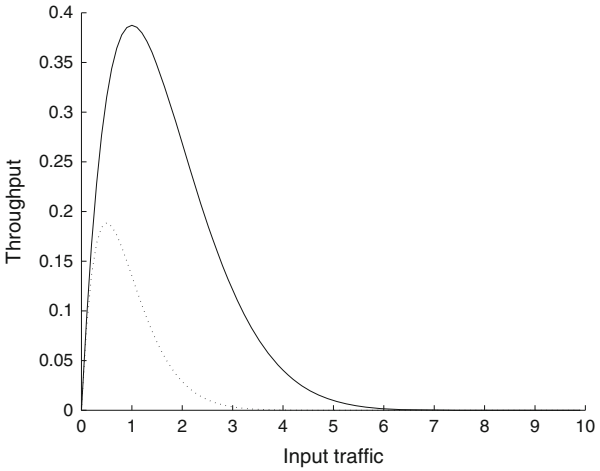



**Fig. 10.4** The throughput for ALOHA versus the average number of requests per time step for the case $N = 10$. *Dotted line* is for ALOHA and *solid line* is for slotted ALOHA

The input traffic is found from the binomial distribution

$$N_a(in) = N\ a \tag{10.23}$$

We define the *acceptance probability* or *efficiency* $p_a$ as the probability of a successful transmission for a given user. $p_a$ is given by

$$\begin{aligned} p_a &= \frac{Th}{N_a(in)} \\ &= (1-a)^{2N-1} \qquad (10.24) \\ &\approx e^{-2aN} \qquad (10.25) \end{aligned}$$

The last equation relies on the approximation

$$\lim_{n\to\infty} \left(1 - \frac{x}{n}\right)^n = e^n$$

The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i+1)(1-p_a)^i\ p_a \\ &= \frac{E_0}{p_a} \qquad (10.26) \end{aligned}$$

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$E/E_0 = -10 \log_{10} p_a \quad \text{dB} \tag{10.27}$$

$$\approx 20aN \log_{10} e = 8.7\, aN \quad \text{dB} \tag{10.28}$$

We see that average energy to transmit a frame increases exponentially with increasing traffic.

The average number of attempts before a successful transmission is

$$n_a = \sum_{n=0}^{\infty} n \ (1 - p_a)^n \ \ p_a \tag{10.29}$$

This evaluates to

$$n_a = \frac{1 - p_a}{p_a} \tag{10.30}$$

If the average duration of the random wait period is $\tau$ (in seconds), then the average wait for a frame before successful transmission is

$$W = n_a \times \tau = \frac{\tau\,(1 - p_a)}{p_a} \qquad \text{s} \tag{10.31}$$

Assume that the number of stations is fixed, in that case we can vary the arrival probability $a$ and investigate how the throughput varies with input traffic. The maximum throughput occurs when

$$\frac{d\ Th}{d\ a} = 0 \tag{10.32}$$

Differentiating (10.20) indicates that the maximum throughput for ALOHA occurs when

$$a_0 = \frac{1}{2\ N} \tag{10.33}$$

Thus, the maximum throughput is theoretically equal to

$$Th(\max) = \frac{1}{2}\left(1 - \frac{1}{2N}\right)^{2N-1} \tag{10.34}$$

The above equation gives the maximum throughput for any number of users $N$. A simple expression is obtained when the user population is very large $N \to \infty$:

$$Th(\max) = \frac{1}{2\,e}$$

$$= 18.394\,\% \tag{10.35}$$

In summary, maximum throughput occurs when $N \to \infty$ and we would have

$$Th = 0.18394 \quad \text{frames/time step} \tag{10.36}$$

$$N_a(in) = 0.5 \quad \text{frames/time step} \tag{10.37}$$

$$a_0 = 1/(2N) \tag{10.38}$$

$$p_a = Th/N_a(in) = 0.367 \tag{10.39}$$

This discrete-time, and very general, result compares extremely well with the throughput estimate obtained using fluid flow analysis [1] for a continuous-time system with Poisson traffic. Thus when the number of users increases, the transmission request probability must decrease in proportion. For example, when the system has $N = 50$ users, then maximum throughput is approximately 18.487 %, using Eq. (10.34), and the transmission request probability for maximum throughput must be $a \approx 0.01$. The problems at the end of the chapter confirm our predictions.

*Example 10.2.* Assume an ALOHA network supporting $N = 20$ users and each user issues a request per time step with probability $a = 0.01$. Find:

(a) The throughput
(b) The average number of time steps before successful transmission
(c) The maximum throughput
(d) The value for $a$ for maximum throughput

The performance figures are as follows

$$\begin{aligned} Th &= 0.1351 \text{ frames/time step} \\ n_a &= 0.4799 \text{ attempts} \\ Th(\max) &= 0.1839 \text{ frames/time step} \\ a_0 &= 0.025 \quad \text{for maximum throughput} \end{aligned}$$

■

## 10.4 Slotted ALOHA

Slotted ALOHA was proposed to improve the throughput of the original ALOHA [2]. As the name implies, time in slotted ALOHA is divided into slots and the value of one time step equals the frame time $T$. Users know about the

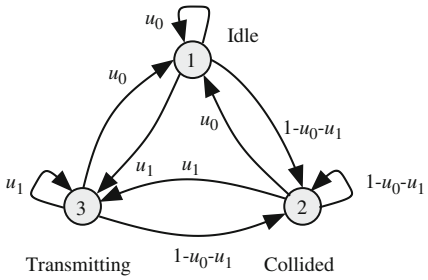



**Fig. 10.5** State transition diagram for the slotted ALOHA channel

start of a new slot through a synchronizing signal that is transmitted by a source. A user with data to send is only permitted to transmit at the start of a time step. This removes the chaos of pure ALOHA and improves the efficiency as we shall see below.

### *10.4.1 Modeling the Slotted ALOHA Network*

In this section we perform Markov chain analysis of the slotted ALOHA network. We employ the same assumptions that we used to model ALOHA in Sect. 10.3.1 with the only exception that users are allowed to transmit only at the start of a time step and not at any time as before.

Based on our assumptions the wireless channel can be in one of three states: *idle, collided*, or *transmitting*. Figure 10.5 shows the transition diagram for our Markov chain. Further, the time step value is naturally chosen equal to the slot time value. Now it is a good time for the reader to compare that figure with the pure ALOHA transition diagram of Fig. 10.3. What is the major difference?

The major difference is that the channel can move from collided state to transmitting state in one time step. This basically creates more chances for the channel to move to the transmitting state and this enhances the throughput and performance in general.

**Idle state:** We remain in $s_1$ as long as all users are idle (probability $u_0$). We move to transmitting state if exactly one user requests access (probability $u_1$) and we move to collided state if two or more users request access (probability $1 - u_0 - u_1$).

**Transmitting state:** We move to idle state if all users are idle. We move to transmitting state if exactly one user requests access (probability $u_1$) and we move to collided state if two or more users request access (probability $1 - u_0 - u_1$).

**Collided state:** We move to idle state if all users are idle. We move to transmitting state if exactly one user requests access (probability $u_1$) and we stay in collided state if two or more users request access (probability $1 - u_0 - u_1$).

The probability that $k$ users request access during a time step is given as before by binomial statistics

$$u_k = \binom{N}{k} a^k (1-a)^{N-k} \tag{10.40}$$

The transition matrix of the system is

$$\mathbf{P} = \begin{bmatrix} u_0 & u_0 & u_0 \\ 1-u_0-u_1 & 1-u_0-u_1 & 1-u_0-u_1 \\ u_1 & u_1 & u_1 \end{bmatrix} \tag{10.41}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{10.42}$$

$$s_1 + s_2 + s_3 = 1 \tag{10.43}$$

The solution to the above two equations is simple:

$$\begin{aligned} s_1 &= u_0 \\ s_2 &= 1 - u_0 - u_1 \\ s_3 &= u_1 \end{aligned} \tag{10.44}$$

Now it is a good time for the reader to compare the above equation with the corresponding equation for the equilibrium distribution vector of pure ALOHA in (10.18). Several observations can be made even before we get any numerical values:

1. Pure ALOHA and slotted ALOHA channels have the same probability of being in state $s_1$ (the idle state).
2. Slotted ALOHA has a *lower* probability of being in state $s_2$ (the collided state) compared to pure ALOHA.
3. Slotted ALOHA has a *higher* probability of being in state $s_3$ (the transmitting state) compared to pure ALOHA.

All these factors contribute to enhance the performance of slotted ALOHA.

### *10.4.2 Slotted ALOHA Performance*

We are interested in the throughput of slotted ALOHA which is given by

$$Th = s_3 = u_1 = N\ a\ (1-a)^{N-1} \tag{10.45}$$

A frame will be transmitted if only one user requests to access the channel irrespective of the previous state of the channel. For large $N$, the throughput can be expressed in terms of the input traffic as

$$Th = Nae^{-Na} \tag{10.46}$$

Slotted ALOHA has higher throughput compared to ALOHA by a factor of $u_0^{-1} = (1-a)^{-N}$. We expect the two network to perform almost the same for very low traffic conditions ($a \ll 1$). Slotted ALOHA will perform better than ALOHA by *orders of magnitude* for high values of traffic ($a \approx 1$) especially for systems with many users ($N \gg 1$).

The throughput in units of frames/s is

$$Th' = \frac{Th}{T} \tag{10.47}$$

The solid line in Fig. 10.4 shows the variation of the throughput against the average number of users transmitting frames per time step for the case when $N = 10$ users. The average number of users is given by the binomial distribution

$$N_a(in) = N\ a \tag{10.48}$$

We define the *acceptance probability* or *efficiency* $p_a$ as the probability of a successful transmission for a given user. $p_a$ is given by

$$\begin{aligned} p_a &= \frac{Th}{N_a(in)} \\ &= (1-a)^{N-1} && (10.49) \\ &\approx e^{-aN} && (10.50) \end{aligned}$$

The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i+1)(1-p_a)^i\ p_a \\ &= \frac{E_0}{p_a} && (10.51) \end{aligned}$$

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$\begin{aligned} E/E_0 &= -10 \log_{10} p_a \quad \text{dB} && (10.52) \\ &\approx 10aN \log_{10} e = 4.3\, aN \quad \text{dB} && (10.53) \end{aligned}$$

We see that average energy to transmit a packet increases exponentially with increasing traffic but at a rate half that of pure ALOHA.

The average number of attempts before a successful transmission is

$$n_a = \sum_{n=0}^{\infty} n \ (1 - p_a)^n \ \ p_a \tag{10.54}$$

This evaluates to

$$n_a = \frac{1 - p_a}{p_a} \tag{10.55}$$

If the average duration of the random wait period is $\tau$ (in seconds), then the average wait for a frame before successful transmission is

$$W = n_a \times \tau = \frac{\tau \, (1 - p_a)}{p_a} \qquad \text{s} \tag{10.56}$$

Assume that the number of stations is fixed, in that case we can vary the arrival probability $a$ and investigate how the throughput varies. The maximum throughput occurs when

$$\frac{d \ Th}{d \ a} = 0 \tag{10.57}$$

Differentiating (10.20) indicates that the maximum throughput for ALOHA occurs when

$$a_0 = \frac{1}{N} \tag{10.58}$$

Thus, the maximum throughput is theoretically equal to

$$Th(\max) = \left(1 - \frac{1}{N}\right)^{N-1} \tag{10.59}$$

The above equation gives the maximum throughput for any number of users $N$. A simple expression is obtained when the user population is very large $N \to \infty$:

$$\begin{aligned} Th(\max) &= \frac{1}{e} \\ &= 36.788\,\% \end{aligned} \tag{10.60}$$

In summary, maximum throughput occurs when $N \to \infty$ and we have

$$Th = 36.788\,\% \quad \text{frames/time step} \tag{10.61}$$

$$N_a(in) = 1 \quad \text{frames/time step} \tag{10.62}$$

$$a_0 = 1/N \tag{10.63}$$

$$p_a = Th/N_a(in) = 0.367 \tag{10.64}$$

This compares very well with the throughput estimate obtained using fluid flow analysis [1] where for a continuous-time system with Poisson traffic. Thus when the number of users increases, the transmission request probability must decrease in proportion. For example, when the system has $N = 100$ users, then maximum throughput is approximately 36.788 % and the transmission request probability must be $a \approx 0.01$.

Comparing these predictions with pure ALOHA we find that slotted ALOHA could support double the number of users and achieve double the throughput.

We note that at maximum throughput, pure ALOHA and slotted ALOHA have the same efficiency. This is a bit surprising since it was not previously reported in the literature. In fact, the efficiency for pure ALOHA is given by the expressions

$$p_a = (1 - a_0)^{2N-1}$$
$$a_0 = 1/2N$$

And the efficiency for slotted ALOHA is given by the expressions

$$p_a = (1 - a_0)^{N-1}$$
$$a_0 = 1/N$$

Both of these expressions evaluate to $e^{-1}$ as $N \to \infty$ and the arrival probability at maximum throughput is taken as $a_0 = 1/2N$ (for pure ALOHA) and $a_0 = 1/N$ (for slotted ALOHA).

Both systems show maximum efficiency at very low traffic. As traffic increases the efficiencies of both systems start to decrease. However, the efficiency of slotted ALOHA decreases at a slower rate compared to pure ALOHA.

*Example 10.3.* Repeat Example 10.2 assuming a slotted ALOHA network.

The performance figures are as follows:

$$\begin{aligned} Th &= 0.1652 \text{ frames/time step} \\ n_a &= 0.2104 \text{ attempts} \\ Th(\max) &= 0.3679 \text{ frames/time step} \\ a_0 &= 0.05 \quad \text{for maximum throughput} \end{aligned}$$

We note that the throughput of slotted ALOHA is slightly higher than pure ALOHA but it is not its double since we are far from the optimum traffic arrival probability for either systems.

Similarly, the average number of tries is less for slotted ALOHA. ■

## 10.5 CSMA/CD and the IEEE 802.3 (Ethernet) Protocol

The CSMA/CD and the IEEE 802.3 standard is used for *wired LANs* where the time required for *one bit* to travel between the two farthest stations (propagation time) is much smaller than the time required for *one frame* to be sent by the sender (transmission delay). The IEEE Standard 802.3 is based on CSMA/CD.

Signals on the channel travel very close to the speed of light and it takes a finite amount of time before all stations become aware that a channel is starting to access the medium. Therefore, a collision is said to take place when two or more stations start transmitting within the frame propagation delay. Because during this time, transmitting stations think that the medium is idle. When that happens, the two colliding stations *stop transmitting* and wait for a random amount of time before attempting to transmit again. This reduces the chance that the stations will once again transmit simultaneously. The maximum distance limitation for CSMA/CD is about 2,500 m (1.5 miles). At this value, the ratio of propagation delay to transmission delay is less than 0.1 [3].

To summarize, in CSMA/CD protocol all stations monitor the channel to determine when it is free. This is done by special *carrier sensing* circuits in each station. If the channel is busy, a station backs off and starts sensing the channel with probability $p$. This is called $p$-persistent CSMA/CD. The station refrains from transmitting on an idle channel with probability $1 - p$. This reduces the probability of collisions. If the channel is sensed free, the station starts to transmit. Transmitting stations monitor the signal on the channel and compare it to the transmitted signal to decide if a collision is taking place or not. This is done by special *collision detection* circuits. When the LAN contains $N$ stations, $p$ is chosen such that the product $N\ p < 1$ [3].

The IEEE 802.3 standard describes a 1-persistent CSMA/CD with exponential backoff strategy which is more commonly known as Ethernet.

At this point, it is worthwhile mentioning three different CSMA access strategies:

- 1-persistent CSMA
- Nonpersistent CSMA
- $p$-persistent CSMA

In a 1-persistent CSMA, a station with frame to send senses the channel. If the channel is sensed free, the frame is sent immediately. If the channel is busy, the station *continuously* monitors the channel and sends the frame when the channel is sensed idle. When a collision takes place, the station adopts a backoff strategy where

it waits for a random amount of time before it starts to sense the channel. The IEEE 802.3 (Ethernet) adopts this backoff strategy.

In a nonpersistent CSMA, a station with frame to send senses the channel. If the channel is sensed free, the frame is sent immediately. If the channel is busy, the station *waits* for a random amount of time before monitoring the channel and sends the frame when the channel is sensed idle. If the channel is sensed busy, the station repeats the random wait. Of course, when a channel suffers a collision, it adopts a backoff strategy where the collided user sends a frame only when the channel is not busy and a random wait period has expired.

In a $p$-persistent CSMA, a station with frame to send senses the channel. If the channel is sensed free, the frame is sent with probability $p$. The transmission is deferred for the next time slot with probability $1 - p$. This process is repeated until the frame is sent. The IEEE 802.11 (WiFi) adopts this transmission strategy.

### *10.5.1 CSMA/CD Model Assumptions*

A simple analysis of IEEE 802.3 was given in [4, 5]. A more complicated analysis of IEEE 802.3 can be found [6] which is by no means more accurate since it makes drastic assumptions about the channel states and the probability of a successful transmission. We follow here the middle ground and provide a tractable analysis making reasonable approximations.

Let us define $\tau_c$ to be the delay time required for a user to detect that a collision has taken place. This delay is approximately equal to the propagation delay $\tau_p$. We define $\tau_t$ to be the transmission delay for one frame. Typically $\tau_c \ll \tau_t$ since collision detection is done using fast electronic circuits. Therefore, periods of transmission are separated by one or more *contention minislots* [7]. Similar to ALOHA, a user could determine if there is contention or not during a time period equal to the propagation delay, i.e. $\tau_p \approx \tau_c$.

To start our analysis, we employ a set of assumptions for IEEE 802.3 model as follows:

1. Since the current state of the channel depends only on its immediate past history, we can model the channel using Markov chain analysis.
2. The states of the Markov chain represent the states of the channel: *idle*, *transmitting*, and *collided*.
3. The channel is shared among $N$ stations.
4. There is a single station class (equal priority).
5. The frame arrival probability per time step is $a$.
6. All transmitted frames have equal lengths.
7. A frame duration is equal to the transmission delay of a frame $\tau_t$.
8. The time step of the Markov chain is chosen equal to the collision detection delay, i.e. $T = \tau_c$.

9. We define $n$ as the ratio of transmission delay to propagation delay, i.e., $n = \tau_t/\tau_c$. We assume that $n > 1$ which is true for small LANs carrying long frames or operating at low transmission speeds.
10. The time required to detect a collision is less than or equal to the time step value $T$.
11. Probability that an idle station receives a frame for transmission during a frame transmission time (or frame duration) is $a$.
12. A $p$-persistent CSMA/CD is assumed. When $p = 1$, we get the model for the IEEE 802.3 Ethernet protocol.
13. A collided users transmits with probability $\gamma$ when it senses the channel free.
14. A station can have at most one message waiting for transmission.

### *10.5.2 CSMA/CD State Transition Diagram*

In this section we attempt to model the states of a tagged user. It is best to model the user as opposed to modeling the channel since the CSMA/CD medium access control protocol (MAC) specifies certain actions of the user such as backoff and persistence. These actions cannot be explicitly modelled by the channel.

Based on the above assumptions the user can be in one of the following states:

1. Idle state ($s_i$) where the user has no frames to send.
2. Wait state ($s_w$) where the user has a frame to send but sensed the channel busy.
3. Collided state ($s_c$) where the transmitting user just sensed a collision after the first transmission time step.
4. $n$ transmitting states ($s_{t_i}$) with $1 \leq i \leq n$.

Figure 10.6 shows the state diagram of CSMA/CD protocol. There are several transmitting states because the time required for transmitting one frame ($\tau_t$) is bigger

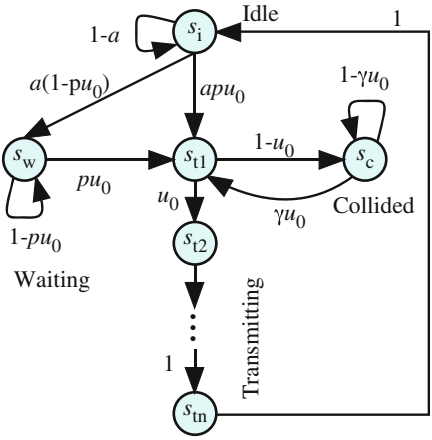



**Fig. 10.6** State transition diagram for the IEEE 802.3 CSMA/CD tagged user

than the propagation delay $\tau_p$. In the figure $u_0$ denotes the probability that all $N-1$ users, apart from the tagged user, will not transmit when the channel is free. The probability that a user will not start transmission even when the channel is sensed free is given by:

$$p_{idle} = (1 - a'p)s_i + (1 - p)s_w + (1 - \gamma)s_c \tag{10.65}$$

where $a' = a/n$ is the probability that a station requests a transmission during a time step, $p$ is the persistence probability and $\gamma$ is the probability that a collided user starts a transmission.

Based on that, excepting the tagged user, the probability all untagged users will not start transmission is given by:

$$u_0 = \left[(1 - a')s_i + (1 - p)s_w + (1 - \gamma)s_c\right]^{N-1} \tag{10.66}$$

We organize the distribution vector at equilibrium as follows.

$$\mathbf{s} = \left[\, s_i \;\; s_w \;\; s_c \;\; s_{t_1} \;\; s_{t_2} \;\cdots\; s_{t_n} \,\right]^t \tag{10.67}$$

The corresponding transition matrix of the user when $n = 3$ is given by

$$\mathbf{P} = \begin{bmatrix} 1-a' & 0 & 0 & 0 & 0\;1 \\ a'(1-pu_0) & 1-pu_0 & 0 & 0 & 0\;0 \\ 0 & 0 & 1-\gamma u_0 & 1-u_0 & 0\;0 \\ a'pu_0 & pu_0 & \gamma u_0 & 0 & 0\;0 \\ 0 & 0 & 0 & u_0 & 0\;0 \\ 0 & 0 & 0 & 0 & 1\;0 \end{bmatrix} \tag{10.68}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{10.69}$$

$$\sum \mathbf{s} = 1 \tag{10.70}$$

However, the terms in $\mathbf{P}$ depend on the state vector components. This constitutes a highly nonlinear set of equations. The solution for $\mathbf{s}$ is obtained through several techniques such as optimization or iterative techniques as follows:

1. Input the values of $p$, $\gamma$, $a$, $N$, and $n$.
2. Assume a trial value for state vector $\mathbf{s}$.
3. Start the iterations by obtaining the probability $u_0$
4. Substitute the value of $u_0$ to obtain an updated value $\mathbf{s}(updated)$.
5. Calculate the error $\mathbf{e} = \mathbf{s}(updated) - \mathbf{s}$ and the root mean square error $e_{rms}$.
6. Update the state vector

$$\mathbf{s} = \mathbf{s} + \alpha\,\mathbf{e}$$

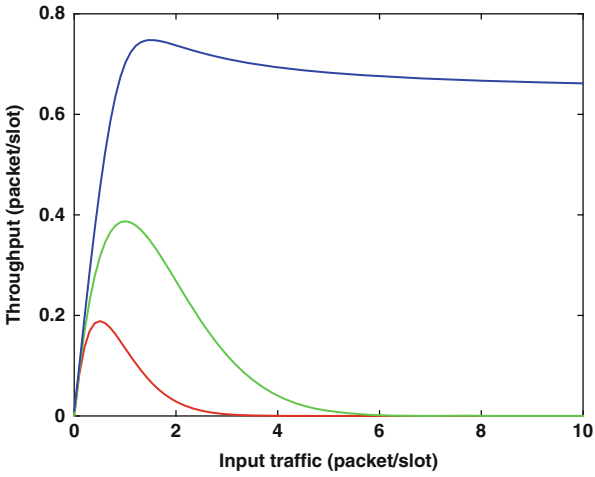



**Fig. 10.7** The throughput for persistent CSMA/CD versus the average input traffic when $n = 20$, $N = 10$, $p = 0.1$, and $\gamma = 0.1$. The *blue line* is the throughput of $p$-persistent CSMA/CD, the *green line* represents the throughput of slotted ALOHA, and the *red line* represents the throughput of pure ALOHA

where $\alpha$ is the update step size which is usually taken equal to 0.1 or even smaller.

7. Repeat the iterations starting at Step 3 and stop when $e_{rms}$ is below a certain value.

### *10.5.3 IEEE 802.3 (CSMA/CD) Protocol Performance*

The throughput is given by the equation

$$Th = nNT_n \tag{10.71}$$

Figure 10.7 shows the throughput of the $p$-persistent CSMA/CD when $n = 20$, $N = 10$, $p = 0.1$, and $\gamma = 0.1$. The blue line is the throughput of $p$-persistent CSMA/CD, the green line represents the throughput of slotted ALOHA, and the red line represents the throughput of pure ALOHA. For a given value of $N$, the two main parameters that affect the throughput are $n$ and $\gamma$. Increasing $n$ improves the throughput since little time is allocated for collisions detection. At high traffic, the users are mostly in the collided state. Increasing $\gamma$ will result in more collisions and less throughput since most users will attempt to transmit when they sense the channel free. Decreasing $\gamma$ will also result in more collisions and less throughput since collided users will not attempt to transmit even when the channel is sensed free. The optimum value for $\gamma = 1/N$ through experimentation. Unlike ALOHA or Slotted ALOHA, CSMA/CD shows high throughput even when traffic level is high. This is a definite advantage that resulted from two factors:

1. Adoption of backoff strategy for collided users
2. Monitoring the channel state and only transmitting when the channel is sensed idle.

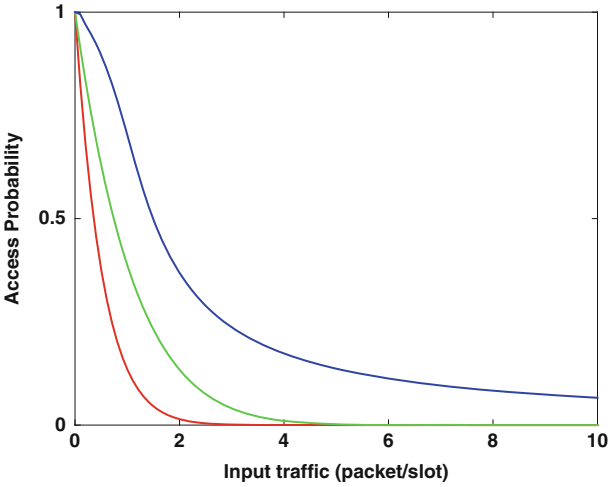



**Fig. 10.8** Access probability for $p$-persistent CSMA/CD versus the average input traffic when $n = 20$, $N = 10$, $p = 0.1$, and $\gamma = 0.1$. The *blue line* is the access probability of persistent CSMA/CD, the *green line* represents the access probability of slotted ALOHA, and the *red line* represents the access probability of pure ALOHA

The access probability for the user in the $p$-persistent CSMA/CD protocol is given by

$$p_a = \frac{Th}{Na} \tag{10.72}$$

Figure 10.8 shows the access probability of the persistent CSMA/CD protocol. The blue line is the access probability of $p$-persistent CSMA/CD, the green line represents the access probability of slotted ALOHA, and the red line represents the access probability of pure ALOHA. We notice that $p$-persistent CSMA/CD has the highest value for $p_a$ followed by slotted ALOHA then ALOHA, as expected.

The average number of attempts for a successful transmission is

$$\begin{aligned} n_a &= \sum_{i=0}^{\infty} i \ (1 - p_a)^i \ p_a \\ &= \frac{1 - p_a}{p_a} \end{aligned} \tag{10.73}$$

Figure 10.9 shows the frame delay for $p$-persistent CSMA/CD versus the average input traffic when $n = 20$, $N = 10$, and $\gamma = 0.1$. The blue line is the delay of $p$-persistent CSMA/CD, the green line represents the delay of slotted ALOHA, and the red line represents the delay of pure ALOHA. We notice that persistent CSMA/CD shows the least delay and that the delay shows signs of saturation for high values of input traffic.

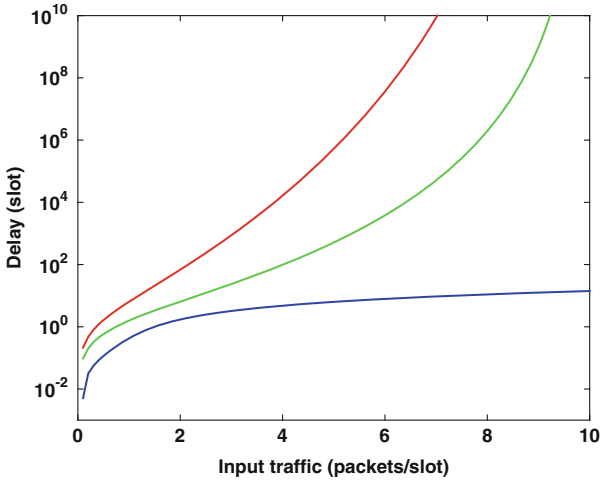



**Fig. 10.9** Frame delay for $p$-persistent CSMA/CD versus the average input traffic when $n = 20$, $N = 10$, $p = 0.1$, and $\gamma = 0.1$. The *blue line* is the delay of $p$-persistent CSMA/CD, the *green line* represents the delay of slotted ALOHA, and the *red line* represents the delay of pure ALOHA

The average energy required for a successful transmission is given by:

$$\begin{aligned} E &= E_0 + \frac{E_0}{n} \sum_{i=0}^{\infty} (i-1)\,(1-p_a)^i\; p_a \\ &= E_0 \left(1 + \frac{1}{np_a}\right) \end{aligned} \tag{10.74}$$

where $E_0$ is the energy required to send one frame. Figure 10.10 shows the frame energy for $p$-persistent CSMA/CD versus the average input traffic when $n = 20$, $N = 10$, $p = 0.1$, and $\gamma = 0.1$. The blue line is the energy of $p$-persistent CSMA/CD, the green line represents the energy of slotted ALOHA, and the red line represents the energy of pure ALOHA. We notice that $p$-persistent CSMA/CD shows the least energy and that the energy shows signs of saturation for high values of input traffic. The low energy required to transmit a frame is due to the fact that CSMA/CD detects a collision in one slot and back off when it occurs. On the other hand, ALOHA protocols detect a collision at the end of the frame transmission and this leads to more lost energy.

## 10.6 Carrier Sense Multiple Access–Collision Avoidance

CSMA/CA is used in *wireless LANs* where a transmitting station is unable to determine if a collision occurred while transmitting or not. Collision detection, as is employed in Ethernet, cannot be used for the radio frequency transmissions.

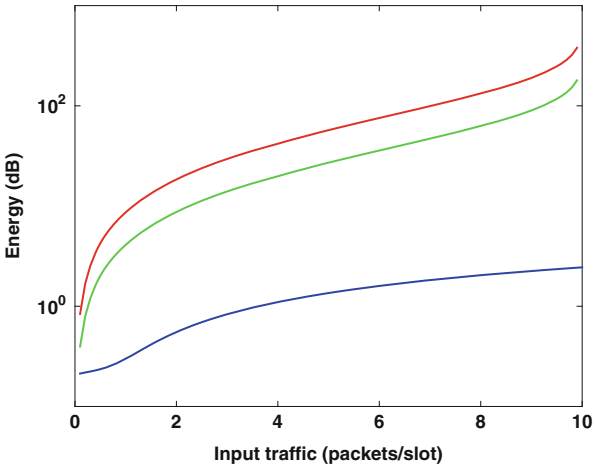



**Fig. 10.10** Average energy required to send a frame versus the average input traffic when $n = 20$, $N = 10$, $p = 0.1$, and $\gamma = 0.1$. The *blue line* is the energy of $p$-persistent CSMA/CD, the *green line* represents the energy of slotted ALOHA, and the *red line* represents the energy of pure ALOHA

The reason for this is that when a node is transmitting it cannot hear any other node in the system which may be transmitting, since its own signal will drown out other signals arriving at the node. A station will ultimately know when a collision has taken place by reception of negative acknowledgment or by timeout mechanisms.

An ad hoc network is a collection of communicating nodes that do not have established infrastructure or centralized administration [8]. CSMA/CA protocol is useful in ad hoc networks where access to the network is decentralized since each station coordinates its own decisions for accessing the medium. There is no central access point to coordinate activities of all stations. Thus ad hoc networks are simpler to implement and to modify. The price for this simplicity is that ad hoc networks are prone to collisions.

### *10.6.1 CSMA/CA Model Assumptions*

Let us define $\tau_p$ to be the propagation delay between users, and $\tau_t$ to be the transmission delay for one frame. To start our analysis, we employ a set of assumptions for CSMA/CA model as follows:

1. Since the current state of the channel depends only on its immediate past history, we can model the channel using Markov chain analysis.
2. The states of the Markov chain represent the states of the channel: *idle*, *transmitting*, and *collided*.
3. The channel is shared among $N$ stations.

4. There is a single station class (equal priority).
5. The frame arrival probability per time step is $a$.
6. All transmitted frames have equal lengths.
7. A frame duration is equal to the transmission delay of a frame $\tau_t$.
8. The time step of the Markov chain is chosen equal to the propagation delay, i.e. $T = \tau_p$.
9. We define $n$ as the ratio of transmission delay to propagation delay, i.e., $n = \tau_t/\tau_p$. We assume that $n > 1$ which is true for small LANs carrying long frames or operating at low transmission speeds.
10. A $p$-persistent CSMA/CA is assumed.
11. A station can have at most one message waiting for transmission.

### *10.6.2 CSMA/CA State Transition Diagram*

In this section we attempt to model the states of a tagged user. It is best to model the user as opposed to modeling the channel since the CSMA/CD medium access control protocol (MAC) specifies certain actions of the user such as backoff and persistence. These actions cannot be explicitly modelled by the channel.

Based on the above assumptions the user can be in one of the following states:

1. Idle state ($s_i$) where the user has no frames to send.
2. Wait state ($s_w$) where the user has a frame to send but sensed the channel busy.
3. $n$ transmitting states ($s_{t_i}$) with $1 \leq i \leq n$.
4. $n$ collided states ($s_{c_i}$) with $1 \leq i \leq n$ where the transmitting user sensed a collision after the end of transmission.

Figure 10.11 shows the state diagram of CSMA/CA protocol. There are several transmitting states because the time required for transmitting one frame ($\tau_t$) is bigger than the propagation delay $\tau_p$. There are also several collided states since a user continues to transmit after a collision has taken place. In the figure $u_0$ denotes the probability that all $N-1$ users, apart from the tagged user, will not transmit when the channel is free. The probability that a user will not start transmission even when the channel is sensed free is given by:

$$p_{idle} = (1 - a'p)s_i + (1 - p)s_w + (1 - \gamma)s_{c_1} \tag{10.75}$$

where $a' = a/n$ is the probability that a station requests a transmission during a time step, $p$ is the persistence probability when the user is waiting for the channel to be free, and $\gamma$ is the probability that a collided user starts a transmission. Based on that, excepting the tagged user, the probability all untagged users will not start transmission is given by:

$$u_0 = \left[(1 - a'p)s_i + (1 - p)s_w + (1 - \gamma)s_{c1}\right]^{N-1} \tag{10.76}$$

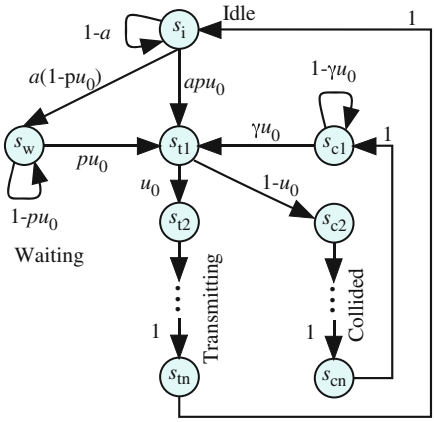



**Fig. 10.11** State transition diagram for the CSMA/CA tagged user

We organize the distribution vector at equilibrium as follows.

$$\mathbf{s} = \left[ s_i \; s_w \; s_{t_1} \; s_{t_2} \cdots s_{t_n} \; s_{c_1} \; s_{c_2} \cdots s_{c_n} \right]^t \tag{10.77}$$

The corresponding transition matrix of the channel for the case when $n = 3$ is given by

$$\mathbf{P} = \begin{bmatrix} 1-a & 0 & 0 & 0\,1 & 0 & 0\,0 \\ a(1-pu_0) & 1-pu_0 & 0 & 0\,0 & 0 & 0\,0 \\ apu_0 & pu_0 & 0 & 0\,0 & \gamma u_0 & 0\,0 \\ 0 & 0 & u_0 & 0\,0 & 0 & 0\,0 \\ 0 & 0 & 0 & 0\,0 & 0 & 0\,0 \\ 0 & 0 & 1-u_0 & 0\,0 & 1-\gamma u_0 & 0\,1 \\ 0 & 0 & 0 & 0\,0 & 0 & 0\,0 \\ 0 & 0 & 0 & 1\,0 & 0 & 1\,0 \end{bmatrix} \tag{10.78}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{10.79}$$

$$\sum \mathbf{s} = 1 \tag{10.80}$$

However, the terms in $\mathbf{P}$ depend on the state vector components. This constitutes a highly nonlinear set of equations. The solution for $\mathbf{s}$ is obtained through several techniques such as optimization or iterative techniques as follows:

1. Input the values of $p$, $\gamma$, $a$, $N$, and $n$.
2. Assume a trial value for state vector $\mathbf{s}$.
3. Start the iterations by obtaining the probability $u_0$
4. Substitute the value of $u_0$ to obtain an updated value $\mathbf{s}(updated)$.

5. Calculate the error $\mathbf{e} = \mathbf{s}(updated) - \mathbf{s}$ and the root mean square error $e_{rms}$.
6. Update the state vector

$$\mathbf{s} = \mathbf{s} + \alpha\, \mathbf{e}$$

where $\alpha$ is the update step size which is usually taken equal to 0.1 or even smaller.
7. Repeat the iterations starting at Step 3 and stop when $e_{rms}$ is below a certain value.

### *10.6.3 CSMA/CA Protocol Performance*

The throughput is given by the equation:

$$Th = nNT_n \tag{10.81}$$

Figure 10.12 shows the throughput of $p$-persistent CSMA/CA. Figure 10.12a shows the throughput when $n = 20$, $N = 10$, $\gamma = 0.01$, and $p = 0.5$. The blue line is the throughput of $p$-persistent CSMA/CA, the green line represents the throughput of slotted ALOHA, and the red line represents the throughput of pure ALOHA.

It is interesting to compare Fig. 10.7 for CSMA/CD and Fig. 10.12 for CSMA/CA. The latter protocol shows lower throughput for the same set of parameters as the former. This is due to the fact that CSMA/CA keeps transmitting frames even when collisions have taken place. Therefore precious bandwidth and time is wasted transmitting frames while CSMA/CD stops the transmission

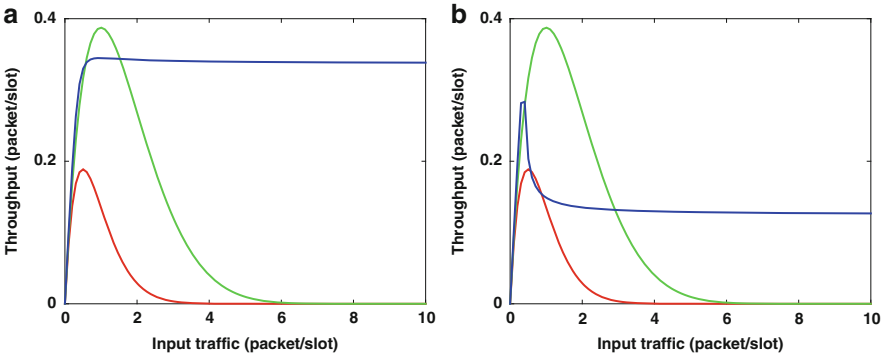



**Fig. 10.12** The throughput for CSMA/CA versus the average input traffic. (**a**) is case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. (**b**) is case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.1$. The *blue line* is the throughput of $p$-persistent CSMA/CA, the *green line* represents the throughput of slotted ALOHA, and the *red line* represents the throughput of pure ALOHA

promptly. Unlike ALOHA or Slotted ALOHA, CSMA/CA shows high throughput even when traffic level is high, similar to CSMA/CD. This is a definite advantage that resulted from two factors:

1. Adoption of backoff strategy for collided users
2. Monitoring the channel state and only transmitting when the channel is sensed idle.
3. Adopting a persistence probability $p$.

The access probability for a user in the CSMA/CA protocol is given by

$$p_a = \frac{Th}{Na} \tag{10.82}$$

Figure 10.13 shows the access probability of CSMA/CA. (a) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. (b) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.1$. The blue line is the access probability of $p$-persistent CSMA/CA, the green line represents the access probability of slotted ALOHA, and the red line represents the access probability of pure ALOHA.

The average number of attempts for a successful transmission is

$$\begin{aligned} n_a &= \sum_{i=0}^{\infty} i\ (1 - p_a)^i\ p_a \\ &= \frac{1 - p_a}{p_a} \end{aligned} \tag{10.83}$$

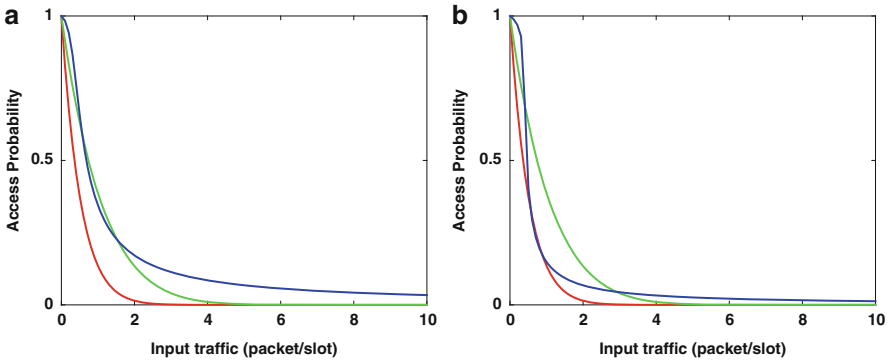



**Fig. 10.13** Access probability of CSMA/CA. (**a**) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. (**b**) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.1$. The *blue line* is the access probability of $p$-persistent CSMA/CA, the *green line* represents the access probability of slotted ALOHA, and the *red line* represents the access probability of pure ALOHA

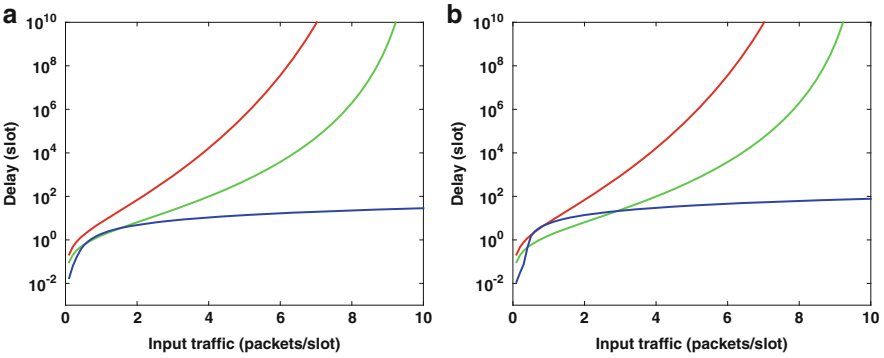



**Fig. 10.14** Frame delay of CSMA/CA. (**a**) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. (**b**) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.1$. The *blue line* is the delay of $p$-persistent CSMA/CA, the *green line* represents the delay of slotted ALOHA, and the *red line* represents the delay of pure ALOHA

Figure 10.14 shows the delay of the CSMA/CA protocol when $n = 50$, $N = 10$, and $p = 0.5$. Figure 10.14a is the case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. Figure 10.14b is the case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.1$. The blue line is the delay of $p$-persistent CSMA/CA, the green line represents the delay of slotted ALOHA, and the red line represents the delay of pure ALOHA.

The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i+1)(1-p_a)^i \, p_a \\ &= \frac{E_0}{p_a} \end{aligned} \tag{10.84}$$

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$E/E_0 = -10 \log_{10} p_a \quad \text{dB} \tag{10.85}$$

Figure 10.15 shows the average energy needed to transmit a frame of the CSMA/CA protocol when $n = 50$, $N = 10$, and $p = 0.5$. Figure 10.15a is the case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. Figure 10.15b is the case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.1$. The blue line is the energy of $p$-persistent CSMA/CA, the green line represents the energy of slotted ALOHA, and the red line represents the energy of pure ALOHA.

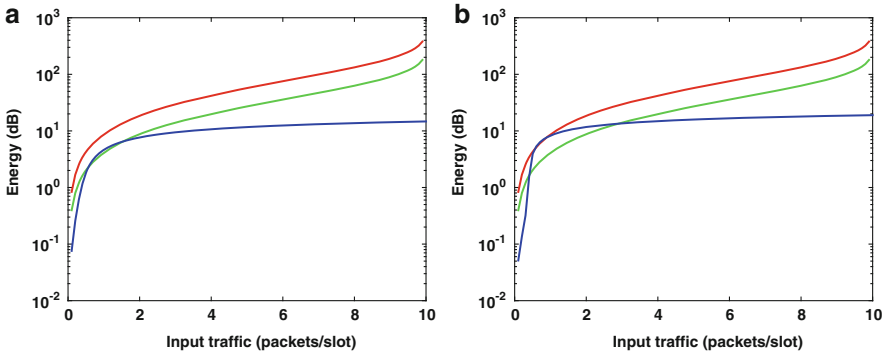



**Fig. 10.15** Average energy to transmit a frame for the CSMA/CA. (**a**) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. (**b**) Case when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.1$. The *blue line* is the energy of $p$-persistent CSMA/CA, the *green line* represents the energy of slotted ALOHA, and the *red line* represents the energy of pure ALOHA

## 10.7 Problems

### *ALOHA Network*

**10.1.** Use Eq. (10.20) to find the maximum value for the throughput of an ALOHA network and the value of $a$ at the maximum.

**10.2.** Using the results of Sect. A.6 in Appendix A on page 543 prove that the maximum throughput of the ALOHA network approaches the value $1/2e$ as $N \to \infty$.

**10.3.** Assume an ALOHA network that is operating at its optimum conditions. What is the average number of attempts for a user to be able to transmit a frame under these conditions?

**10.4.** Assume an ALOHA network where the frame length is a multiple of some unit of length with an upper limit on the maximum frame size. Draw a possible transition diagram for such system and write down the corresponding state transition matrix.

**10.5.** Assume an ALOHA network where the propagation delay is bigger than the frame time $T$. What would be a good choice for the time step of the Markov chain? Draw a possible transition diagram for such system and write down the corresponding state transition matrix.

**10.6.** Assume there are 25 users in an ALOHA network. What is the transmission request probability $a$ that corresponds to maximum throughput and what is the value of the maximum throughput?

**10.7.** Assume there are 25 users in an ALOHA network and the probability that a user request access is $a = 0.06$. What is the throughput of the channel and what is the probability that a user will successfully transmit frame after three unsuccessful attempts?

**10.8.** What is the average number of unsuccessful attempts before a user can transmit a frame in the above problem?

## *Slotted ALOHA Network*

**10.9.** What is the major difference between ALOHA and slotted ALOHA?

**10.10.** What are the major differences between the transition diagrams of ALOHA and slotted ALOHA?

**10.11.** Write down the expressions for the steady state distribution vectors for ALOHA and slotted ALOHA and comment on their similarities and differences. Explain why slotted ALOHA is expected to perform better than ALOHA.

**10.12.** Write down the ratio of throughput for slotted ALOHA compared to ALOHA. Approximate the expression for the limits when $a \ll 1$ and $a \approx 1$.

**10.13.** Use Eq. (10.45) to find the maximum value for the throughput of a slotted ALOHA network and the value of $a$ at the maximum.

**10.14.** Using the results of Sect. A.6 in Appendix A on page 543 prove that the maximum throughput of the slotted ALOHA network approaches the value $1/e$ as $N \to \infty$.

**10.15.** Assume a slotted ALOHA network that is operating at its optimum conditions. What is the average number of attempts for a frame to be transmitted under these conditions?

**10.16.** Assume a slotted ALOHA network where the frame length is a multiple of some unit of length with an upper limit on the maximum frame size. Draw a possible transition diagram for such system and write down the corresponding transition matrix.

**10.17.** Assume a slotted ALOHA network where the propagation delay is bigger than the frame time $T$. Draw a possible transition diagram for such system and write down the corresponding transition matrix.

**10.18.** Assume there are 25 users in a slotted ALOHA network. What is the transmission request probability $a$ that corresponds to maximum throughput and what is the value of the maximum throughput?

**10.19.** Assume there are 25 users in a slotted ALOHA network and the probability that a user request access is $a = 0.06$. What is the throughput of the channel and what is probability that a user will successfully transmit frame after three unsuccessful attempts?

**10.20.** What is the average number of unsuccessful attempts before a user can transmit a frame in the above problem?

**10.21.** Compare the maximum throughput values for ALOHA and slotted ALOHA and the level of activity for the sources under these conditions assuming the same number of user in both systems.

**10.22.** Assume an ALOHA network and a slotted ALOHA network. Both systems support $N$ users and each user is active with probability $a = 0.02$. What is the optimum value of $N$ for maximum throughput for both systems?

## *p-Persistent CSMA/CD and IEEE 802.3 (Ethernet)*

**10.23.** Consider Fig. 10.6 for the $p$-persistent CSMA/CD MAC protocol. What would the state diagram be when a binary exponential backoff strategy is used?

**10.24.** Consider the state transition diagram in Fig. 10.6. At saturation we have $a \approx 1$ and we can solve for the states in terms of the probability $u_0$, $n$, $p$, and $\gamma$. Find expressions for the states and the throughput.

**10.25.** Assume an IEEE 802.3 network where the frame length is not constant. In that case the number of time slots required by the transmitted frames could take between $n_{min}$ to $n_{max}$ slots.

(a) Draw the resulting state transition diagram.
(b) Indicate on the diagram the transition probabilities.
(c) Write down the state transition matrix.

**10.26.** Assume an IEEE 802.3 network in which a transmitting user could move on to transmit another frame without turning to the idle state. The probability of this event happening is $c$.

(a) Identify the possible states of this Markov chain.
(b) Study the state transition probabilities and write down the state transition matrix.

**10.27.** Assume in the 802.3 that a collided station attempts a retransmission for a limited number of times (assume two only). If it fails after two attempts, it returns back to being idle. Analyze this situation.

### *CSMA/CA*

**10.28.** Consider Fig. 10.11 for the $p$-persistent CSMA/CA MAC protocol. What would the state diagram be when a binary exponential backoff strategy is used?

**10.29.** Consider the state transition diagram in Fig. 10.11. At saturation we have $a \approx 1$ and we can solve for the states in terms of the probability $u_0$, $n$, $p$, and $\gamma$. Find expressions for the states and the throughput.

**10.30.** Assume a CSMA/CA network where the frame length is not constant. In that case the number of time slots required by the transmitted frames could take between $n_{min}$ to $n_{max}$ slots.

(a) Draw the resulting state transition diagram.
(b) Indicate on the diagram the transition probabilities.
(c) Write down the state transition matrix.

**10.31.** Assume a CSMA/CA network where the frame length is not constant. In that case the number of time slots required by the transmitted frames could take between $n_{min}$ to $n_{max}$ slots.

(a) Draw the resulting state transition diagram.
(b) Indicate on the diagram the transition probabilities.
(c) Write down the state transition matrix.

**10.32.** Assume a CSMA/CA network in which a transmitting user could move on to transmit another frame without turning to the idle state. The probability of this event happening is $c$.

(a) Identify the possible states of this Markov chain.
(b) Study the state transition probabilities and write down the state transition matrix.

**10.33.** Assume in the CSMA/CA protocol that a collided station attempts a retransmission for a limited number of times (assume two only). If it fails after two attempts, it returns back to being idle. Analyze this situation.

## References

1. A.S. Tanenbaum, *Computer Networks* (Prentice Hall PTR, Upper Saddle River, 1996)
2. L. Roberts, Extension of frame communication technology to a hand held personal terminal. National Computer Conference, AFIPS (1973), pp. 711–716
3. S. Keshav, *An Engineering Approach to Computer Networks* (Addison-Wesley, Reading, 1997)
4. R.M. Metcalfe, D.R. Boggs, Ethernet: distributed frame switching for local computer networks. Commun. ACM **19**, 395–404 (1976)
5. F.A. Tobagi, Analysis of a two-hop centralized frame radio network: part I—slotted aloha. IEEE Trans. Commun. **COM-28**, 196–207 (1980)

6. M.E. Woodward, *Communication and Computer Networks* (IEEE Computer Society Press, Los Alamitos, 1994)
7. A. Leon-Garcia, I. Widjaja, *Communication Networks* (McGraw-Hill, New York, 2000)
8. D.B. Jhonson, D.A. Maltz, Dynamic source routing in ad hoc wireless networks, in *Mobile Computing*, ed. by T. Imielinski, H. Korth (Kluwer Academic Publishers, Dordrecht, 1996)

# Chapter 11
# Modeling IEEE 802.11 (WiFi) Protocol

## 11.1 Introduction

The IEEE 802.11 (WiFi) is the most widely used medium access control protocol for wireless local area networks (LANs). Being a wireless protocol, it is based on $p$-persistent CSMA/CA discussed in Chap. 10. The only difference between WiFi and CSMA/CA is that when a user has a packet or frame to send and senses channel is free, it will unconditionally go through a random wait before it attempts a transmission. On the other hand, CSMA/CA under the same conditions will immediately send the frame. WiFi underwent several versions such as 802.11a, 802.11b, 802.11g, 802.11n, 802.11ac, 802.11ad, and 802.11ah. We shall discuss here simplified versions of the protocol since our intention is to illustrate how such protocol is using the features common in all versions. IEEE 802.11 wireless LAN standard is used for infrastructure as well as ad hoc networks.

Infrastructure wireless networks have a central controller called *access point* (AP) that coordinates medium access among the users. This part of the protocol is referred to as Point Coordination Function (PCF) and it occupies a short time period at the start of each transmitted frame as shown in Fig. 11.1. The frame starts with a PCF period which is a time period to enable prioritized access for control messages and time-critical traffic [1–3]. This form of *centralized medium access scheme* enables the IEEE 802.11 to offer some quality of service (QoS) guarantees through implementation of a *scheduling algorithm* at the AP.

Ad hoc wireless networks do not have a central controller. Instead, each node or user attempts to access the shared medium on its own. This part of the protocol is referred to as Distributed Coordination Function (DCF) and it follows the PCF period of each transmitted frame as shown in Fig. 11.1. This is a form of *distributed reservation scheme* that could provide statistical QoS guarantees.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_11

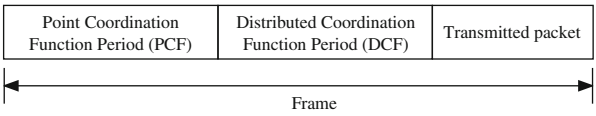



**Fig. 11.1** Location of PCF and DCF periods in the frame of IEEE 802.11 protocol

## 11.2 IEEE 802.11: Basic DCF for Ad Hoc Wireless LANs

The DCF MAC part of IEEE 802.11 standard is based on CSMA-CA (listen-before-talk) with rotating backoff window [1]. When a node receives a frame to be transmitted, it chooses a random backoff value which determines the amount of time the node must wait until it is allowed to transmit its frame. A node stores this backoff value in a *backoff counter*. During periods in which the channel is clear, the node decrements its backoff counter. (When the channel is busy it does not decrement its backoff counter.) When the backoff counter reaches zero, the node transmits the frame. Since the probability that two nodes will choose the same backoff factor is small, collisions between frames are minimized. Collision detection, as is employed in Ethernet, cannot be used for the radio frequency transmissions of IEEE 802.11. The reason for this is that when a node is transmitting it cannot hear any other node in the system which may be transmitting, since its own signal will drown out any others arriving at the node.

In that sense, 802.11 could be classified as CSMA/CA but with provisions for reducing the chance of collisions through adoption of the reservation slots using the backoff counters. The slots have the effect of ensuring that a reduced number of users compete for access to the channel during any given reservation slot.

Figure 11.2 shows the DCF part of the IEEE 802.11 frame. After the PCF period (i.e., SIFS), there is the DCF period (i.e., DIFS) which is a contention window that is divided into reservation slots. The figure shows six such slots. The duration of a reservation slot depends on the propagation delay between stations. The rest of the frame is dedicated to transmitting the frames.

A station that intends to transmit senses if the channel is busy. It will then wait for the end of the current transmission and the PCF delay. It then randomly selects a reservation slot within the backoff window. The figure shows that a station in time reservation slot 2 starts transmitting a frame since the channel was not used during reservation slots 0 and 1.

Collisions occur when two or more stations select the same reservation slot. If another station started transmission at an earlier reservation slot, the station freezes its backoff counter and waits for the remaining contents of this counter after the current transmission ends.

The basic DCF uses a two-way handshaking data exchange mechanism shown in Fig. 11.3.

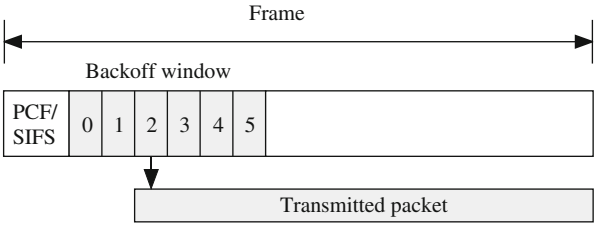



**Fig. 11.2** Basic DCF of the IEEE 802.11 frame

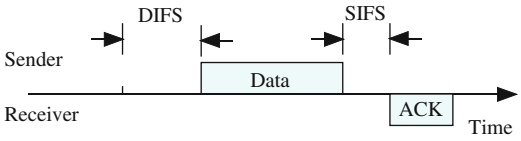



**Fig. 11.3** Two-way handshaking data exchange mechanism for the basic DCF function of the IEEE 802.11 WiFi MAC protocol

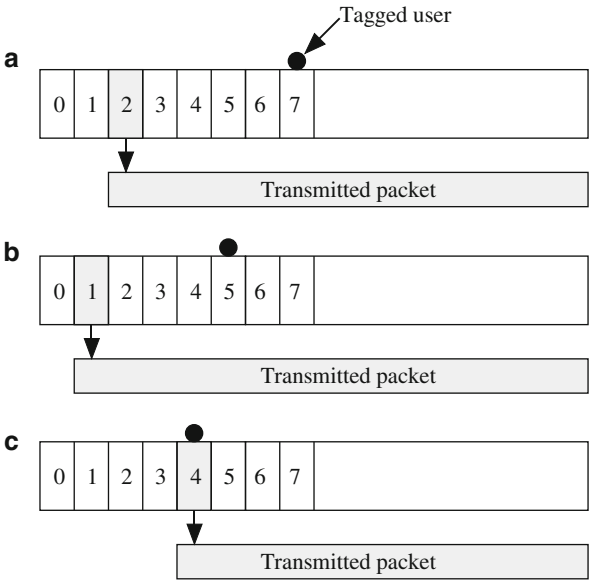



**Fig. 11.4** IEEE 802.11 MAC scheme

### *11.2.1 Markov Chain Modeling of the Basic DCF*

We consider the behavior of one user, which we term the tagged user. Figure 11.4 shows the IEEE 802.11 MAC scheme as viewed by a certain user (called the *tagged user*). Figure 11.4 indicates that the tagged user, as indicated by the black circle, randomly selected reservation slot 7 to start transmission. So its backoff counter contains the value 7 now. However, another user starts transmission at reservation slot 2 as indicated by the grey box. Since the channel was quite for two reservation slots (slot 0 and 1), the backoff counter of the tagged user will contain the value 5 at the end of the current frame.

Figure 11.4b shows next frame. However, another user at reservation slot 1 started transmission. Since the channel was quite for one reservation slots (slot 0), the backoff counter of the tagged user will contain the value 4 at the end of the current frame.

Figure 11.4c shows next frame. The tagged user is successful in starting transmission since the channel was quite for four reservation slots (0, 1, 2, and 3).

Modeling the behavior of the backoff counter can certainly be done using Markov chains. However, we can equally model such random behavior through a backoff, or persistence, probability $p$ when the user is waiting for the channel to be free before attempting to transmit a packet. We will follow this approach in the following subsection.

### *11.2.2 IEEE 802.11: Basic DCF Model Assumptions*

We employ the following simplifying assumptions.

1. Since the current state of the user depends only on its immediate past history, we can model the user using Markov chain analysis.
2. The states of the Markov chain represent the states of the user: idle, waiting, transmitting, and collided.
3. There are $N$ equal priority users in the network. By network we mean a single-hop network or the nodes within the transmission range of a particular node.
4. We replace the waiting backoff window with a transmit probability $p$ when the channel is sensed idle.
5. The duration of one time step in the contention window is roughly taken equal to the maximum expected propagation delay $\tau_p$ plus the time it takes a station to sense the presence of a carrier. This time is called the Distributed Interframe Spacing (DIFS).
6. The Markov chain time step is taken equal to the DIFS period.
7. The ratio of frame transmission delay to contention window delay is $n > 1$.
8. All frames have equal lengths such that a frame takes $n$ time steps to be transmitted.
9. Probability that an idle station receives a frame for transmission during a frame period is $a$.
10. A station can have at most one message waiting for transmission.
11. Collided users employ a random backoff strategy with transmit probability $\gamma$ when the channel is sensed idle.

Figure 11.5 shows the state transition diagram for the IEEE 802.11 WiFi tagged user when the basic DCF protocol is used. There are several transmitting states because the time required for transmitting one frame ($\tau_t$) is bigger than the propagation delay $\tau_p$. There are also several collided states since a user continues to transmit after a collision has taken place.

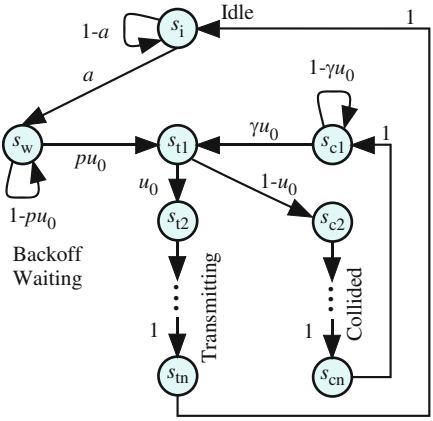



**Fig. 11.5** State transition diagram for the IEEE 802.11 WiFi tagged user when the basic DCF protocol is used

In the figure $u_0$ denotes the probability that all $N-1$ users, apart from the tagged user, will not transmit when the channel is free. The probability that a user will not start transmission even when the channel is sensed free is given by:

$$p_{idle} = s_i + (1-p)s_w + (1-\gamma)s_{c_1} \tag{11.1}$$

where $a' = a/n$ is the probability that a station requests a transmission during a time step, $p$ is the persistence probability when the user is waiting for the channel to be free, and $\gamma$ is the probability that a collided user starts a transmission. Based on that, excepting the tagged user, the probability all untagged users will not start transmission is given by:

$$u_0 = [s_i + (1-p)s_w + (1-\gamma)s_c]^{N-1} \tag{11.2}$$

We organize the distribution vector at equilibrium as follows.

$$\mathbf{s} = \left[ s_i \;\; s_w \;\; s_{t_1} \;\; s_{t_2} \;\cdots\; s_{t_n} \;\; s_{c_1} \;\; s_{c_2} \;\cdots\; s_{c_n} \right]^t \tag{11.3}$$

The corresponding transition matrix of the channel for the case when $n = 3$ is given by:

$$\mathbf{P} = \begin{bmatrix} 1-a' & 0 & 0 & 0\,1 & 0 & 0\,0 \\ a' & 1-pu_0 & 0 & 0\,0 & 0 & 0\,0 \\ 0 & pu_0 & 0 & 0\,0 & \gamma u_0 & 0\,0 \\ 0 & 0 & u_0 & 0\,0 & 0 & 0\,0 \\ 0 & 0 & 0 & 0\,0 & 0 & 0\,0 \\ 0 & 0 & 1-u_0 & 0\,0 & 1-\gamma u_0 & 0\,1 \\ 0 & 0 & 0 & 0\,0 & 0 & 0\,0 \\ 0 & 0 & 0 & 1\,0 & 0 & 1\,0 \end{bmatrix} \tag{11.4}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{11.5}$$

$$\sum \mathbf{s} = 1 \tag{11.6}$$

However, the terms in **P** depend on the state vector components. This constitutes a highly nonlinear set of equations. The solution for **s** is obtained through several techniques such as optimization or iterative techniques as follows:

1. Input the values of $p$, $\gamma$, $a$, $N$, and $n$.
2. Assume a trial value for state vector **s**.
3. Start the iterations by obtaining the probability $u_0$.
4. Substitute the value of $u_0$ to obtain an updated value $\mathbf{s}(updated)$.
5. Calculate the error $\mathbf{e} = \mathbf{s}(updated) - \mathbf{s}$ and the root mean square error $e_{rms}$.
6. Update the state vector

$$\mathbf{s} = \mathbf{s} + \alpha\,\mathbf{e}$$

   where $\alpha$ is the update step size which is usually taken equal to 0.1 or even smaller.
7. Repeat the iterations starting at Step 3 and stop when $e_{rms}$ is below a certain value.

### *11.2.3 IEEE 802.11 WiFi: Basic DCF Protocol Performance*

The throughput is given by the equation:

$$Th = nNT_n \tag{11.7}$$

Figure 11.6 shows the throughput of IEEE 802.11 WiFi for the IEEE 802.11/DCF protocol versus the average input traffic when $n = 20$, $N = 10$, $p = 0.8$, and $\gamma = 0.01$. The black line is the throughput of IEEE 802.11 WiFi, the blue line is the throughput of $p$-persistent CSMA/CA with the same parameters as the WiFi, the green line is the throughput of slotted ALOHA, and the red line is the throughput of pure ALOHA.

At low input traffic, the WiFi basic DCF protocol throughput is comparable to $p$-persistent CSMA/CA but then it becomes a bit smaller than it since users with a frame to transmit have to wait before accessing the channel.

Changing the value of $p$ has little effect on the throughput. However, the collided backoff probability drastically affects the throughput. When $\gamma$ is increased from 0.01 to 0.05, the throughput of both CSMA/CA and IEEE 802.11 is reduced. Increasing $\gamma$ to 0.1 will reduce the IEEE 802.11 drastically, a very small value.

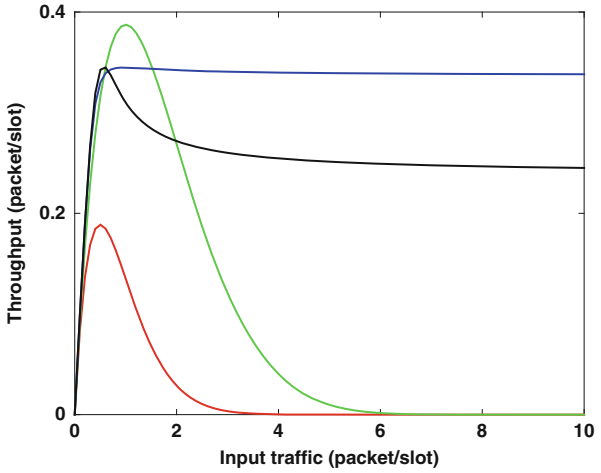
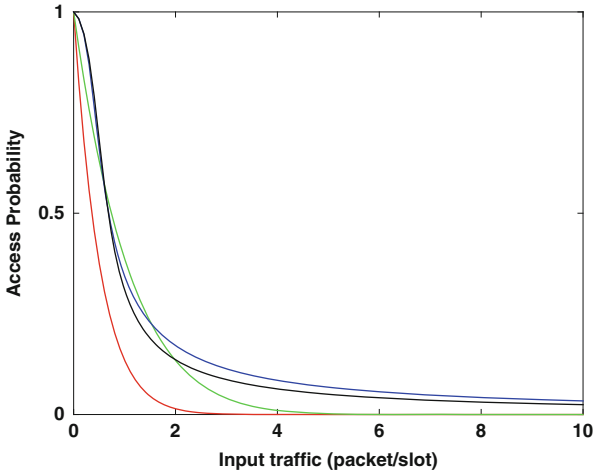



**Fig. 11.6** Throughput for the IEEE 802.11/DCF protocol versus the average input traffic when $n = 20$, $N = 10$, $p = 0.8$, and $\gamma = 0.01$. The *black line* is the throughput of IEEE 802.11 WiFi, the *blue line* is the throughput of $p$-persistent CSMA/CA, the *green line* is the throughput of slotted ALOHA, and the *red line* is the throughput of pure ALOHA

**Fig. 11.7** Access probability of IEEE 802.11 basic DCF when $n = 20$, $N = 10$, $p = 0.8$, and $\gamma = 0.01$. The *black line* is the access probability of IEEE 802.11 basic DCF, the *blue line* is the access probability of $p$-persistent CSMA/CA, the *green line* is the access probability of slotted ALOHA, and the *red line* is the access probability of pure ALOHA




The user access probability is given by:

$$p_a = \frac{Th}{Na} \tag{11.8}$$

Figure 11.7 shows the access probability of IEEE 802.11 when $n = 20$, $N = 10$, $p = 0.8$, and $\gamma = 0.01$. Again, we see that $p_a$ for IEEE 802.11 basic DCF is equal to or a bit smaller than $p_a$ for CSMA/CA.

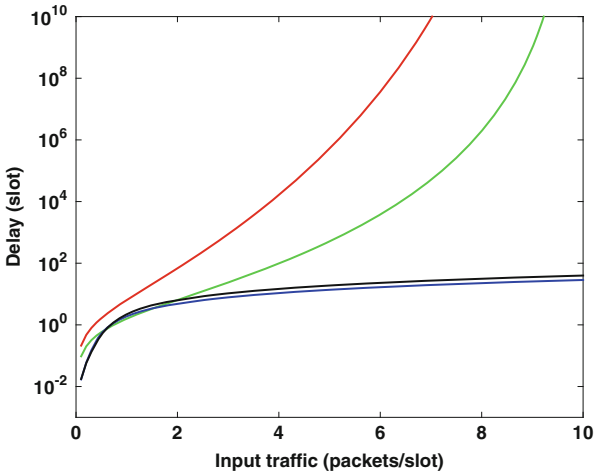



**Fig. 11.8** Frame delay of IEEE 802.11 basic DCF when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. The *black line* is the delay of IEEE 802.11 basic DCF, the *blue line* is the delay of $p$-persistent CSMA/CA, the *green line* is the delay of slotted ALOHA, and the *red line* is the delay of pure ALOHA

The average number of attempts for a successful transmission is

$$\begin{aligned} n_a &= \sum_{i=0}^{\infty} i \ (1 - p_a)^i \ p_a \\ &= \frac{1 - p_a}{p_a} \end{aligned} \tag{11.9}$$

Figure 11.8 shows the delay of the IEEE 802.11 basic DCF protocol when $n = 50$, $N = 10$, and $p = 0.5$ when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. The black line is the delay of IEEE 802.11 basic DCF, the blue line is the delay of $p$-persistent CSMA/CA, the green line is the delay of slotted ALOHA, and the red line is the delay of pure ALOHA.

The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i + 1)(1 - p_a)^i \ p_a \\ &= \frac{E_0}{p_a} \end{aligned} \tag{11.10}$$

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$E/E_0 = -10 \log_{10} p_a \quad \text{dB} \tag{11.11}$$

Figure 11.9 shows the average energy needed to transmit a frame of the IEEE 802.11 basic DCF protocol when $n = 20$, $N = 10$, $p = 0.8$, and $\gamma = 0.01$. The black

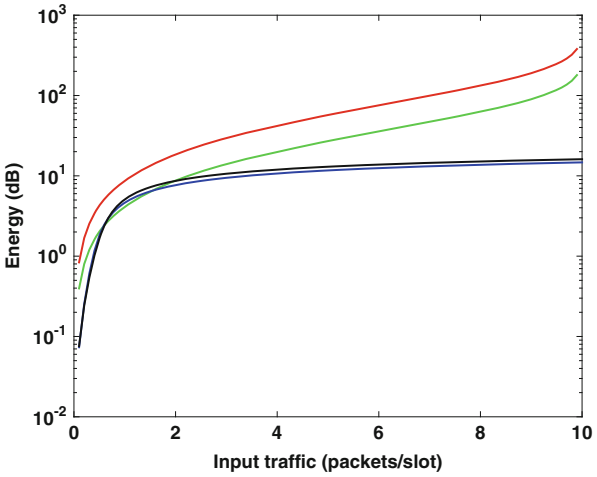



**Fig. 11.9** Average energy to transmit a frame for the IEEE 802.11 basic DCF when $n = 20$, $N = 10$, $p = 0.8$, and $\gamma = 0.01$. The *black line* is the energy of IEEE 802.11 basic DCF, the *blue line* is the energy of $p$-persistent CSMA/CA, the *green line* is the energy of slotted ALOHA, and the *red line* is the energy of pure ALOHA

line is the energy of IEEE 802.11 WiFi, the blue line is the energy of $p$-persistent CSMA/CA, the green line is the energy of slotted ALOHA, and the red line is the energy of pure ALOHA.

## 11.3 IEEE 802.11: DCF Using RTS/CTS for Ad Hoc Wireless LANs (MACA)

As was mentioned in Sect. 11.2, the basic DCF function is based on CSMA/CA for MAC functionality. That is why we notice that the system throughput was equal to, or slightly below, the CSMA/CA throughput. We also saw that in Chap. 10 that CSMA/CD protocol has higher throughput compared to CSMA/CA. This is the main motivation for attempting to modify the IEEE 802.11 DCF function to incorporate the CSMA/CD protocol. This is achieved through the use of four-way handshaking protocol as shown in Fig. 11.10.

A station with a packet to send will first send a *request to send* packet (RTS) when it senses the channel is free for a minimum of DIFS time. If the RTS packet is successfully received without suffering collisions, the intended receiver will issue a *clear to send* packet (CTS). After this, the sender will commend to send the data and wait for an acknowledgment (ACK) for the receiver.

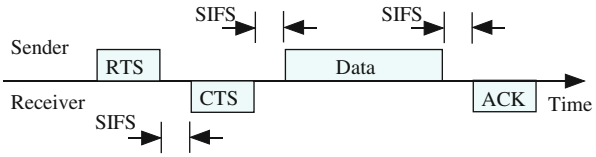



**Fig. 11.10** Four-way handshaking RTS/CTS (MACA) protocol for IEEE 802.11 WiFi DCF function

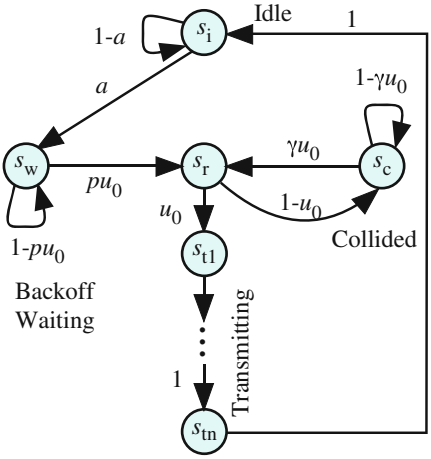



**Fig. 11.11** State transition diagram for the IEEE 802.11 WiFi tagged user when the RTS/CTS DCF protocol is used

### *11.3.1 Modeling DCF Using RTS/CTS*

To derive a simple model for WiFi DCF using RTS/CTS, we employ a set of assumptions similar to those employed in Sect. 11.2.2. The only difference is that the RTS/CTS mechanism is used to start assure no collisions will take place. We note that the time a station requires to determine if a collision occurred is the duration of the RTS/CTS packets. This is definitely shorter than the data duration. Thus the RTS/CTS mechanism mimics the CSMA/CD and hopefully the performance will improve over that of the basic DCF function.

Figure 11.11 shows the state transition diagram for the IEEE 802.11 WiFi tagged user when the RTS/CTS DCF protocol is used. There are several transmitting states because the time required for transmitting one frame ($\tau_t$) is bigger than the propagation delay $\tau_p$. There is only one collided state since a user will not transmit after a collision has taken place.

In the figure $u_0$ denotes the probability that all $N-1$ users, apart from the tagged user, will not transmit when the channel is free. The probability that a user will not start transmission even when the channel is sensed free is given by:

$$p_{idle} = s_i + (1-p)s_w + (1-\gamma)s_{c_1} \tag{11.12}$$

where $p$ is the persistence probability when the user is waiting for the channel to be free and $\gamma$ is the probability that a collided user starts a transmission. Based on that, excepting the tagged user, the probability all untagged users will not start transmission is given by:

$$u_0 = [s_i + (1-p)s_w + (1-\gamma)s_c]^{N-1} \tag{11.13}$$

We organize the distribution vector at equilibrium as follows:

$$\mathbf{s} = \left[ s_i \;\; s_w \;\; s_r \;\; s_c \;\; s_{t_1} \;\; s_{t_2} \;\; \cdots \;\; s_{t_n} \right]^t \tag{11.14}$$

The corresponding transition matrix of the channel for the case when $n = 3$ is given by:

$$\mathbf{P} = \begin{bmatrix} a & 1-pu_0 & 0 & 0 & 0\,0\,0 \\ 0 & pu_0 & 0 & \gamma u_0 & 0\,0\,0 \\ 0 & 0 & 1-u_0 & 1-\gamma u_0 & 0\,0\,0 \\ 0 & 0 & u_0 & 0 & 0\,0\,0 \\ 0 & 0 & 0 & 0 & 1\,0\,0 \\ 0 & 0 & 0 & 0 & 0\,1\,0 \end{bmatrix} \tag{11.15}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{11.16}$$

$$\sum \mathbf{s} = 1 \tag{11.17}$$

However, the terms in $\mathbf{P}$ depend on the state vector components. This constitutes a highly nonlinear set of equations. The solution for $\mathbf{s}$ is obtained through several techniques such as optimization or iterative techniques as follows:

1. Input the values of $p$, $\gamma$, $a$, $N$, and $n$.
2. Assume a trial value for state vector $\mathbf{s}$.
3. Start the iterations by obtaining the probability $u_0$.
4. Substitute the value of $u_0$ to obtain an updated value $\mathbf{s}(updated)$.
5. Calculate the error $\mathbf{e} = \mathbf{s}(updated) - \mathbf{s}$ and the root mean square error $e_{rms}$.
6. Update the state vector

$$\mathbf{s} = \mathbf{s} + \alpha\,\mathbf{e}$$

   where $\alpha$ is the update step size which is usually taken equal to 0.1 or even smaller.
7. Repeat the iterations starting at Step 3 and stop when $e_{rms}$ is below a certain value.

### 11.3.2 IEEE 802.11 WiFi: RTS/CTS Protocol Performance

The throughput is given by the equation:

$$Th = nNT_n \tag{11.18}$$

Figure 11.12 shows the throughput for the IEEE 802.11 RTS/CTS protocol versus the average input traffic when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The black line is the throughput of IEEE 802.11 RTS/CTS, the blue line is the throughput of $p$-persistent CSMA/CD, the green line is the throughput of slotted ALOHA, and the red line is the throughput of pure ALOHA.

At low input traffic, the WiFi RTS/CTS protocol throughput is comparable to $p$-persistent CSMA/CD but then it becomes a bit smaller than it since users with a frame to transmit have to wait before accessing the channel.

Changing the value of $p$ has little effect on the throughput. However, the collided backoff probability drastically affects the throughput. When $\gamma$ is increased from 0.01 to 0.05, the throughput of both CSMA/CD and IEEE 802.11 RTS/CTS is reduced.

The user access probability is given by:

$$p_a = \frac{Th}{Na} \tag{11.19}$$

Figure 11.13 shows the access probability of IEEE 802.11 RTS/CTS when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The black line is the throughput of IEEE 802.11

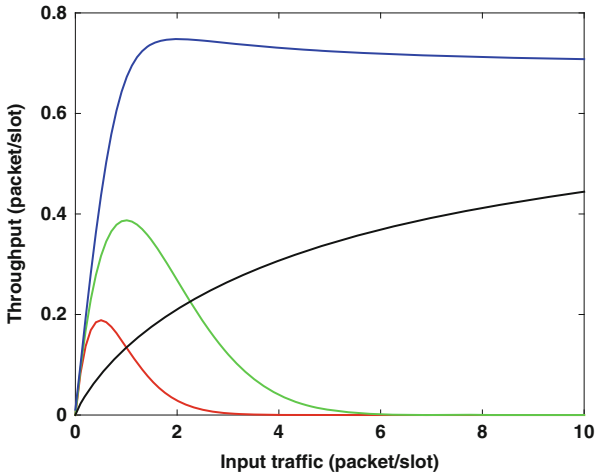



**Fig. 11.12** Throughput for the IEEE 802.11 RTS/CTS protocol versus the average input traffic when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The *black line* is the throughput of IEEE 802.11 RTS/CTS, the *blue line* is the throughput of $p$-persistent CSMA/CD, the *green line* is the throughput of slotted ALOHA, and the *red line* is the throughput of pure ALOHA

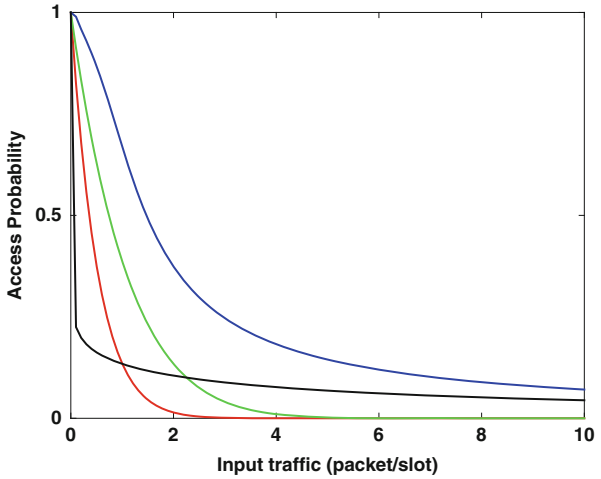



**Fig. 11.13** Access probability of IEEE 802.11 RTS/CTS when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The *black line* is the throughput of IEEE 802.11 RTS/CTS, the *blue line* is the throughput of $p$-persistent CSMA/CD, the *green line* is the throughput of slotted ALOHA, and the *red line* is the throughput of pure ALOHA

RTS/CTS, the blue line is the throughput of $p$-persistent CSMA/CD, the green line is the throughput of slotted ALOHA, and the red line is the throughput of pure ALOHA. Again, we see that $p_a$ for IEEE 802.11 RTS/CTS is almost equal to the access probability for the $p$-persistent CSMA/CD protocol.

The average number of attempts for a successful transmission is

$$\begin{aligned} n_a &= \sum_{i=0}^{\infty} i \ (1 - p_a)^i \ p_a \\ &= \frac{1 - p_a}{p_a} \end{aligned} \tag{11.20}$$

Figure 11.14 shows the delay of the IEEE 802.11 RTS/CTS protocol when $n = 50$, $N = 10$, and $p = 0.5$ when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The black line is the delay of IEEE 802.11 WiFi, the blue line is the delay of $p$-persistent CSMA/CD, the green line is the delay of slotted ALOHA, and the red line is the delay of pure ALOHA.

The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i + 1)(1 - p_a)^i \ p_a \\ &= \frac{E_0}{p_a} \end{aligned} \tag{11.21}$$

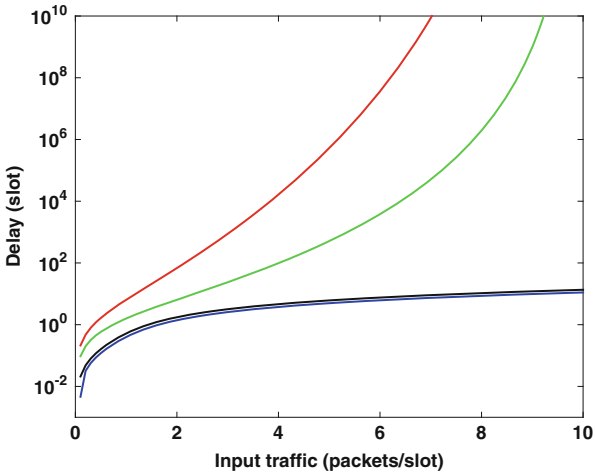



**Fig. 11.14** Frame delay of CSMA/CA when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The *black line* is the delay of IEEE 802.11 RTS/CTS, the *blue line* is the delay of $p$-persistent CSMA/CD, the *green line* is the delay of slotted ALOHA, and the *red line* is the delay of pure ALOHA

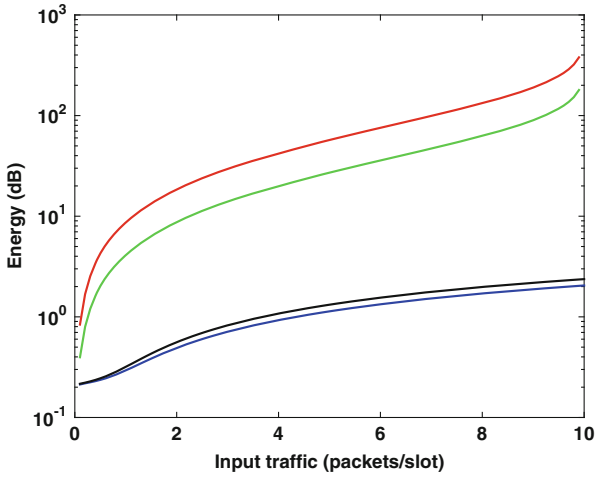



**Fig. 11.15** Average energy to transmit a frame for the IEEE 802.11 RTS/CTS when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The *black line* is the energy of IEEE 802.11 RTS/CTS, the *blue line* is the energy of $p$-persistent CSMA/CD, the *green line* is the energy of slotted ALOHA, and the *red line* is the energy of pure ALOHA

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$E/E_0 = -10 \log_{10} p_a \quad \text{dB} \tag{11.22}$$

Figure 11.15 shows the average energy required to transmit a packet for the IEEE 802.11 RTS/CTS protocol when $n = 20$, $N = 10$, $p = 0.05$, and $\gamma = 0.01$. The

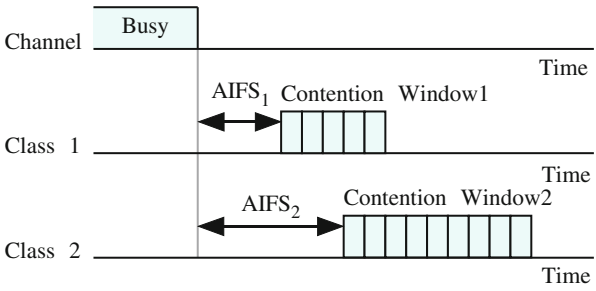



**Fig. 11.16** AIFS channel access for the IEEE 802.11e EDCA (WMM)

black line is the energy of IEEE 802.11 RTS/CTS, the blue line is the energy of $p$-persistent CSMA/CD, the green line is the energy of slotted ALOHA, and the red line is the energy of pure ALOHA.

## 11.4 IEEE 802.11e EDCA (WMM) Protocol

The IEEE802.11e standard was introduced to support QoS. The new standard defines the Hybrid Coordination Function (HCF) for supporting QoS over the wireless communication. The HCF uses two access modes: the Enhanced Distributed Channel Access (EDCA) for ad hoc wireless networks and Hybrid Coordination function Control Channel Access (HCCA) for infrastructure wireless networks. The contention-based EDCA replaces the DCF of the standard 802.11 and the contention-free HCCA replaces the PCF of the standard 802.11. Sometimes EDCA is called WiFi Multi Media (WMM) and is essentially DCF with four priority classes: background, best effort, video, and audio.

The basic idea for QoS support is to provide different minimum and maximum backoff slots for the service classes. The minimum backoff value dictates a deterministic delay before the user is able to start the backoff timer access the channel. The minimum backoff delay is called Arbitration Inter-Frame Space (AIFS). Figure 11.16 shows the AIFS channel access for the IEEE 802.11e EDCA (WMM).

Note in the figure that Class 1 service is the higher priority class since its contention window starts earlier and the size of its backoff counter is smaller. Both these factors give class 1 service better chance of accessing the service before the lower priority class.

### *11.4.1 Modeling the IEEE 802.11e EDCA (WMM) Protocol*

Let us make few simplifying assumptions:

1. All users can communicate together using one hop. This assumption eliminates the hidden terminal problem.

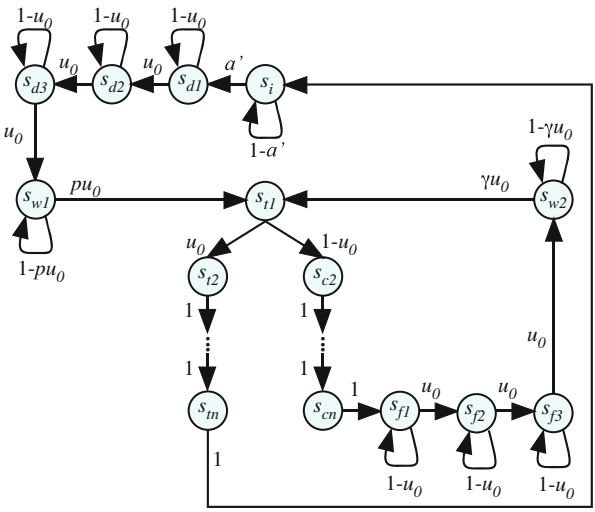



**Fig. 11.17** State transition diagram for the tagged user in class 1 or class 2 when the IEEE 802.11.e EDCA (WMM) protocol is used

2. There are only two classes of service with class 1 being the higher priority class.
3. The number of users in class 1 is $N_1$ and the number of users in class 2 is $N_2$.
4. The probability that a class 1 user has a packet to send in one frame time is $a_1$. $a_2$ is a similar probability for class 2 users.
5. The AIFS delay for class 1 is $d_1$ slots and $d_2 > 1_1$ for class 2.
6. The transmit opportunity (TXOP) is the same for both classes of service which corresponds to same packet size $n$.
7. The probability that a class 1 uncollided user transmits when the channel is free is $p_1$ and $p_2$ for class 2.
8. A collided class 1 user attempts a retransmission with probability $\gamma_1$ and $\gamma_2$ for class 2 user.

Based on the above assumptions we can derive a state transition diagram for either class of service as shown in Fig. 11.17. The figure can be considered an extension of the IEEE 802.11basic DCF MAC protocol and bears some resemblance to the CSMA/CA protocol. However, to support QoS, there are two sets of delay states. The first set $s_{d_i}$ $(1 \leq i \leq m)$ belongs to the idle users that just got a packet to send. This set of states provides a deterministic delay for the uncollided users before they even try to send a frame when the channel becomes free. Note that the delay counter is decremented only when the channel is sensed free.

The second set of delay states is $s_{f_i}(1 \leq i \leq m)$ provides a deterministic delay for the collided users before they even try to send a frame when the channel becomes free. Note that the delay counter is decremented only when the channel is sensed free. Without this delay, and at heavy traffic conditions, most users would be in the collided state and the system would revert to the basic DCF function.

In the figure $u_0$ denotes the probability that all users (except the tagged user) will not transmit when the channel is free. To derive this probability we need to find

the probability that a user is idle and will not transmit. The probability that a user in class 1 or class 2 will not start transmission even with the channel is free is given by:

$$p_{idle} = s_i + \sum_{i=1}^{m} s_{d_i} + (1-p)s_{w_1} + \sum_{i=1}^{m} s_{f_i} + (1-\gamma)s_{w_2} \tag{11.23}$$

The probability that all users (excepting the tagged user) will not start a transmission is given by:

$$u_0 = p_{idle,1}^{N_1} \times p_{idle,2}^{N_2} \tag{11.24}$$

Assuming that AIFS $m = 2$ and $n = 3$, we organize the distribution vector at equilibrium as follows:

$$\mathbf{s} = \begin{bmatrix} s_i & s_{d_1} & s_{d_2} & s_{w_1} & s_{t_1} & s_{t_2} & s_{t_3} & s_{c_2} & s_{c_3} & s_{f_1} & s_{f_2} & s_{w_2} \end{bmatrix}^t \tag{11.25}$$

The corresponding transition matrix of the channel for the case when AIFS = 2 and $n = 3$ is given by:

$$\mathbf{P} = \begin{bmatrix} 1-a' & 0 & 0 & 0 & 0 & 0\,1\,0\,0 & 0 & 0 & 0 \\ a' & 1-u_0 & 0 & 0 & 0 & 0\,0\,0\,0 & 0 & 0 & 0 \\ 0 & u_0 & 1-u_0 & 0 & 0 & 0\,0\,0\,0 & 0 & 0 & 0 \\ 0 & 0 & u_0 & 1-pu_0 & 0 & 0\,0\,0\,0 & 0 & 0 & 0 \\ 0 & 0 & 0 & pu_0 & 0 & 0\,0\,0\,0 & 0 & 0 & \gamma u_0 \\ 0 & 0 & 0 & 0 & u_0 & 0\,0\,0\,0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1\,0\,0\,0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1-u_0 & 0\,0\,0\,0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0\,0\,1\,0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0\,0\,0\,1 & 1-u_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0\,0\,0\,0 & u_0 & 1-u_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0\,0\,0\,0 & 0 & u_0 & 1-\gamma u_0 \end{bmatrix} \tag{11.26}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{11.27}$$

$$\sum \mathbf{s} = 1 \tag{11.28}$$

However, the terms in **P** depend on the state vector components. This constitutes a highly nonlinear set of equations. The solution for **s** is obtained through several techniques such as optimization or iterative techniques as follows:

1. Input the values of $p_1$, $p_2\gamma_1$, $\gamma_2$, $a_1$, $a_2$, $N_1$, $N_2$, $n$, and $m$.
2. Assume a trial value for state vector **s**.
3. Start the iterations by obtaining the probability $u_0$.
4. Substitute the value of $u_0$ to obtain an updated value $\mathbf{s}(updated)$.
5. Calculate the error $\mathbf{e} = \mathbf{s}(updated) - \mathbf{s}$ and the root mean square error $e_{rms}$.
6. Update the state vector

$$\mathbf{s} = \mathbf{s} + \alpha\, \mathbf{e}$$

   where $\alpha$ is the update step size which is usually taken equal to 0.1 or even smaller.
7. Repeat the iterations starting at Step 3 and stop when $e_{rms}$ is below a certain value.

### *11.4.2 IEEE 802.11.e EDCA (WMM) Protocol Performance*

The throughput is given by the equation:

$$Th = \left[ s_{t_1} + \sum_{i=2}^{n} s_{t_i} \right] N \tag{11.29}$$

Figure 11.18 shows the throughput of IEEE 802.11.e EDCA protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10$, $n = 20$, $m_1 = 5$, $m_2 = 10$, $p_1 = 0.5$, $p_2 = 0.05$, $\gamma_1 = 0.01$, and $\gamma_2 = 0.001$. The top black line is the throughput of IEEE 802.11.e EDCA class 1, the bottom black line is the throughput of IEEE 802.11e EDCA class 2, the red line is the throughput of the IEEE 802.11 basic DCF protocol, the green line is the throughput of slotted ALOHA, and the blue line is the throughput of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol.

We note that the class 1 throughput is comparable to the basic DCF protocol and that at most traffic levels the throughput of both is close to the maximum value of the slotted ALOHA protocol. The chosen values for delay states $d_i$ and $f_i$ in Fig. 11.17 apparently have no significant effect on the class 1 performance in comparison with the basic DCF protocol.

The class 2 EDCA protocol has almost one-half the throughput of the class 1 protocol due to the chosen parameter values. That is why class 2 throughput is close to the peak of the pure ALOHA protocol.

The user access probability is given by:

$$p_a = \frac{Th}{Na} \tag{11.30}$$

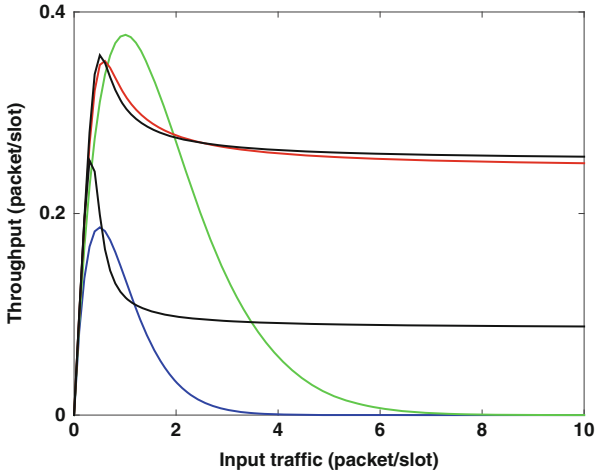



**Fig. 11.18** Throughput of IEEE 802.11.e EDCA (WMM) protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10$, $n = 20$, $m_1 = 5$, $m_2 = 10$, $p_1 = 0.5$, $p_2 = 0.05$, $\gamma_1 = 0.01$, and $\gamma_2 = 0.001$. The *top black line* is the throughput of IEEE 802.11.e EDCA class 1, the *bottom black line* is the throughput of IEEE 802.11e EDCA class 2, the *red line* is the throughput of the IEEE 802.11 basic DCF protocol, the *green line* is the throughput of slotted ALOHA, and the *blue line* is the throughput of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol

Figure 11.19 shows the access probability of IEEE 802.11.e EDCA protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10, n = 20$, $m_1 = 5$, $m_2 = 10$, $p_1 = 0.5$, $p_2 = 0.05$, $\gamma_1 = 0.01$, and $\gamma_2 = 0.001$. The top black line is the throughput of IEEE 802.11.e EDCA class 1, the bottom black line is the throughput of IEEE 802.11e EDCA class 2, the IEEE 802.11 basic DCF protocol red line almost coincides with the top black line, the green line is the throughput of slotted ALOHA, and the blue line is the throughput of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol. The higher $p_a$ of class 1 compared to class 2 traffic is indicative of the higher QoS afforded through the use of different parameters for the two class.

The average number of attempts for a successful transmission is

$$\begin{aligned} n_a &= \sum_{i=0}^{\infty} i\ (1-p_a)^i\ p_a \\ &= \frac{1-p_a}{p_a} \end{aligned} \tag{11.31}$$

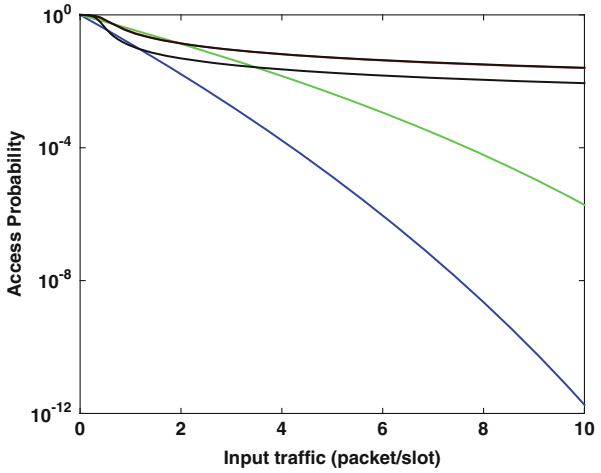



**Fig. 11.19** Access probability of IEEE 802.11.e EDCA (WMM) protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10$, $n = 20$, $m_1 = 5$, $m_2 = 10$, $p_1 = 0.5$, $p_2 = 0.05$, $\gamma_1 = 0.01$, and $\gamma_2 = 0.001$. The *top black line* is the throughput of IEEE 802.11.e EDCA class 1, the *bottom black line* is the throughput of IEEE 802.11e EDCA class 2, the IEEE 802.11 basic DCF protocol *red line* almost coincides with the *top black line*, the *green line* is the throughput of slotted ALOHA, and the *blue line* is the throughput of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol

Figure 11.20 shows the delay of the IEEE 802.11.e EDCA protocol when $n = 50$, $N = 10$, and $p = 0.5$ when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. The bottom black line is the delay of IEEE 802.11.e EDCA class 1, the top black line is the delay of IEEE 802.11e EDCA class 2, the IEEE 802.11 basic DCF protocol red line almost coincides with the top black line, the green line is the delay of slotted ALOHA, and the blue line is the delay of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol.

The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i+1)(1-p_a)^i \; p_a \\ &= \frac{E_0}{p_a} \end{aligned} \tag{11.32}$$

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$E/E_0 = -10 \log_{10} p_a \quad \text{dB} \tag{11.33}$$

Figure 11.21 shows the average energy required to transmit a packet for the IEEE 802.11.e EDCA protocol when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. The

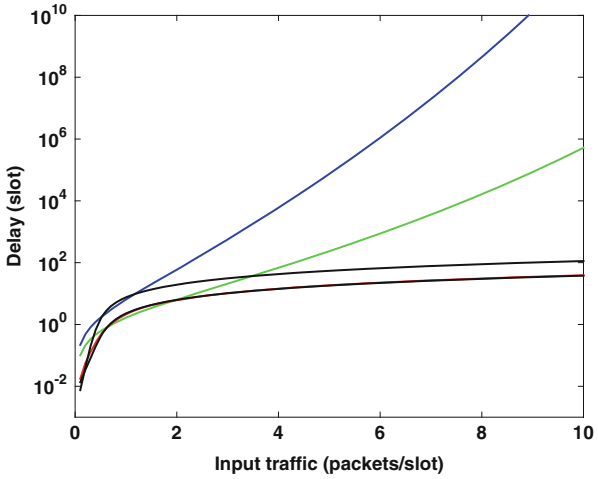



**Fig. 11.20** Frame delay of IEEE 802.11.e EDCA when $n = 20$, $N = 10$, $p = 0.5$, and $\gamma = 0.01$. The *bottom black line* is the delay of IEEE 802.11.e EDCA class 1, the *top black line* is the delay of IEEE 802.11e EDCA class 2, the IEEE 802.11 basic DCF protocol *red line* almost coincides with the *top black line*, the *green line* is the delay of slotted ALOHA, and the *blue line* is the delay of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol

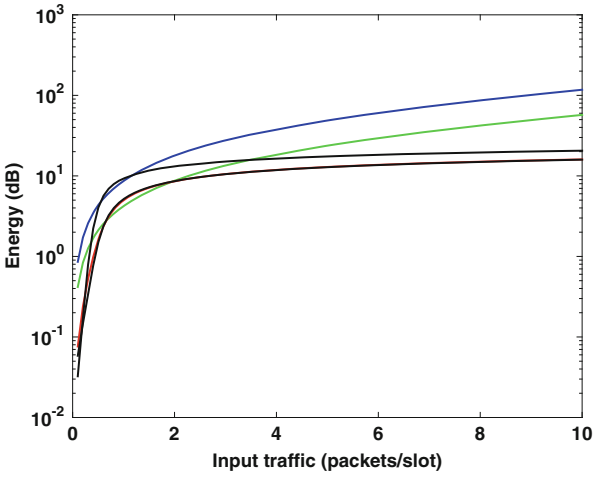



**Fig. 11.21** Average energy to transmit a frame for the IEEE 802.11 EDCA protocol when $n = 20$, $N = 10$, $p = 0.8$, and $\gamma = 0.01$. The *bottom black line* is the energy of IEEE 802.11.e EDCA class 1, the *top black line* is the energy of IEEE 802.11e EDCA class 2, the IEEE 802.11 basic DCF protocol *red line* almost coincides with the *top black line*, the *green line* is the energy of slotted ALOHA, and the *blue line* is the energy of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol

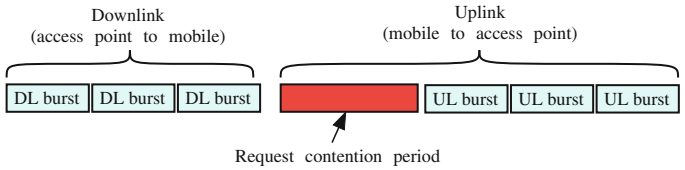



**Fig. 11.22** The IEEE 802.11e HCCA frame structure

bottom black line is the energy of IEEE 802.11.e EDCA class 1, the top black line is the energy of IEEE 802.11e EDCA class 2, the IEEE 802.11 basic DCF protocol red line almost coincides with the top black line, the green line is the energy of slotted ALOHA, and the blue line is the energy of pure ALOHA. The parameters for basic DCF protocol were the same as the parameters for the class 1 EDCA protocol.

## 11.5 IEEE 802.11e HCCA Protocol

The HCCA is an evolution of the IEEE 802.11 basic PCF protocol. The HCCA is the most advanced coordination function and requires the availability of an Access Point (AP) to coordinate the medium access. Another name for AP is Hybrid Controller (HC). HCCA is not mandatory in IEEE 802.11e.

Figure 11.22 shows a simplified view of how HCCA divides the time into frames. Each frame is composed of a downlink subframe followed by an uplink subframe. The downlink subframe is composed of several downlink data bursts sent by the access point. The burst is composed of downstream data and control messages informing each user or station if they are allowed to send data in the uplink subframe.

The HCCA standard does not specify how the upstream request contention period operates. This period is shown as the red block in Fig. 11.22. Several modes of sending a station request could be used:

1. Using an ALOHA-type protocol, users could transmit their requests and hope there are no users that simultaneously send their requests. The chance of collided requests increases with increasing number of users and with increased traffic. QoS support could be established using different persistence and backoff probabilities. We expect that at heavy traffic conditions the number of transmitted requests will drop off in a manner similar to the exponentially decreasing throughput of the ALOHA protocol, refer to Eq. (10.21) on page 344 or the S-ALOHA protocol, refer to Eq. (10.46) on page 350.
2. The access point could poll the users in a round robin fashion. Request collisions will not take place but still not all requests will be granted due to the limited uplink duration and TXOP limitations. QoS could be supported by

controlling the round robin frequency and allocation of bandwidth. However, maintaining fairness and QoS requires accurate control of the polling sequence. The performance of the individual users will be drastically affected by the polling sequence.

3. Any multiple access multiplexing technique could be used to reduce the chance of request collisions. Examples include: time division multiple access (TDMA), code division multiple access (CDMA), orthogonal frequency division multiple access (OFDMA), etc. This approach starts to look similar to the IEEE 802.16 (WiMax) protocol discussed in Chap. 12.

### *11.5.1 A Simple Markov Model for IEEE 802.11 HCCA*

In order to derive a simple model for the WiMax protocol we make several assumptions as follows:

1. All users can communicate with the access point using one hop. This assumption eliminates the hidden terminal problem.
2. The states of the Markov chain represent the states of the user: idle, waiting, transmitting, and collided.
3. There are only two classes of service with class 1 being the higher priority class.
4. The number of users in class 1 is $N_1$ and the number of users in class 2 is $N_2$.
5. The probability that a class 1 user has a packet to send in one frame time is $a_1$. $a_2$ is a similar probability for class 2 users.
6. The transmit opportunity (TXOP) is the same for both classes of service which corresponds to same packet size $n$.
7. The probability that a class 1 uncollided user transmits when the channel is free is $p_1$ and $p_2$ for class 2.
8. A collided class 1 user attempts a retransmission with probability $\gamma_1$ and $\gamma_2$ for class 2 user.
9. We use a shared multiple access request multiplexing scheme where there are $K < \min(N_1, N_2)$ contention slots. The duration of each contention slot is sufficient to send a request.
10. There are enough bandwidth to allow up to $K$ successful users to transmit in the next uplink subframe.

The restriction $K < \min(N_1, N_2)$ guarantees that there will be no idle contention slots when most of the users are active.

Each user can be in one of three states as shown in Fig. 11.23. The figure can be considered an extension of the IEEE 802.11basic DCF MAC protocol and bears some resemblance to the CSMA/CA protocol.

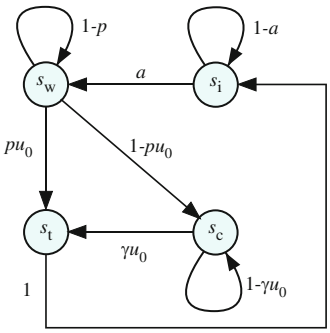



**Fig. 11.23** The IEEE 802.11e HCCA Markov chain model for a user

A tagged user's request will not collide with another user under two conditions: either the other user is not attempting a transmission or the other user has selected another contention slot. The probability of this event happening is given by:

$$x = s_i + (1-p)s_w + (1-\gamma)s_c + (ps_w + \gamma s_c)(1 - 1/K) \tag{11.34}$$

The value of $p$ and $\gamma$ will depend on the user's class of service which will lead to two probabilities $x_1$ and $x_2$ for users in class 1 or class 2, respectively.

The probability that the tagged user's request will not suffer a collision and will receive an acknowledgement is given by:

$$u_0 = x_1^{N_1} \times x_2^{N_2} \tag{11.35}$$

We organize the distribution vector at equilibrium as follows.

$$\mathbf{s} = \begin{bmatrix} s_i & s_w & s_t & s_c \end{bmatrix}^t \tag{11.36}$$

The corresponding transition matrix of the channel is given by:

$$\mathbf{P} = \begin{bmatrix} 1-a & 0 & 1 & 0 \\ a & 0 & 0 & 0 \\ 0 & pu_0 & 0 & \gamma u_0 \\ 0 & 1-pu_0 & 0 & 1-\gamma u_0 \end{bmatrix} \tag{11.37}$$

At equilibrium the distribution vector is obtained by solving the two equations

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{11.38}$$

$$\sum \mathbf{s} = 1 \tag{11.39}$$

However, the terms in $\mathbf{P}$ depend on the state vector components. This constitutes a highly nonlinear set of equations. The solution for $\mathbf{s}$ is obtained through several techniques such as optimization or iterative techniques as follows:

1. Input the values of $p$, $\gamma$, $a$, $N$, and $K$.
2. Assume a trial value for state vector $\mathbf{s}$.
3. Start the iterations by obtaining the probability $u_0$.
4. Substitute the value of $u_0$ to obtain an updated value $\mathbf{s}(updated)$.
5. Calculate the error $\mathbf{e} = \mathbf{s}(updated) - \mathbf{s}$ and the root mean square error $e_{rms}$.
6. Update the state vector

$$\mathbf{s} = \mathbf{s} + \alpha\, \mathbf{e}$$

where $\alpha$ is the update step size which is usually taken equal to 0.1 or even smaller.
7. Repeat the iterations starting at Step 3 and stop when $e_{rms}$ is below a certain value.

### 11.5.2 IEEE 802.11.e HCCA Protocol Performance

The throughput for each class of service is given by the equation:

$$Th = s_t\, N \tag{11.40}$$

Figure 11.24 shows the throughput of IEEE 802.11.e HCCA protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The top black line is the throughput of IEEE 802.11.e HCCA class 1, the bottom black line is the throughput of IEEE 802.11e HCCA class 2, the blue line is the throughput of slotted ALOHA, and the red line is the throughput of pure ALOHA.

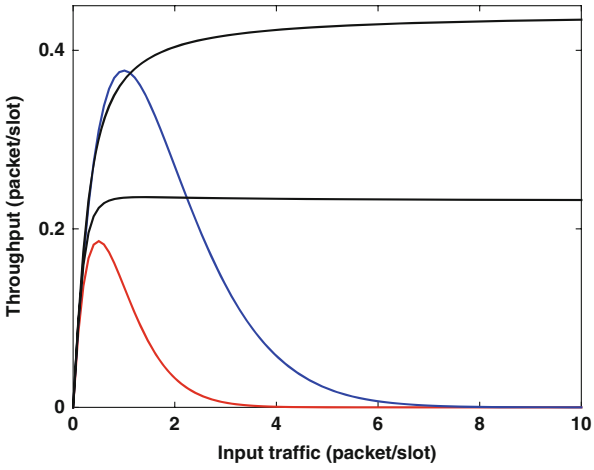



**Fig. 11.24** Throughput of IEEE 802.11.e HCCA protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The *top black line* is the throughput of IEEE 802.11.e HCCA class 1, the *bottom black line* is the throughput of IEEE 802.11e HCCA class 2, the *blue line* is the throughput of slotted ALOHA, and the *red line* is the throughput of pure ALOHA

We note that the class 1 throughput is close to the maximum value of the slotted ALOHA protocol. Reducing the value of the persistence probability $p$ results in reduced throughput. The highest throughput was achieved when $p = 1$. However, service differentiation and optimization of the throughput is accomplished by the proper choice of $\gamma$. Small or large values of $\gamma$ reduce the throughput.

The class 2 EDCA protocol has almost one-half the throughput of the class 1 protocol due to the chosen parameter values. That is why class 2 throughput is close to the peak of the pure ALOHA protocol.

The user access probability is given by:

$$p_a = \frac{Th}{Na} \tag{11.41}$$

Figure 11.25 shows the access probability of IEEE 802.11.e HCCA protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The top black line is the access probability of IEEE 802.11.e HCCA class 1, the bottom black line is the access probability of IEEE 802.11e HCCA class 2, the blue line is the access probability of slotted ALOHA, and the red line is the access probability of pure ALOHA. The higher $p_a$ of class 1 compared to class 2 traffic is indicative of the higher QoS afforded through the use of different parameters for the two classes.

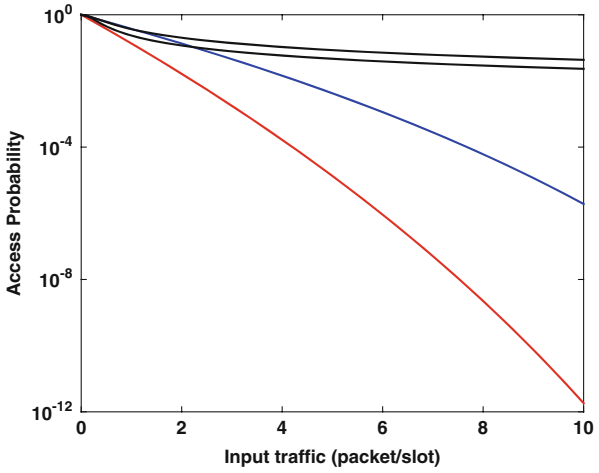



**Fig. 11.25** Access probability of IEEE 802.11.e HCCA protocol versus the average input traffic for two classes of service when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The *top black line* is the access probability of IEEE 802.11.e HCCA class 1, the *bottom black line* is the access probability of IEEE 802.11e HCCA class 2, the *blue line* is the access probability of slotted ALOHA, and the *red line* is the access probability of pure ALOHA

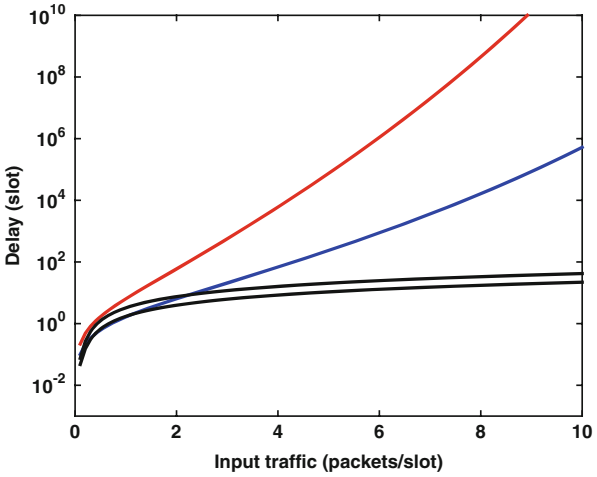



**Fig. 11.26** Frame delay of IEEE 802.11.e HCCA when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The *bottom black line* is the delay of IEEE 802.11.e HCCA class 1, the *top black line* is the delay of IEEE 802.11e HCCA class 2, the *blue line* is the delay of slotted ALOHA, and the *red line* is the delay of pure ALOHA

The average number of attempts for a successful transmission is

$$\begin{aligned} n_a &= \sum_{i=0}^{\infty} i \ (1 - p_a)^i \ p_a \\ &= \frac{1 - p_a}{p_a} \end{aligned} \tag{11.42}$$

Figure 11.26 shows the delay of the IEEE 802.11.e HCCA protocol when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The bottom black line is the delay of IEEE 802.11.e HCCA class 1, the top black line is the delay of IEEE 802.11e HCCA class 2, the blue line is the delay of slotted ALOHA, and the red line is the delay of pure ALOHA.

The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i + 1)(1 - p_a)^i \ p_a \\ &= \frac{E_0}{p_a} \end{aligned} \tag{11.43}$$

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$E/E_0 = -10 \log_{10} p_a \quad \text{dB} \tag{11.44}$$

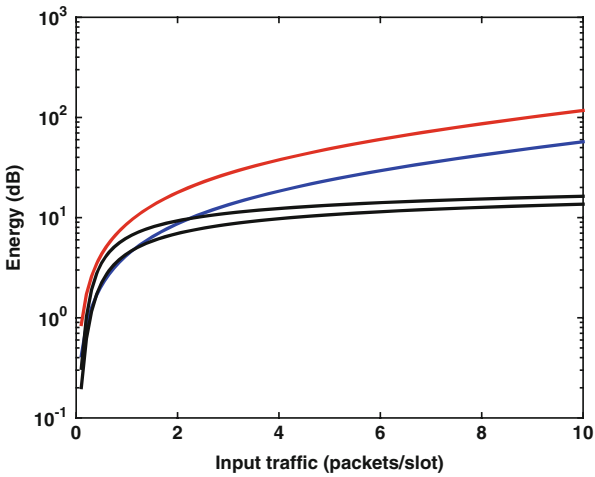



**Fig. 11.27** Average energy to transmit a frame for the IEEE 802.11e HCCA protocol when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The *bottom black line* is the delay of IEEE 802.11.e HCCA class 1, the *top black line* is the delay of IEEE 802.11e HCCA class 2, the *blue line* is the delay of slotted ALOHA, and the *red line* is the delay of pure ALOHA

Figure 11.27 shows the average energy required to transmit a packet for the IEEE 802.11e HCCA protocol when $n = 50$, $N = 10$, and $p = 0.5$. Figure 11.21 is the case when $N_1 = N_2 = 10$, $p_1 = 1$, $p_2 = 1$, $\gamma_1 = 0.1$, and $\gamma_2 = 0.05$. The bottom black line is the delay of IEEE 802.11.e HCCA class 1, the top black line is the delay of IEEE 802.11e HCCA class 2, the blue line is the delay of slotted ALOHA, and the red line is the delay of pure ALOHA.

## 11.6 IEEE 802.11 Final Remarks

The DCF mode of the IEEE 802.11 protocols has been studied by many researchers. We tried to present here a simple model to start the reader in the area of modeling protocols. However, there are many ripe areas that have not been adequately explored for this protocol and for others also. We enumerate some of these directions:

1. Channel errors have not been considered. This is a physical layer problem but could also be considered in a cross-layer modeling. What matters here is to obtain the probability that a frame or packet is in error.
2. Channel fading has not been considered. This is called cross-layer modeling. It becomes useful when adaptive modulation and decoding are used. Chapter 13 attempts to provide a simple discussion of modeling channel fading.
3. Simple backoff strategies were used here in order to obtain simple expressions. Adopting binary exponential backoff (BEB) strategy would result in a more realistic model.
4. Perhaps the most serious deficiency in the models in this chapter is the implicit assumption that no new traffic arrives while the user is attempting to access the channel. This amounts to assuming the *transmit* buffer has a single storage location only. Using a more realistic buffer would require solving two queuing systems.

5. It was implicitly assumed that the transmit antennas were omnidirectional. The author's group, as well as others, have dealt with the interesting case of using directional antennas and its concomitant *deafness* problem.
6. It was implicitly assumed that all $N$ users in our system can talk to each other through one hop. Using a multi-hop network would improve the performance even with the *hidden terminal* problem.

## 11.7 Problems

### *IEEE 802.11 Basic DCF*

**11.1.** Explain what is meant by the following PHY layer terms:

1. Frequency-hopping spread spectrum (FHSS),
2. Direct-sequence spread-spectrum (DSSS) link layer,
3. Orthogonal Frequency Division Multiple Access (OFDM).

**11.2.** Explain what is meant by infrastructure wireless networks and relate that to the concepts of Ad-Hoc networks and base stations.

**11.3.** Explain what is meant by ad hoc wireless networks.

**11.4.** Explain what is meant by wireless sensor networks. How these differ from ad hoc networks.

**11.5.** Explain the operation of the DCF and indicate the type of LAN that uses it (infrastructure or ad hoc)?

**11.6.** Explain how DCF reduces the probability of collisions.

**11.7.** Is the basic DCF function close to CSMA/CA or CSMA/CD? Explain your answer.

**11.8.** Simulate the performance of the IEEE 802.11 basic DCF protocol for different values of the persistence probability $p$.

**11.9.** The analysis of the IEEE 802.11 basic DCF protocol assumed equally likely assignment of users to the $w$ reservation slots. Develop a model of the channel when the assignment of active users to reservation slots follows a distribution different from the uniform distribution.

**11.10.** The analysis of the IEEE 802.11 basic DCF protocol assumed that the backoff counters of active users decrement by one when the channel is free. Develop a model of channel when the backoff counters assume a new random value each time the channel is busy. Only users that can transmit a frame are the ones that happen to have a backoff counter value of 0.

**11.11.** In an attempt to improve the performance of the IEEE 802.11 basic DCF protocol let us assume that an idle user will *immediately* attempt a transmission when the channel is sensed idle. Analyze this situation.

**11.12.** Model the IEEE 802.11 basic DCF protocol when truncated binary backoff strategy is employed. In other words when the collided user attempts $m$ retransmission attempts before declaring the channel unavailable.

## *IEEE 802.11 RTS/CTS*

**11.13.** Is the RTS/CTS mechanism of the IEEE 802.11 close to CSMA/CA or CSMA/CD? Explain your answer.

**11.14.** Simulate the performance of the IEEE 802.11 RTS/CTS protocol for different values of the persistence probability $p$.

**11.15.** Simulate the performance of the IEEE 802.11 RTS/CTS protocol for different values of the backoff probability $\gamma$.

**11.16.** The actual implementation of the IEEE 802.11 RTS/CTS protocol assumed that backoff is accomplished using a counter that decrements by one when the channel is free. Develop a model of system.

**11.17.** In an attempt to improve the performance of the RTS/CTS protocol let us assume that an idle user will *immediately* attempt an RTS transmission when the channel is sensed idle. Analyze this situation.

**11.18.** Model the IEEE 802.11 RTS/CTS protocol when truncated binary backoff strategy is employed. In other words when the collided user attempts $m$ retransmission attempts before declaring the channel unavailable.

## *IEEE 802.11 EDCA*

**11.19.** Model the EDCA protocol when the TXOP for the two classes of service is not equal. The higher value of XTOP is assigned to the higher priority class.

**11.20.** How will the EDCA model in Sect. 11.4 change if four classes of service are modeled? How will you ascertain that there is service differentiation for the developed model?

## *IEEE 802.11e HCCA*

**11.21.** Explain the operation of the PCF and indicate the type of LAN that uses it (infrastructure or ad hoc).

**11.22.** Explain if IEEE 802.11e HCCA eliminates collisions.

**11.23.** Investigate the types of scheduling protocols that could be used in the HCCA portion of the IEEE 802.11e protocol.

**11.24.** Develop a discrete-time Markov chain model for the IEEE 802.11e user when the access point uses polling.

**11.25.** Assume the IEEE 802.11e HCCA protocol that serves 10 customers and the channel speed is 1 Mbps. Each customer is assumed to issue requests at a rate of 100 requests/s and the average length of a frame is 5.12 kb. Obtain the performance of this system.

**11.26.** Analyze the IEEE 802.11e HCCA protocol in which there are two customer classes. Class 1 has $N_1$ customers and class 2 has $N_2$ customers. Users in class 1 can access the channel when they issue a request, while users in class 2 can only access the channel when none of the users of class 1 has a request.

**11.27.** In the analysis of the IEEE 802.11e HCCA protocol, where users had transmit buffers, it was assumed that requests arriving at a given time step are processed in that time step. Develop a new analysis that processes these requests at the next time step.

# References

1. J. Weinmiller, H. Woesner, A. Wolisz, Analyzing and improving the IEEE 802.11-MAC protocol for wireless LANs, in *Proceedings of the Fourth International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'96)*, pp. 200–206 (1996)
2. S. Khurana, A. Kahol, S.K.S. Gupta, P.K. Srimani, Performance evaluation of distributed coordination function for IEEE802.11 wireless LAN protocol in presence of mobile and hidden terminals, in *Proceedings of 7th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 40–47 (1999)
3. J. Liu, D.M. Nicol, L.F. Perrone, M. Liljenstam, Towards high performance modeling of the 802.11 wireless protocol, in *Proceedings of the Winter Simulation Conference*, vol. 9, pp. 1315–1320 (2001)

# Chapter 12
# Modeling IEEE 802.16 (WiMAX) Protocol

## 12.1 Introduction

The IEEE 802.16 (WiMAX) provides a wireless high speed access (2–11 GHz) to users through a centralized base station that can cover several thousand square kilometers. In that sense, WiMAX is a metropolitan area network (MAN). The base stations are connected together and to the rest of the Internet using high speed wired links or microwave links. The IEEE 802.16e-2005 allows mobile users.

Table 12.1 provides a comparison between WiFi and WiMAX.

## 12.2 IEEE 802.16 Frame Structure

Figure 12.1 shows a simplified view of how WiMAX divides the time into frames. Each frame is composed of a downlink subframe followed by an uplink subframe. The downlink subframe is composed of several downlink data bursts sent by the base station. The burst is composed downstream data and control messages informing each user if they are allowed to send data in the uplink subframe and which frequency band to use.

The upstream subframe starts with several contention slots. WiMAX uses a scheduling algorithm such that each user is assigned an access slot in the uplink phase to use exclusively by that user. In that sense each user will not suffer any collisions from other user requests.

After the contention slots, the uplink subframe contains uplink data bursts from the users to the base station.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_12

**Table 12.1** Comparing WiFi and WiMAX

| IEEE 802.11 WiFi | IEEE 802.16 WiMAX |
|---|---|
| Ad Hoc networking where medium access is distributed among the users | Infrastructure networking where access control is done through a base station |
| Local area network access (LAN) | Metropolitan area network (MAN) |
| Support tens of users | Support hundreds of users |
| Fixed 20 MHz channel size | Flexible channel size from 1.5 to 20 MHz |
| User with data to send performs CSMA/CA. When channel is free, it waits for random delay before attempting to transmit | User with data to send issues a request in the uplink phase and waits for a grant from the base station |

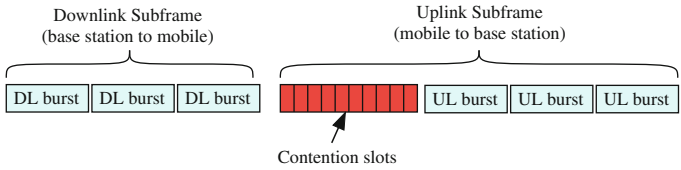



**Fig. 12.1** The IEEE 802.16 frame structure

## 12.3 A Simple Model for the IEEE 802.16 WiMAX

This section deals with modeling the IEEE 802.16 protocol. We choose to model the performance of the system through modeling the access point (AP) since this is where all the user requests are processed and scheduled. We make the following assumptions for our analysis of the WiMAX system

1. The states of the Markov chain represent the number of queued users requesting access at the start of any time step.
2. The Markov chain time step is taken equal to the frame duration.
3. The system has a fixed population of $N$ users.
4. There is a single customer class.
5. A user can have, at most, one message waiting for transmission. At the end of certain time step users that have requests pending cannot issue more requests at the next time step.
6. There are $K$ channels and they are assumed to be always available for transmission in every time step.
7. The AP will store the requests that were denied access and will include them in the pool of requests at the next frame cycle.
8. Probability that a user requests transmission is $a$ and probability that the user is idle is $b = 1 - a$.
9. The scheduler will randomly select up to $K$ requests from the queued requests and assign those to the available $K$ channels.

The states of the system are chosen to represent the number of queued requests at a given time step. Therefore we have $N - K + 1$ states in our state vector:

$$\mathbf{s} = \begin{bmatrix} s_0 \; s_1 \cdots s_{N-K} \end{bmatrix}^t \tag{12.1}$$

where $s_i$ indicates there are $i$ queued requests. We have only $N-K+1$ components because the AP will never have more than $N-K$ queued requests at any given time. The worst case happens when $N$ requests are received on the contention slots from all $N$ users. In the downlink phase the AP will award $K$ users access to the channel and only $N - K$ unsatisfied requests will remain.

Starting at state $j$, the probability of making a transition to state $i$ is governed by the following observations.

1. When the AP is in state $s_j$, the number of new received requests will vary between 0 and $N - i$.
2. When the AP is in state $s_j$ with $0 \leq j \leq K$, the next state will be $s_i$ where $0 \leq i \leq N - j$.
3. When the AP is in state $s_j$ with $K < j \leq N - j$, the next state will be $s_i$ where $j - K \leq i \leq N - K$.

Based on the above observations, the corresponding transition matrix of the AP is $N - K + 1 \times N - K + 1$. The state transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix} y_0 & y_1 & \cdots & 0 \\ x(N, K+1) & x(N-1, 1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x(N, N) & x(N-1, N-1) & \cdots & x(N-K, N-K) \end{bmatrix} \tag{12.2}$$

where $x(i, j)$ is the probability that there were $j$ queued requests from the previous frame and $j$ new requests were received from $j$ of the $N - j$ users that could potentially issue a request at the current frame. The probability $x(i, j)$ is given by:

$$x(i, j) = \binom{N-j}{i} a^i b^{N-j-i} \tag{12.3}$$

The entries $y_j$ are given by:

$$y_j = \sum_{k=0}^{K-j} x(N-j, k) \tag{12.4}$$

For the case $N = 8$ and $K = 3$, the transition matrix is $6 \times 6$ and is given by:

$$\mathbf{P} = \begin{bmatrix} y_0 & y_1 & y_2 & x(5,0) & 0 & 0 \\ x(8,4) & x(7,3) & x(6,2) & x(5,1) & x(4,0) & 0 \\ x(8,5) & x(7,4) & x(6,3) & x(5,2) & x(4,1) & x(3,0) \\ x(8,6) & x(7,5) & x(6,4) & x(5,3) & x(4,2) & x(3,1) \\ x(8,7) & x(7,6) & x(6,5) & x(5,4) & x(4,3) & x(3,2) \\ x(8,8) & x(7,7) & x(6,6) & x(5,5) & x(4,4) & x(3,3) \end{bmatrix} \tag{12.5}$$

where, for example, $y_1$ is given by:

$$y_1 = x(7,0) + x(7,1) + x(7,2) \tag{12.6}$$

Notice that for this case the system had one queued request from the previous frame and could accommodate up to two new requests before it clears all requests and go back to state $s_0$ at the next frame.

### 12.3.1 IEEE 802.16 WiMAX System Performance

Having obtained the transition matrix, we are able to find the performance of the IEEE 802.16 system.

The throughput of the system is given by:

$$Th = \sum_{k=1}^{K-1} k \sum_{j=0}^{k} s_j x(N-j, k-j) + K \sum_{j=0}^{K-1} s_j \sum_{i=K-j}^{N-j} x(N-j, i) + K \sum_{j=K}^{N-K} s_j \tag{12.7}$$

The first term on the right-hand side corresponds to the case when the number of queued requests from previous frame is in the range $[0, K)$ such that the total number of requests is less than $K$.

The second term on the right-hand side corresponds to the case when the number of queued requests from the previous frame is in the range $[1, K)$ such that the total number of requests is equal to or greater than $K$.

The third term on the right-hand side corresponds to the case when the number of queued requests from previous frame is in the range $[K, N-K]$.

Figure 12.2 shows the throughput of the IEEE 802.16 WiMAX protocol when $N = 50$ and $K = 10$. Notice that the throughput reaches the value $K$ as soon as the input traffic approaches $K$ since at that level there is a good probability that $K$ users have frames to send. Further, the storage of queued requests at the AP ensures that arriving requests are not dropped.

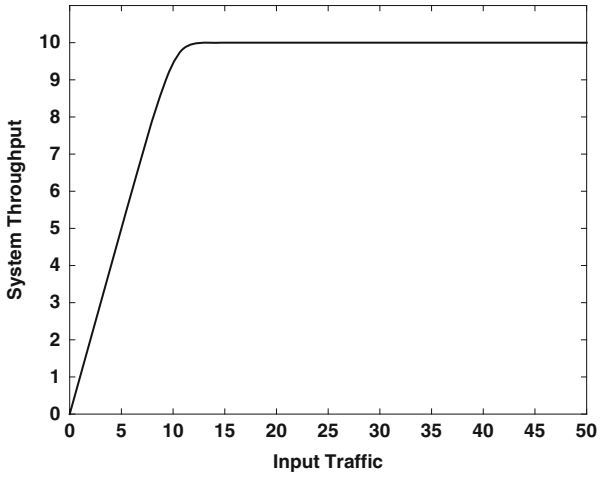



**Fig. 12.2** The throughput for IEEE 802.16 WiMAX versus the average input traffic when $N = 50$ and $K = 10$

The channel utilization is defined as:

$$u = \frac{Th}{K} \tag{12.8}$$

where $0 \leq u \leq 1$ indicates the percentage of using a channel from the $K$ available channels. Figure 12.3 shows the channel utilization of the IEEE 802.16 WiMAX protocol when $N = 50$ and $K = 10$. Notice that the utilization reaches the value 100 % as soon as the input traffic approaches $K$ since at that level there is a good probability that $K$ users have frames to send. Further, the storage of queued requests at the AP ensures that arriving requests are not dropped.

We define $p_a$ as the access probability for a user's request which is given by:

$$p_a = \frac{Th(\text{user})}{a} = \frac{Th}{N_a(in)} \tag{12.9}$$

Figure 12.4 shows the access probability versus input traffic when $N = 50$ and $K = 10$.

Having found the acceptance probability, we are able to determine the average number of frames before a user gets access to the medium.

$$n_a = \sum_{k=0}^{\infty} k\,(1 - p_a)^k \;\; p_a \tag{12.10}$$

$$= \frac{1 - p_a}{p_a} \tag{12.11}$$

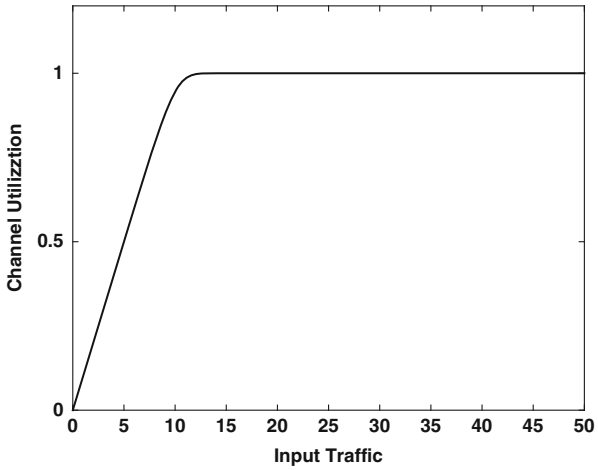



**Fig. 12.3** Channel utilization for IEEE 802.16 WiMAX versus the average input traffic when $N = 50$ and $K = 10$

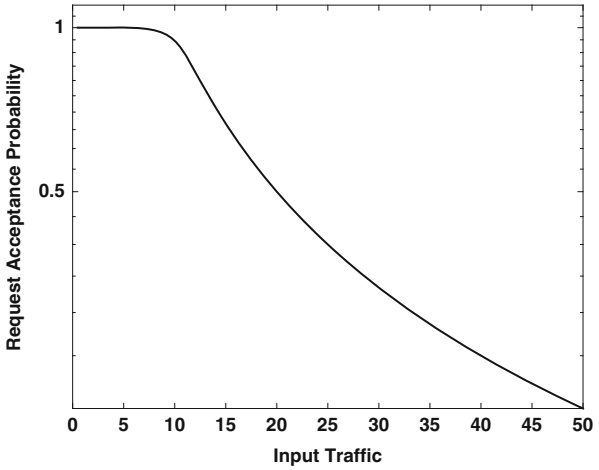



**Fig. 12.4** The access probability in IEEE 802.11/PCF versus the average input traffic when $N = 50$ and $K = 10$

Figure 12.5 shows the delay of IEEE 802.16 WiMAX when $N = 50$ and $K = 10$.

The average number $Q$ of queued users with waiting requests is found as:

$$Q = \sum_{j=0}^{K} j s_j \tag{12.12}$$

Figure 12.6 shows the average number of queued requests of IEEE 802.16 WiMAX when $N = 50$ and $K = 10$. We notice that at the highest input traffic of 50 requests/frame, the maximum number of queued users becomes 40, as expected since $K = 10$ and at these conditions, ten requests are accommodated and 40 remain queued.

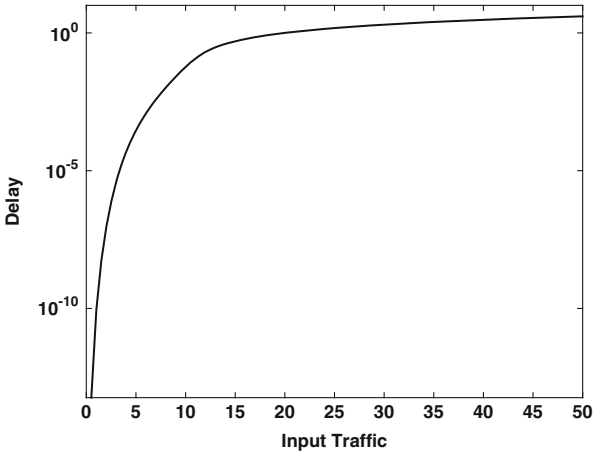



**Fig. 12.5** Delay for the IEEE 802.16 WiMAX protocol versus the average input traffic when $N = 50$ and $K = 10$

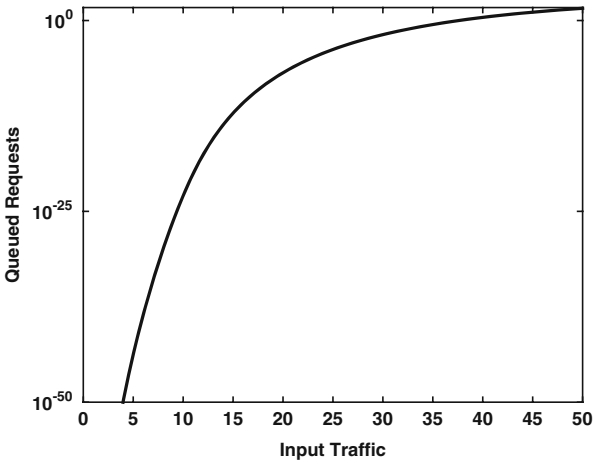



**Fig. 12.6** Average number of queued requests for the IEEE 802.16 WiMAX protocol versus the average input traffic when $N = 50$ and $K = 10$

### *12.3.2 IEEE 802.16 WiMAX User Performance*

The previous section modeled the states all the $N$ users of the IEEE 802.16 WiMAX. In this section we study an individual user, usually called the tagged user.

Because all users are identical, the throughput seen by the tagged user is given by:

$$Th(\text{user}) = \frac{Th}{N} \tag{12.13}$$

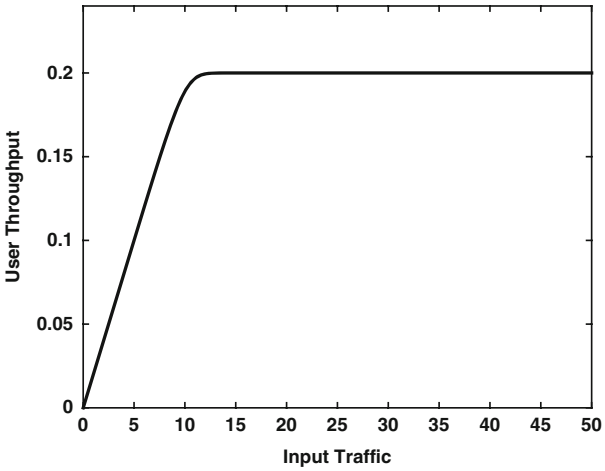



**Fig. 12.7** User throughput for the IEEE 802.16 WiMAX protocol versus the average input traffic when $N = 50$ and $K = 10$

Figure 12.7 shows the user throughput of IEEE 802.16 WiMAX when $N = 50$ and $K = 10$. As expected, the maximum user throughput is given by:

$$Th(\text{user})_{max} = \frac{K}{N} \tag{12.14}$$

For the case in the figure, the maximum throughput is $10/50 = 0.2$.

*Example 12.1.* An IEEE 802.16 WiMAX has $K = 10$ channels and serves 60 customers. The channel bit rate is 1,920 kbps and each customer issues 200 requests per second. Assuming the average frame length is 5.12 kb, obtain the performance of this system.

First we must find the time step value $T$ knowing the duration of an average frame.

$$T = 5.12/1920 = 2.7 \qquad \text{ms}$$

To find the frame arrival probability $a$ per time step we need to estimate the number of requests issued by the user over some observation time and we need also to find the number of time steps during this same observation period. Take the observation period to be 1 s. Thus the total user traffic during this period is:

$$N_a(in) = \text{user request rate} \times 1 = 200 \text{ requests}$$

The number of frames during this period $n$ is given by:

$$n = \lceil \frac{1\text{s}}{T} \rceil = 375 \quad \text{frames}$$

The number of requests from the binomial distribution is given from the relation

$$a \times n = N_a(in)$$

Thus we have the user request probability as

$$a = 0.5333 \quad \text{frame/time step}$$

Now that we know $a$, $N$ and $K$, the transition matrix $\mathbf{P}$ has dimensions $51 \times 51$ and the bottom right six entries are given by:

$$\mathbf{P} = \begin{bmatrix} \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & 0.1238 & 0.1547 & 0.1865 & 0.2152 & 0.2353 & 0.2407 \\ \cdots & 0.0643 & 0.0884 & 0.1184 & 0.1537 & 0.1921 & 0.2292 \\ \cdots & 0.0245 & 0.0367 & 0.0541 & 0.0781 & 0.1098 & 0.1497 \\ \cdots & 0.0065 & 0.0105 & 0.0169 & 0.0268 & 0.0418 & 0.0642 \\ \cdots & 0.0011 & 0.0018 & 0.0032 & 0.0056 & 0.0096 & 0.0163 \\ \cdots & 0.0001 & 0.0002 & 0.0003 & 0.0005 & 0.0010 & 0.0019 \end{bmatrix}$$

The associated distribution vector has 51 components and the last ten entries are given by:

$$\mathbf{s} = \begin{bmatrix} \cdots & 0.0525 & 0.0235 & 0.0080 & 0.0019 & 0.0003 & 0.0000 \end{bmatrix}^t$$

The performance figures are

$$\begin{array}{lcll} Th & = & 10 & \text{frames/time step} \\ Q & = & 0.0011 & \text{requests} \\ p_a & = & 0.3125 & \\ n_a & = & 2.2 & \text{time steps} \end{array}$$

■

## 12.4 Problems

**12.1.** Assuming that a user's request in a contention slot was correctly received by the base station but still, the user did not receive a grant during the next downlink subframe. Can you figure out why this might happen?

**12.2.** Use the simple model provided in this chapter to explore the effect of the number of users $N$ on the system performance.

**12.3.** Use the simple model provided in this chapter to explore the effect of the number of data channels $K$ on the system performance.

**12.4.** Modify the simple model provided in this chapter to explore the effect of having the AP use a binary exponential back off strategy for the ungranted requests.

**12.5.** The simple model provided in this chapter assumed that the base station stores a received request if it did not issue a grant to the user. How can you modify this model to develop an access protocol where the base station does not store received requests that were not issued a grant.

# Chapter 13
# Modeling of Wireless Fading Channels

## 13.1 Introduction

A wireless signal sent from a transmitter to a receiver undergoes several energy loss mechanisms:

1. Path loss (large-scale phenomenon)
2. Shadowing (large-scale phenomenon)
3. Multipath fading (small-scale phenomenon)

## 13.2 Path Loss

*Path loss*, or free-space propagation, occurs due to the spread of the signal beam over distance and absorption through the channel. It is a large-scale phenomenon since variations in the signal occur over distances large compared to the signal wavelength. Figure 13.1 shows path loss due to two types of antennas. Figure 13.1a shows two omnidirectional antennas being used by the transmitter and receiver. The combined antenna gain $G = 1$. Figure 13.1b shows two directional antennas where the combined antenna gain is $G > 1$. The received signal power $P_r$ is related to the transmitted signal power $P_t$ through the free-space fading formula [1]:

$$P_r = P_t G \times \left(\frac{\lambda}{4\pi d}\right)^2 \tag{13.1}$$

where $G$ is the product of the transmitter and receiver antenna gains due to directivity and efficiency, $\lambda$ is the signal wavelength, and $d$ is the distance between the transmitting and receiving antennas.

*Example 13.1.* Assume the signal carrier frequency $f_c$ is 900 MHz, the transmitted power is 1 mW, the transmit antenna has a gain of 1 and the receive antenna has



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_13

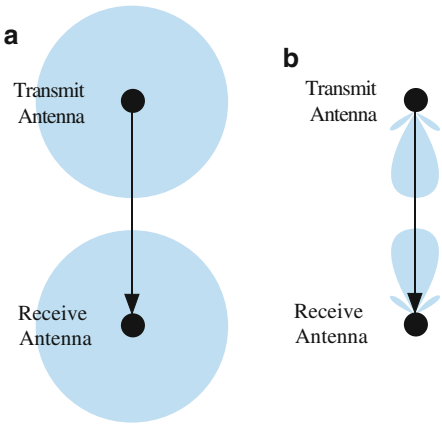



**Fig. 13.1** Transmitted signal energy loss due to free-space propagation. (**a**) Transmit and receive antennas are omnidirectional. (**b**) Transmit and receive antennas have directionality

a gain of 1.5. Find the power level when transmitter and receiver are 50 m apart assuming free-space propagation.

The wavelength of the signal is:

$$\lambda = \frac{c}{f_c} = 0.3333 \text{ m}$$

The antennal gain is $G = 1.5$. Applying (13.1) the received power is given by:

$$P_r = 0.7958 \ \mu\text{W}$$ ■

Actual path loss seldom follows the simple relation in (13.1). A more general formula is given by [1]:

$$P_r = P_t K \times \left(\frac{d_0}{d}\right)^{\gamma} \tag{13.2}$$

where $K$ is a dimensionless constant determined by the antennas used, $d_0$ is some reference distance and $\gamma$ is the *path loss exponent*.

Sometimes the value of $K$ is chosen equal to the free-space formula by combining (13.1) and rewriting (13.2) as follows:

$$P_r = P_t \left(\frac{d_0 \lambda}{4\pi d_0 d}\right)^{\gamma} \tag{13.3}$$

$$\approx P_t \left(\frac{\lambda}{4\pi d_0}\right)^2 \left(\frac{d_0}{d}\right)^{\gamma} \tag{13.4}$$

$$K = \left(\frac{\lambda}{4\pi d_0}\right)^2 \tag{13.5}$$

Note that $K$ uses the free-space exponent of 2 instead of $\gamma$.

## 13.3 Shadowing

*Shadowing* is also a large-scale propagation effect due to presence of objects between the transmitter and receiver. The intervening object could reflect of scatter some of the transmitted signal and it could also attenuate its magnitude due to its dielectric properties. Figure 13.2 shows the reflected and scattered beams due to the presence of an object in the path of the transmitted signal between the transmit and receive antennas. Shadowing effect is modeled using *log-normal shadowing*. The pdf distribution of the received power is given by [1]:

$$p(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left[-\frac{(\ln x - \mu)^2}{2\sigma^2}\right] \tag{13.6}$$

where $x = P_t/P_r > 0$, $\mu$ is the mean of $x$, and $\sigma$ is the standard deviation of $x$.

## 13.4 Multipath Small-Scale Fading

*Multipath fading* is a small-scale propagation effect since it occurs over short distances comparable to the signal wavelength. Fading refers to the rapid fluctuations in the received signal amplitude and power due to reflections, refraction, and scattering off or through intervening objects. This phenomenon occurs when the transmitted signal travels through different paths before arriving at the receive antenna. The different paths are due to signal *reflection* and *scattering* off objects and signal *diffraction* when passing through objects with different dielectric constants. Because of constructive and destructive interference, the received signal fluctuates in amplitude.

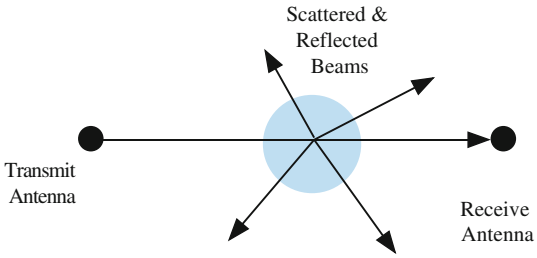



**Fig. 13.2** Transmitted signal energy loss due to reflection and scattering of the beam due to an object

Propagation of the signal over multipaths creates different types of random and rapid distortions:

1. Changes in received signal amplitude and power due to coherent interference between the different received signal copies.
2. Frequency modulation due to Doppler shifts due to the motion of the transmitter, receiver or surrounding objects.
3. Dispersion or spreading of the signal due to the different time delays of the paths.

Multipath signals experience different delays before arriving at the receiver. The *delay spread* $T_d$ (s) of the channel is the time difference between arrival of the earliest component (line of sight path) and the latest time of arrival of the multipath components. Delay spread affects inter symbol interference (ISI).

The *symbol duration* $T_s$ (s) is the time required to transmit one symbol. When $T_s \ll T_d$, we can expect ISI-free transmission.

In the frequency domain, *coherence bandwidth* $B_c = 1/T_d$ (Hz) corresponds to delay spread in the time domain. Smaller values of $T_d$ result in large values of coherence bandwidth $B_c$.

Frequency selectivity is caused by self-interference of the transmitted signal caused by multipath arrival of the signal components. We can intuitively figure out that when $T_d$ is small, and $B_c$ is large, the phase differences between the arriving signals will almost be the same and constructive interference results. Such a channel will show the same constructive interference results for a signal with a small value of $B_s$ since the signals have almost the same wavelength and have close phase variations.

The opposite situation happens when $T_d$ is large (i.e., small $B_c$). The phase differences of the arriving signals could approach 180° and destructive interference results.

We can classify the frequency selectivity of the channel based on the relative magnitudes of signal bandwidth $B_s$ (Hz) and coherence bandwidth $B_c$:

$$B_s < B_c \rightarrow \text{flat in frequency (narrow fading) channel} \tag{13.7}$$

$$B_s \geq B_c \rightarrow \text{frequency selective channel} \tag{13.8}$$

Most wireless communication systems are modeled as flat in frequency/narrow fading channels.

A mobile device experiences Doppler shift in the frequency of the received signals due to relative motion of the receiver or transmitter. The difference between the maximum Doppler shift and minimum received signal frequencies is defined as the *Doppler spread* $f_c$ (Hz). The channel *coherence time* is defined as $T_c = 1/f_c$ (s). The coherence time is thought of as the time period over which the arriving signals have almost the same phase.

We can classify the time selectivity of the channel based on the signal symbol duration $T_s$:

$$T_s < T_c \rightarrow \text{flat in time channel} \tag{13.9}$$

$$T_s \geq T_c \rightarrow \text{time selective channel} \tag{13.10}$$

Most wireless systems are also modeled as flat in time channels.

## 13.5 Statistical Modeling of Wireless Narrow Fading Channels

The received signal amplitude power varies randomly with time. The Rayleigh fading model provides a simple expression for the pdf of the received signal amplitude $x$ [1–3]. The pdf of the Rayleigh distribution is given in Sect. 1.21 by:

$$f_X(x) = \frac{x}{a^2} e^{-x^2/(2a^2)}, \qquad x \geq 0, \quad a > 0 \tag{13.11}$$

where $a$ is the shape parameter.

The random variable representing the signal amplitude was denoted $x$. We denote the random variable for the received signal power by $y = x^2$. Therefore, the average signal power is obtained by:

$$\begin{aligned} E(y) = P &= \int_{x=0}^{\infty} x^2 \times f_X(x) \\ &= \int_{x=0}^{\infty} x^2 \times \frac{x}{a^2} e^{-x^2/2a^2} \, \mathrm{dx} \end{aligned} \tag{13.12}$$

We change the variables as $y = x^2$, and we have $\mathrm{d}y = 2x \, \mathrm{d}x$. Substituting this in the above equation gives us:

$$\begin{aligned} E(y) &= \int_{y=0}^{\infty} y \, \frac{1}{a^2} e^{-y/2a^2} \, \mathrm{d}y \\ &= \int_{y=0}^{\infty} y \, f_Y(y) \, \mathrm{d}y \end{aligned} \tag{13.13}$$

where $f_Y(y)$ is the pdf for the received signal power and is given by:

$$f_Y(y) = \frac{1}{a^2} e^{-y/2a^2} \tag{13.14}$$

*Example 13.2.* Assume the average received signal power is $P$ and given the pdf for the received signal power in (13.14):

1. Determine the shape parameter value $a$ as a function of $P$.
2. Determine the corresponding pdf's for the received signal amplitude and power in terms of $P$.
3. Determine the corresponding CDF's for the received signal amplitude and power in terms of $P$.
4. Find the average values of the received signal amplitude and received signal power in terms of $P$.

**1.** The mean of the power distribution $P$ in (13.14) is given by:

$$\begin{aligned} P &= \int_{y=0}^{\infty} y \, \frac{1}{a^2} e^{-y/2a^2} \, \mathrm{d}y \\ &= 2a^2 \end{aligned} \tag{13.15}$$

where we made use of the integrations tables in Appendix A. Therefore the shape parameter for a Rayleigh distributed fading signal should be:

$$a = \sqrt{\frac{P}{2}} \tag{13.16}$$

**2.** Substituting the scale parameter into the pdf for the received signal amplitude in (13.11), we get:

$$f_X(x) = \frac{2x}{P} e^{-x^2/P}, \qquad x \geq 0, \quad a > 0 \tag{13.17}$$

From (13.14), the pdf for the received signal power is given by:

$$f_Y(y) = \frac{1}{P} e^{-y/P} \qquad y \geq 0 \tag{13.18}$$

**3.** Using the CDF expression for the CDF of the Rayleigh distribution in (1.55) on page 21, the CDF for the received signal amplitude is given by:

$$F_X(x) = 1 - e^{-x^2/P} \tag{13.19}$$

The CDF for the received signal power is given by:

$$F_Y(y) = 1 - e^{-y/P} \tag{13.20}$$

**4.** Using (13.17), the expected value for the received signal amplitude is given by:

$$\overline{x} = \int_0^{\infty} x f_X(x) = \frac{1}{2} \sqrt{\pi P} \tag{13.21}$$

Using (13.18), the expected value for the received signal power is given by:

$$\overline{y} = \int_0^\infty y f_Y(y) = P \tag{13.22}$$

As expected. ■

*Example 13.3.* Find the root mean square (RMS) value of the received signal amplitude for Example 13.2.

The RMS value of $x$ is $x_{rms}$ and is given by:

$$x_{rms} = \sqrt{\int_0^\infty x^2 \times \frac{2x}{P} e^{-x^2/P} \, \mathrm{d}x}$$

We change variables $y = x^2$ and above equation becomes:

$$\begin{aligned} x_{rms} &= \sqrt{P \int_0^\infty y e^{-y} \, \mathrm{d}y} \\ &= \sqrt{P} \end{aligned}$$

■

Figure 13.3 shows the pdf and CDF for the received signal amplitude for a Rayleigh fading channel for the case when average received signal power is $P = 10\,\mathrm{mW}$. Figure 13.3a is the pdf distribution and Fig. 13.3b is the corresponding CDF.

Figure 13.4 shows the pdf and CDF for the received signal power for a Rayleigh fading channel for the case when average received signal power is $P = 10\,\mathrm{mW}$. Figure 13.4a is the pdf distribution and Fig. 13.4b is the corresponding CDF. There are other models for fading channels such as the Riccian [3] and Nakagami-$m$ [4].

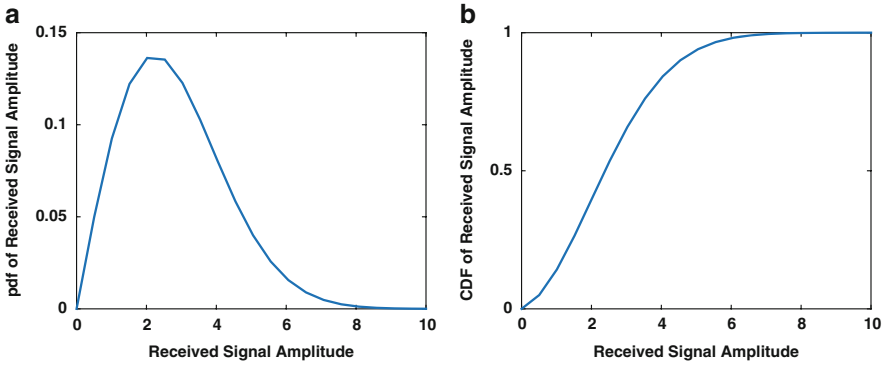



**Fig. 13.3** pdf and CDF for the received signal amplitude for a Rayleigh fading channel for the case when average received signal power is $P = 10\,\mathrm{mW}$. (**a**) The pdf distribution. (**b**) The CDF distribution

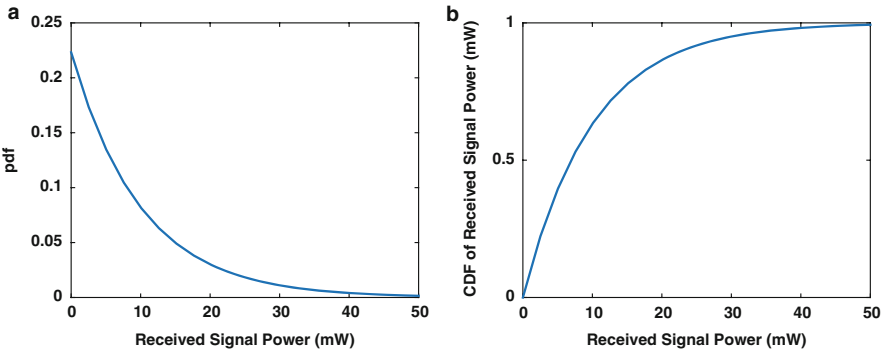



**Fig. 13.4** pdf and CDF for the received signal power for a Rayleigh fading channel for the case when average received signal power is $P = 10\,\text{mW}$. (**a**) The pdf distribution. (**b**) The CDF distribution

*Example 13.4.* Given the pdf of the Rayleigh distribution model for the fading channel, obtain a trace for the received signal amplitude and power.

Let us assume that we need to generate a 100-sample trace and that the average received signal power is 10 mW. We start by obtaining a table for the values of the pdf of the received signal amplitude and power, shown in Fig. 13.3a or Fig. 13.4a, respectively. We ensure that the resulting pdf distribution is normalized and adds up to unity.

Second step is to obtain a table for the values of the CDF shown in Fig. 13.3b. We accumulate the values of the pdf to get the CDF. We ensure that the resulting CDF distribution is normalized and that its maximum value is unity.

To generate random numbers that follow the Rayleigh distribution, we use the inversion method discussed in Chap. 1. We must ensure that the resulting trance is normalized such that the average of the samples is 10 mW. Figure 13.5 shows 100 samples of received signal amplitude and power following the Rayleigh fading model for the case when $P = 10\,\text{mW}$. Figure 13.5a shows a 100-sample trace of the received signal amplitude and Fig. 13.5b shows a 100-sample trace of the received signal power. ■

Other channel fading models include Ricean fading, log-normal distribution, and Nakagami fading. Goldsmith [1], Proakis [2] and Stuber [3] provide detailed discussions about those other models and why they are needed.

## 13.6 Level Crossing Rate

Figure 13.5 shows how the received signal amplitude or power changes over time. We can assign a fade state to the channel at any given time instance based on the amplitude or power of the received signal. Figure 13.6 shows the division of the

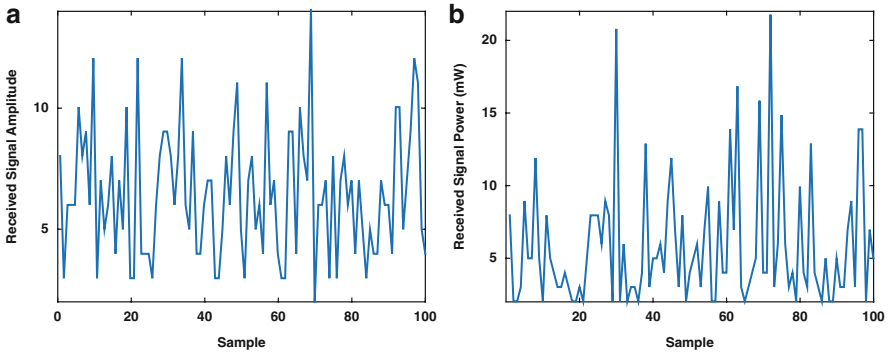



**Fig. 13.5** 100 samples of received signal amplitude and power following the Rayleigh fading model for the case when $P = 10\,\text{mW}$. (**a**) The amplitude trace. (**b**) The power trace

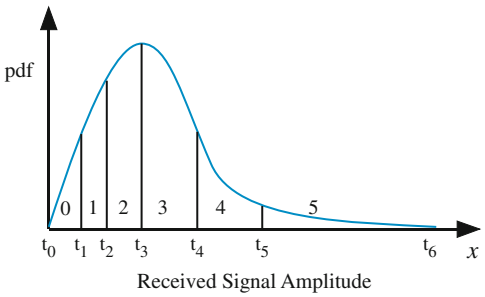



**Fig. 13.6** The received signal amplitude is divided into zones $z_k$ based on its value

channel into states based on the signal amplitude or power. The figure shows the division of the received signal amplitude into six zones where $t_0 = 0$ and $t_6 = \infty$.

The choice of the thresholds $t_k$ between zones is arbitrary. Figure 13.4 showed the pdf and CDF of received signal power. We can choose a range for the power based on the signal mean $\mu$ and standard deviation $\sigma$. The power range could be given as $\mu + 3\sigma$. Since $\mu = P$ and $\sigma = P$, the power range is given by:

$$\mu + 3\sigma = 4P \tag{13.23}$$

The power range could be divided into zones of equal values in the linear scale:

$$(K-1)\Delta \approx \mu + 3\sigma = 4P \tag{13.24}$$

$$\Delta = \frac{4P}{K-1} \tag{13.25}$$

where $K$ is the number of zones with $t_K = \infty$ and $\Delta$ is the step size chosen to cover the expected power range.

The values of the thresholds can be written as:

$$t_k = \Delta\, k, \qquad 0 \leq k < K \tag{13.26}$$

with $t_K = \infty$.

Another choice for the thresholds is to have equal distribution in the logarithmic scale:

$$\sum_{k=1}^{K-1} \Delta^k \approx \mu + 3\sigma, \qquad \Delta \neq 1 \tag{13.27}$$

$$\frac{1-\Delta^K}{1-\Delta} = \mu + 3\sigma + 1 = 4P + 1 \tag{13.28}$$

We can use the above equations to solve for the value of $\Delta$. The ordering of the thresholds depends on the value of $\Delta$. When $\Delta > 1$, the thresholds can be written as:

$$t_0 = 0 \tag{13.29}$$

$$t_k = \Delta^k, \qquad 0 < k < K \tag{13.30}$$

When $\Delta > 1$ the highest-index threshold corresponds to largest received power value. In that case $t_0 < t_1 < t_2 \cdots$. The thresholds will be given by:

$$t_0 = 0, t_1 = \Delta, t_2 = \Delta^2, \cdots t_{K-1} = \Delta^{K-1}, t_K = \infty \tag{13.31}$$

On the other hand, when $\Delta < 1$ the high-index threshold corresponds to the least power value. In that case $t_K < t_{K-1} < t_{K-2} \cdots$. The thresholds will be given by: the thresholds will be given by:

$$t_K = 0, t_{K-1} = \Delta^{K-1}, t_{K-2} = \Delta^{K-2}, \cdots t_1 = \Delta, t_0 = \infty \tag{13.32}$$

According to [5, 6] the number of times the received signal power crosses a power level zone to the next lower power level zone is given by:

$$N_y = \sqrt{\frac{2\pi y}{P}}\, f_m e^{-y/P} \tag{13.33}$$

where $f_m = v/\lambda$ is the maximum Doppler shift with $v$ being the speed of the moving object and $\lambda$ is the signal wavelength.

Using (13.33), we are now ready to derive an expression for the level crossing rate at a signal power threshold $t_k$:

$$N_k = \sqrt{\frac{2\pi t_k}{P}}\, f_m e^{-t_k/P}, \qquad 0 < k < K \tag{13.34}$$

where $N_k$ is taken as the rate of crossing from the higher power level to the lower power level. Note that the index $k$ does not equal the edge values 0 or $K$.

## 13.7 Markov Chain Model for Wireless Narrowband Fading Channels

Figure 13.5 showed the random fluctuations in the received signal amplitude and power. Figure 13.6 shows the pdf of received signal amplitude distribution after it has been divided into zones and each zone is defined based on the amplitude being between two thresholds. For example, the figure shows six zones defined by the thresholds $t_0, t_1, \cdots, t_5$ and $t_6 = \infty$.

We can assign a state with each zone and we would have the channel state vector $\mathbf{s}$ given by:

$$\mathbf{s} = [s_0 \; s_1 \; \cdots \; s_{K-1}]^t \tag{13.35}$$

where $K$ is the number of fade states of the channel. State $s_k$ denotes the probability that the channel is in state $k$, where the signal power lies between the thresholds $t_k \leq y < t_{k+1}$.

Figure 13.7 shows the states of the channel and the transition probabilities among the states. According to the slow fade conditions, it is widely assumed that the transitions are confined to same or the immediate next neighboring state only. The transition probability $p_{i,j}$ indicates the probability of making a transition from state $j$ to state $i$. The associated state transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix} p_{0,0} & p_{0,1} & 0 & \cdots & 0 & 0 \\ p_{1,0} & p_{1,1} & p_{1,2} & \cdots & 0 & 0 \\ 0 & p_{2,1} & p_{2,2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & p_{K-2,K-2} & p_{K-2,K-1} \\ 0 & 0 & 0 & \cdots & p_{K-1,K-2} & p_{K-1,K-1} \end{bmatrix} \tag{13.36}$$

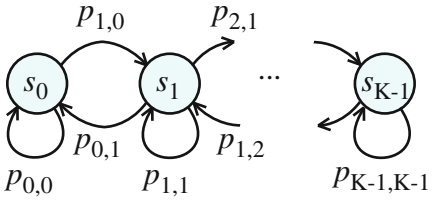



**Fig. 13.7** States of a fading wireless channel

From the figure we have a tridiagonal state transition matrix with entry $p_{i,j}$ in row $i$ and column $j$ is the probability the channel making a transition from state $j$ to state $i$, where $i = j$ or $i = j \pm 1$. The assumption of a simple birth–death process is justified by the slow fade condition compared with the data rate.

The traditional Markov chain analysis of fading states starts by making the fundamental assumption that the two super and subdiagonal of **P** are equal. Following this, it is customary to find the value of the state vector **s**. However, Problem 13.1 assures us that a tridiagonal Markov matrix with equal sub and superdiagonals (i.e., symmetric matrix) will produce a state vector whose components are equal. Specifically:

$$\mathbf{s} = \begin{bmatrix} 1/K & 1/K & \cdots & 1/K \end{bmatrix}^t \tag{13.37}$$

We adopt the assumption that the $k$th component of the distribution vector is given by:

$$\begin{aligned} s_k &= F_Y(t_{k+1}) - F_Y(t_k) \\ &= e^{-t_k/P} - e^{-t_{k+1}/P} \end{aligned} \tag{13.38}$$

where the received power CDF was obtained in (13.20).

Thus we study the states of the fading channel by first determining the following quantities:

1. The received signal power range.
2. The values of the thresholds $t_k$ defining the zones.

Using (13.38) we are able to obtain the values of the distribution vector **s**. Having found **s**, we can obtain the state transition matrix **P** under the assumption that the matrix is tridiagonal but the upper and lower diagonals are not equal. We can use forward or back substation to find the components of **P**. Problem 13.2 is related to finding a technique for estimate **P** knowing the distribution vector **s**.

The transition probabilities in the superdiagonal are given by:

$$p_{k,k-1} = \frac{N_k}{R_k} \tag{13.39}$$

where $N_k$ was given in (13.34) and $R_k$ is the number of symbol transmitted per second when the channel is in state $s_k$. $R_k$ is given by:

$$R_k = R \times s_k \tag{13.40}$$

where $R$ is the symbol transmission rate. Due to the slow fade conditions, we have $N_k \ll R_k$.

Because **P** is column stochastic, the following condition must be satisfied for each column:

$$p_{k-1,k} + p_{k,k} + p_{k+1,k} = 1; \qquad 0 \leq k < K \tag{13.41}$$

At steady state we can write:

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{13.42}$$

Equations (13.41) and (13.42) can be used to find all the components of **P** using forward or back substation.

The main parameters for our problem are:

1. The number of states $K$ for the channel states.
2. The values of the $K$ thresholds $t_0, t_1, \cdots, t_{K-1}$.
3. The mean received signal power $P$.
4. The maximum Doppler shift $f_m$, which depends in turn on the signal frequency and speed of moving objects.
5. The symbol transmission rate $R$.

## 13.8 Bit Error Probability

The wireless communication channel corrupts the received signal by adding random noise. Typically this random noise is treated as additive white Gaussian noise (AWGN). Channel fading affects the signal to noise ratio (SNR) since the received signal strength varies. Within the symbol transmission interval $T$, the received signal amplitude $r$ can be written as:

$$r = x + n \tag{13.43}$$

where $n$ is the random variable corresponding to the AWGN with zero mean and infinite variance. For digital communications, we use the noise power spectral density $N_0$ (W/Hz). The pdf for the noise signal amplitude is given by:

$$f_N(n) = \frac{1}{\sqrt{\pi N_0}} e^{-n^2/N_0} \tag{13.44}$$

where the variance of the noise is $\sigma^2 = N_0/2$.

For the case of binary signalling, we send either of two symbols $x_1$ or $x_2$. In that case (13.43) becomes:

$$r = \pm\sqrt{N_b} + n \tag{13.45}$$

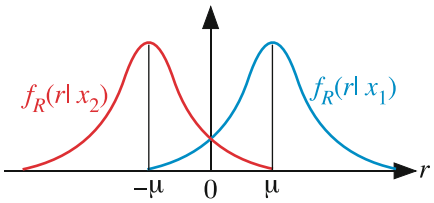



**Fig. 13.8** Conditional probability density functions $f_R(r|x_1)$ and $f_R(r|x_2)$ that correspond to the two situations when symbol $x_1$ or $x_2$ is transmitted

where $N_b$ is the energy used to transmit the symbol $x_1$ or $x_2$, respectively. The more general transmission system comprising $M$-ary symbols is dealt with in many excellent textbooks on wireless communications such as Goldsmith [1], Proakis [2], Stuber [3], Jakes [6], Sklar [7], and Rappaport [8], to name a few.

Figure 13.8 shows the conditional probability density functions $f_R(r|x_1)$ and $f_R(r|x_2)$ that correspond to the two situations when symbol $x_1$ or $x_2$ is transmitted with equal probability. When symbol $x_1$ is transmitted the conditional pdf is given by:

$$f_R(r|x_1) = \frac{1}{\sqrt{\pi N_0}} e^{-(r-\mu)^2/N_0} \tag{13.46}$$

where $\mu = \sqrt{N_b}$ is the mean of the received signal when symbol $x_1$ is transmitted. When symbol $x_2$ is transmitted, the conditional pdf is given by:

$$f_R(r|x_2) = \frac{1}{\sqrt{\pi N_0}} e^{-(r+\mu)^2/N_0} \tag{13.47}$$

The probability of error occurs when symbol $x_2$ is detected while symbol $x_1$ was actually sent or the opposite situation. Thus we can write the probability of error

$$\begin{aligned} p_e &= \frac{1}{2}\left[p(x_2|x_1) + p(x_1|x_2)\right] \\ &= \int_0^{\infty} f_R(r|x_2)\, \mathrm{d}r \\ &= \int_{-\infty}^{0} f_R(r|x_1)\, \mathrm{d}r \end{aligned} \tag{13.48}$$

We can simplify the above equations to the form:

$$p_e = \frac{1}{\sqrt{2\pi}} \int_z^{\infty} e^{-r^2/2}\, \mathrm{d}r \tag{13.49}$$

$$= Q\left(\sqrt{\frac{E_b}{N_0}}\right) \tag{13.50}$$

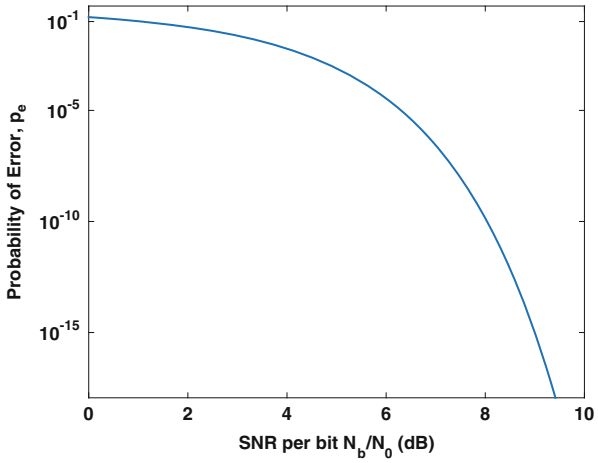



**Fig. 13.9** Probability of error $p_e$ vs. SNR per bit for binary signals

$Q(x)$ is called the $Q$-function and the quantity $E_b/N_0$ is the signal to noise ratio per bit. We should note that the quantity $E_b/N_0$ is dimensionless. As expected, the error probability depends on the strength of the received signal and the level of noise.

The $Q$-function is close to the complementary error function *erfc* by the following relation:

$$Q(x) = \frac{1}{2}\text{erfc}\left(\frac{x}{\sqrt{2}}\right) \tag{13.51}$$

*Example 13.5.* Plot the probability of error $p_e$ versus SNR per bit $(N_b/N_0)$.

We apply Eq. (13.51) for the range of SNR between 0 and 10 dB. The result is shown in Fig. 13.9. ■

In order to find the bit error probability when the channel is in a given state $s_k$, we need to find the average power associated with that state. The average power $P_k$ is given using (13.18) by:

$$\begin{aligned} P_k &= \int_{t_k}^{t_{k+1}} y f_Y(y)\mathrm{d}y \\ &= \int_{t_k}^{t_{k+1}} \frac{y}{P} e^{-y/P}\mathrm{d}y \end{aligned} \tag{13.52}$$

Changing the variable to $x = y/P$, we get:

$$\begin{aligned} P_k &= P \int_{x=t_k/P}^{t_{k+1}/P} x e^{-x} \mathrm{d}x \\ &= -P\,(x+1)\,|_{x=t_k/P}^{t_{k+1}/P} \\ &= (t_k + P)e^{-t_k/P} - (t_{k+P} + 1)e^{-t_{k+1}/P} \end{aligned} \tag{13.53}$$

The energy of the received signal is given by:

$$E_k = P_k T = \frac{P_k}{R} \tag{13.54}$$

where $T$ is the duration of the symbol during transmission.

The signal to noise ratio when the channel is in state $s_k$ is then given by:

$$e_k = Q\left(\frac{E_k}{N_0}\right) \tag{13.55}$$

The average bit error rate is therefore given by:

$$e_b = \sum_k \mathbf{e}^t\,\mathbf{s} \tag{13.56}$$

where $\mathbf{e}$ is the error vector given by:

$$\mathbf{e} = \left[\, e_0\; e_1 \cdots e_{K-1} \,\right]^t \tag{13.57}$$

## 13.9 Numerical Results

Let us consider a wireless system having the parameters in Table 13.1.

We need to calculate the channel state distribution vector $\mathbf{s}$ which will allow us to estimate the bit error rate (BER).

**Table 13.1** Numerical simulation parameters

| Parameter | Symbol | Value |
|---|---|---|
| Transmission rate | $R$ | 900 MHz |
| Number of states | $K$ | 6 |
| Mean received signal power | $P$ | 10 mW |
| Signal frequency | $f_c$ | 900 MHz |
| Maximum speed | $v$ | 10 m/s |
| Noise power spectral density | $N_0$ | 10 μW/Hz |

### 13.9.1 Linear Scale Results

We choose to divide our pdf into zones according to the equal distributions in the linear scale as in (13.25). Therefore we have

$$\Delta = \frac{40}{5} = 8\,\text{mW} \tag{13.58}$$

From (13.26) the thresholds vector is given by:

$$t_k = \begin{bmatrix} 0 & 8 & 16 & 24 & 32 & 40 & \infty \end{bmatrix}^t \tag{13.59}$$

From (13.38) and (13.59) we are able to obtain the distribution vector:

$$\mathbf{s} = \begin{bmatrix} 0.5507 & 0.2474 & 0.1112 & 0.0500 & 0.0224 & 0.0183 \end{bmatrix}^t \tag{13.60}$$

From (13.34), the level crossing rate vector is given by

$$N_k = \begin{bmatrix} 26.6346 & 30.9400 & 32.0637 & 27.0650 & 15.8011 \end{bmatrix}^t \tag{13.61}$$

Now we are able to obtain the corresponding state transition matrix as:

$$\mathbf{P} = \begin{bmatrix} 0.9995 & 0.0012 & 0 & 0 & 0 & 0 \\ 0.0005 & 0.9980 & 0.0017 & 0 & 0 & 0 \\ 0 & 0.0008 & 0.9973 & 0.0021 & 0 & 0 \\ 0 & 0 & 0.0010 & 0.9968 & 0.0024 & 0 \\ 0 & 0 & 0 & 0.0011 & 0.9963 & 0.0015 \\ 0 & 0 & 0 & 0 & 0.0012 & 0.9985 \end{bmatrix} \tag{13.62}$$

From the structure of the system matrix, we can conclude that the channel will remain in the same state at the next time state with high probability, as indicated by the values of the diagonal terms $p_{k,k}$. Also, if the channel is going to change states, the channel will tend to switch to the lower energy state, as indicated by the super diagonal terms $p_{k,k-1}$.

Using (13.55), the bit error probability for each state is given by:

$$e_k = \begin{bmatrix} 0.0279 & 0.0023 & 0.015 & 0.085 & 0.2129 & 0.1799 \end{bmatrix}^t \tag{13.63}$$

Using (13.56) the overall average bit error rate is given by:

$$e_b = 2.9951 \times 10^{-2} \tag{13.64}$$

### 13.9.2 Logarithmic Scale Results

We choose to divide our pdf into zones according to the equal distributions in the logarithmic scale as in (13.28). Therefore we have

$$1 - \Delta^5 = 41(1 - \Delta) \tag{13.65}$$

Thus we have $\Delta = 1.7970\,\text{mW}$.

From (13.31) the thresholds vector is given by:

$$t_k = \begin{bmatrix} 0 & 1.7970 & 3.2294 & 5.8034 & 10.4289 & 10.4289 & \infty \end{bmatrix}^t \tag{13.66}$$

From (13.38) and (13.66) we are able to obtain the distribution vector:

$$\mathbf{s} = \begin{bmatrix} 0.1645 & 0.1115 & 0.1643 & 0.2073 & 0.1989 & 0.1535 \end{bmatrix}^t \tag{13.67}$$

From (13.34), the level crossing rate vector is given by

$$N_k = \begin{bmatrix} 26.6346 & 30.9400 & 32.0637 & 27.0650 & 15.8011 \end{bmatrix}^t \tag{13.68}$$

Now we are able to obtain the corresponding state transition matrix as:

$$\mathbf{P} = \begin{bmatrix} 0.9991 & 0.0013 & 0 & 0 & 0 & 0 \\ 0.0009 & 0.9979 & 0.0005 & 0 & 0 & 0 \\ 0 & 0.0008 & 0.9992 & 0.0021 & 0 & 0 \\ 0 & 0 & 0.0010 & 0.9968 & 0.0002 & 0 \\ 0 & 0 & 0 & 0.0003 & 0.9997 & 0.0001 \\ 0 & 0 & 0 & 0 & 0.0001 & 0.9999 \end{bmatrix} \tag{13.69}$$

From the structure of the system matrix, we can conclude that the channel will remain in the same state at the next time state with high probability, as indicated by the values of the diagonal terms $p_{k,k}$. Also, if the channel is going to change states, the channel will tend to switch to the lower energy state, as indicated by the super diagonal terms $p_{k,k-1}$.

Using (13.55), the bit error probability for each state is given by:

$$e_k = \begin{bmatrix} 0.0279 & 0.0023 & 0.015 & 0.085 & 0.2129 & 0.1799 \end{bmatrix}^t \tag{13.70}$$

Using (13.56) the overall average bit error rate is given by:

$$e_b = 0.1654 \tag{13.71}$$

## 13.10 Problems

**13.1.** The traditional Markov chain analysis of fading states starts by making the fundamental assumption that the two super and subdiagonal are equal (i.e., a symmetric tridiagonal matrix). Prove that a tridiagonal Markov matrix with equal sub and superdiagonals will produce a state vector whose components are equal. Specifically:

$$\mathbf{s} = \left[ 1/K \;\; 1/K \cdots 1/K \right]^t \tag{13.72}$$

where $K$ is the number of rows or columns of $\mathbf{P}$.

**13.2.** Assume that the state distribution vector $\mathbf{s}$ has been estimated as was explained in Sect. 13.7. Since the channel state transition matrix is tridiagonal, it is possible to estimate $\mathbf{P}$ using forward or back substitution. Attempt both techniques.

**13.3.** Prove that the signal to noise ratio $E_b/N_0$ is dimensionless.

**13.4.** Using Eq. (13.51) on page 427 prove that the error probability for symmetric binary modulation is below 0.5; i.e., $p_e < 0.5$.

## References

1. A. Goldsmith, *Wireless Communications* (Cambridge University Press, New York, 2005)
2. J.G. Proakis, *Digital Communications* 3rd edn. (McGraw-Hill, New York, 1995)
3. G.L. Stuber, *Principles of Mobile Communications*, 3rd edn. (Springer, New York, 2011)
4. M.D. Yacoub, J.E.V. Bautistu, L. Guerra de Rezende Guedes, On higher order statistics of the Nakagami-*m* distribution. IEEE Trans. Veh. Technol. **48**, 790–794 (1999)
5. H.S. Wang, N.R. Moayeri, Finite-state Markov channel: a useful model for radio communication channels. IEEE Trans. Veh. Technol. **44**(1), 163–171 (1995)
6. W.C. Jakes, *Microwave Mobile Communications* (Wiley, New York, 1974)
7. B. Sklar, *Digital Communications*, 2nd edn. (Prentice Hall, Upper Saddle River, 2001)
8. T.S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd edn. (Prentice Hall, Upper Saddle River, 2001)

# Chapter 14
# Software Defined Radio

## 14.1 Introduction

Conventional radio (CR) networks require a dedicated channel with little or no interference and provides a fixed functionality for all users. A software defined radio in general can customize its functionality and performance to provide:

- Service and functionality personalization.
- Adapting its medium access protocol and modulation strategy to respond to varying transmission channel impairments.
- Make use of *white space* or inactivity in the assigned radio frequency spectrum.
- Support latest wireless protocols and be fully programmable to support next generation interfaces and standards.
- Take advantage of performance-increasing innovations and over-the-air updates [1, 2].

It was found out that the licensed bands are under-utilized. Figure 14.1 shows the use of the radio frequency bands allocated to three users over time. Utilization of an assigned radio spectrum could be as low as 10 %.

We see that each user transmits data according to its traffic load characteristics. The frequency bands are idle at different times and other users could opportunistically use these bands if they have a means of rapidly sensing the idle channels.

This is the essence of cognitive, and opportunistic radio. Cognitive radio employs agile transmission strategies that can adapt to the channel and has been an active research topic since the early 2,000 s. Cognitive radio relies on software to give it the required responsiveness and adaptability.

We can therefore define software defined radio (SDR) as a radio system or device that uses software to select the parameters of operation. SDR dynamically modifies the communication protocol between the communicating nodes. SDR continuously monitors its own transmission and the channel parameters and channel activity.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_14

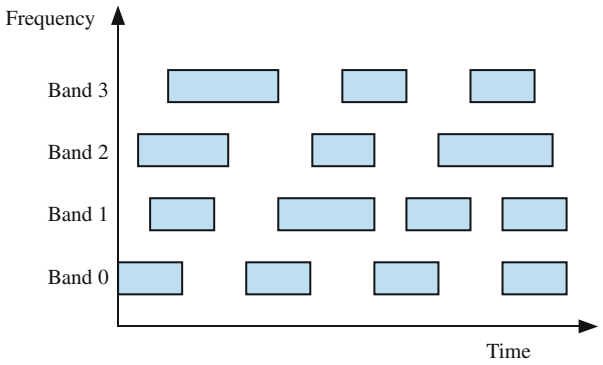
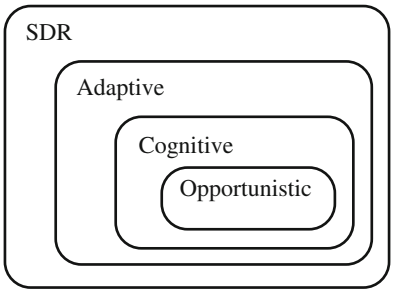



**Fig. 14.1** Assigned radio spectrum utilization over time for the case of four primary users

**Fig. 14.2** SDR technology hierarchy




It can change its transmit parameters such as power, modulation scheme, or search for spectrum bands that are unused or have small channel impairments.

Figure 14.2 shows SDR technology hierarchy.

Adaptive radio is system or device monitors its own performance and varies its parameters to improve its performance.

Cognitive radio (CR) uses software to automatically adjust its operation to achieve a desired objective.

Opportunistic radio uses CR technology to continuously monitor the radio frequency spectrum to search for idle bands; i.e., white space opportunistic usage.

## 14.2 Modeling Adaptive Radio

In an adaptive radio, the system or device monitors its performance and varies its parameters to improve performance. Typically the parameter to be changed is in data modulation and coding [3]. Figure 14.3 shows an adaptive SDR system. The transmitter could send two types of signals: the normal data signal (shows by the top arrow) and the pilot signal (shown by the bottom arrow). The data signal is modulated and coded by the transmitter and is demodulated and decoded by the

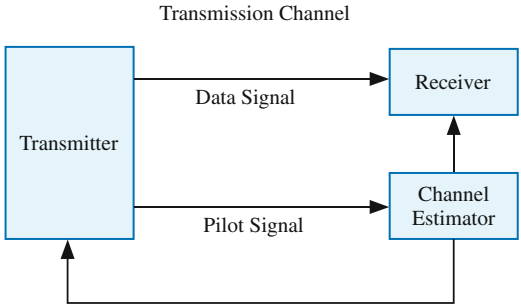



**Fig. 14.3** Adaptive SDR system

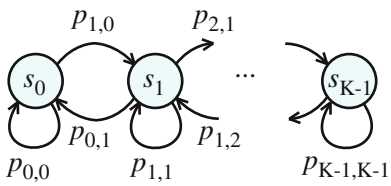



**Fig. 14.4** States of a fading wireless channel

receiver. The pilot signal is sent to the channel estimator which provides an estimate of the channel state and accordingly informs the transmitter and receiver to adjust their parameters. However, certain time delay is encounters and it is possible that the produced estimate could be outdated if the channel changes its state too quickly.

Let us now model the channel states as a Markov chain as shown in Fig. 14.4. We assume that the wireless channel can exist in $n$ states $s_0$, $s_1$, $\cdots$, and $s_{n-1}$. The channel state vector is given by:

$$\mathbf{s} = \begin{bmatrix} s_0 \; s_1 \; \cdots \; s_{n-1} \end{bmatrix}^t \tag{14.1}$$

The associated state transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix} p_{0,0} & p_{0,1} & 0 & \cdots & 0 & 0 \\ p_{1,0} & p_{1,1} & p_{1,2} & \cdots & 0 & 0 \\ 0 & p_{2,1} & p_{2,2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & p_{K-2,K-2} & p_{K-2,K-1} \\ 0 & 0 & 0 & \cdots & p_{K-1,K-2} & p_{K-1,K-1} \end{bmatrix} \tag{14.2}$$

From the figure we have a tridiagonal state transition matrix with entry $p_{i,j}$ in row $i$ and column $j$ is the probability the channel making a transition from state $j$ to state $i$. Chapter 13 explained how the channel fade state can be modeled. The chapter also shows how the bit error rate can be modeled as the error vector **e** associated with each fade state:

$$\mathbf{e} = \left[ e_0 \; e_1 \cdots e_{K-1} \right]^t \tag{14.3}$$

Knowing **s** and **e** we are able to get some performance figures for the adaptive radio (AR).

The most obvious measure is the average bit error rate (BER) which can be estimated as:

$$\begin{aligned} BER &= \mathbf{e}^t \mathbf{s} \\ &= \sum_{j=0}^{K-1} e_j s_j \end{aligned} \tag{14.4}$$

For an adaptive radio, the main idea is to vary the transmission parameters to suit an anticipated channel state such that the error probability will be low. The problem happens when the actual future channel state is not what was anticipated.

We make the following simplifying assumptions to facilitate modeling AR systems:

1. The system transition matrix is tridiagonal.
2. $\overline{p_i}$ is the highest component in column $i$ of the transition matrix **P**.
3. $\breve{e}_i$ is the error probability when the actual channel state matches the estimated next state.
4. $\hat{e}_i$ is the error probability when the actual channel state does not match the estimated next state.

Based on the above assumptions, the bit error probability when the system is in state $j$ is given by:

$$\epsilon_j = \breve{e}_j \overline{p_j} + \hat{e}_j (1 - \overline{p_j}) \tag{14.5}$$

Under these conditions, the resulting average bit error rate is given by:

$$BER = \sum_{j=0}^{K-1} \epsilon_j s_j \tag{14.6}$$

The above equation reduces to (14.4) when adaptive techniques are not used.

## 14.3 Modeling Opportunistic Spectrum Access (ALOHA Access)

In this section we discuss modeling opportunistic radio, or opportunistic spectrum access that uses the ALOHA medium access control protocol to access the available channel. Here the device continuously monitors the radio spectrum for white space or idle channels. Figure 14.1 showed the available white space in the radio spectrum.

The radio spectrum bands shown are assigned or allocated to the primary users. Secondary users are allowed to access these bands opportunistically when a primary user is idle and the particular band is free.

Let us discretize the time into time slots such that the system state changes in each time slot depending on the activity of the primary users.

Let us start by the following simplifying assumptions:

1. There are $N$ primary users and each user is assigned one of $N$ available spectrum bands.
2. The state of activity of each primary user does not depend on its state in the previous time slots.
3. There are $M$ secondary users that are continuously monitoring the $N$ spectrum regions.
4. Probability that a primary user is active at a given time slot is $a$.
5. Probability that a secondary user is active at a given time slot active is $c$.
6. Active secondary users randomly choose one of the available channels.
7. Active secondary users adopt the ALOHA medium access control scheme to start transmitting when they sense the primary user is idle.

Because of assumption 2, we can model our system as a discrete-time Markov chain. The number of states of the system is $N + 1$ where $s_i$ is the state where there are $i$ active primary users in a time slot.

Probability $n_i$ that there are $i$ available channels/bands, out of an available $N$ channels, at a given time slot is given by:

$$n_i = \binom{N}{i}(1-a)^i a^{N-i} \tag{14.7}$$

The above equation indicates that $i$ available channels occur when any $i$ primary users are *inactive* and the other $N - i$ primary users are active.

The state vector $\mathbf{s}$ for the available channel Markove chain is defined as:

$$\mathbf{s} = \begin{bmatrix} s_0 \; s_1 \cdots s_N \end{bmatrix}^t \tag{14.8}$$

where $s_i$ is the state when there are $i$ available channels. The associated state transition matrix is given by:

$$\mathbf{P} = \begin{bmatrix} n_0 & n_0 & n_0 & \cdots & n_0 \\ n_1 & n_1 & n_1 & \cdots & n_1 \\ n_2 & n_2 & n_2 & \cdots & n_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ n_N & n_N & n_N & \cdots & n_N \end{bmatrix} \tag{14.9}$$

We note that all the columns of $\mathbf{P}$ are identical. This special form implies that at steady state the distribution vector $\mathbf{s}$ is given by:

$$\mathbf{s} = \left[ n_0 \; n_1 \cdots n_N \right]^t \tag{14.10}$$

When the system is in state $s_i$, probability that a secondary user chooses an available channel is given by:

$$x_i = \frac{1}{i} \tag{14.11}$$

Let us now turn our attention to the secondary users. Probability $m_j$ that there are $j$ active secondary users, out of potential $M$ secondary users, at a given time slot is given by:

$$m_j = \binom{M}{j} c^j (1-c)^{M-j} \tag{14.12}$$

Assume the system is in state $s_i$ and there are $j$ secondary active users. We define $y_{i,j}$ as the probability that a tagged channel is available and is chosen by exactly one active secondary user when there are $i$ available channels and $j$ active secondary users. $y_{i,j}$ is given by:

$$y_{j,j} = \binom{i}{1} \left(\frac{1}{i}\right) \left(1 - \frac{1}{i}\right)^{j-1}, \qquad 1 \le i \le N, \quad 1 \le j \le M \tag{14.13}$$

When we have $N$ primary users and $M$ secondary users, the average secondary user throughput $Th(N, M)$ is given by

$$Th(N, M) = \sum_{i=1}^{N} s_i \sum_{j=1}^{M} y_{i,j} m_j \tag{14.14}$$

Figure 14.5 shows the secondary user throughput of an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.2$ and $c = 0.1$. Red, green, and blue lines are for $N = 1$, 2, and 4, respectively. As expected, secondary user throughput increases with the increasing values of $M$. Further, that throughput increases with increasing number of primary users $N > 1$. The case when $N = 1$ shows constant value for the throughput independent of $M$ because of two factors: a low value for $a = 0.2$ which indicates that 80 % of the time the channel is idle and ready to accommodate secondary user requests.

Let us now try to reverse the situation and have the activities of the primary users increase such that $a = 0.8$. Figure 14.6 shows the secondary user throughput of an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. Red, green, and blue lines are for $N = 1$, 2, and 4, respectively. Under these conditions, secondary user throughput is almost constant with the increasing values of $M$. This is because there is a small number of available

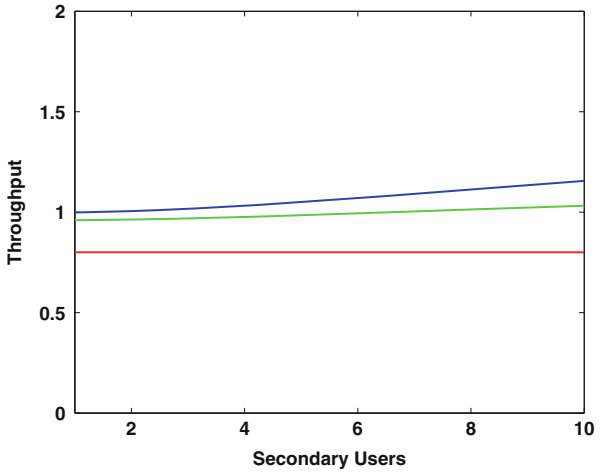



**Fig. 14.5** Throughput of an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.2$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

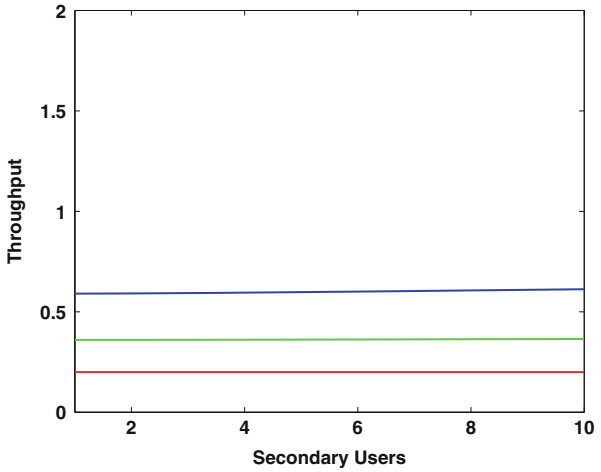



**Fig. 14.6** Throughput of an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

channels and they are always used by the secondary users. However, that throughput increases more significantly with increasing number of primary users $N$.

Probability of channel access is given by the ratio of system throughput divided by the number of secondary users:

$$p_a(N, M) = \frac{Th(N, M)}{M} \tag{14.15}$$

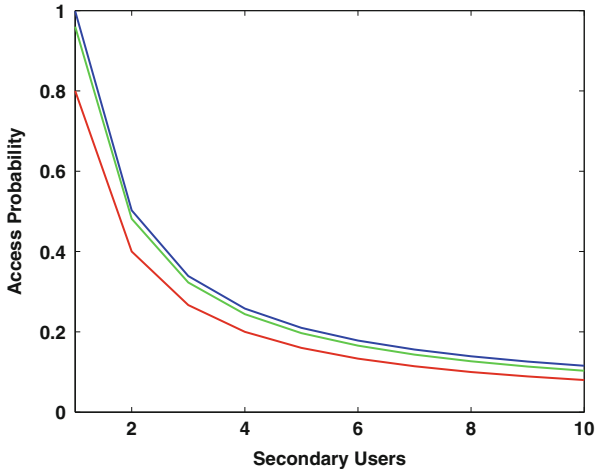



**Fig. 14.7** Access probability of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.2$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

Figure 14.7 shows the access probability of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.2$ and $c = 0.1$. As expected, the secondary user access probability decreases with increasing values of $M$ due to the increased competition from other secondary users. Also, the secondary user access probability increases (improves) with increasing number of primary users. This is due to the increased chances of finding an idle channel at a given time slot. However, we note that increasing values of $N$ beyond 2 does not really lead to increased values of $p_a$.

Figure 14.8 shows the access probability of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. The access probability $p_a$ is smaller compared to its values in Fig. 14.7 are more active and offer small number of available channels. As expected, the secondary user access probability decreases with increasing values of $M$ due to the increased competition from other secondary users. Also, the secondary user access probability increases (improves) with increasing number of primary users. This is due to the increased chances of finding an idle channel at a given time slot.

Beside estimating the average throughput and secondary user access probability, we are also interested in estimating how much energy and delay are needed before a secondary user frame is successfully transmitted. The average energy required to transmit a frame is estimated as

$$\begin{aligned} E &= E_0 \sum_{i=0}^{\infty} (i+1)(1-p_a)^i \; p_a \\ &= \frac{E_0}{p_a} \end{aligned} \tag{14.16}$$

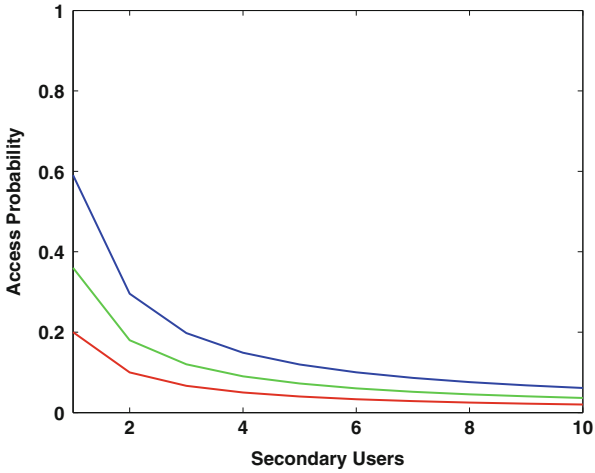



**Fig. 14.8** Access probability of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

where $E_0$ is the energy required to send the one frame. In dB, the above equation can be written as

$$E/E_0 = -10 \log_{10} p_a \quad \text{dB} \tag{14.17}$$

Figure 14.9 shows the average energy to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.2$ and $c = 0.1$. The average energy increases as the number of secondary users increases, as expected, due to increased collisions with other secondary users. Also, the average energy decreases as the number of primary users increases. This is due to the reduced chance of collision with other secondary users.

Figure 14.10 shows the average energy to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. We notice that the average energy required to transmit a packet increases above its values in Fig. 14.9 due to the increasing number of attempts to successfully access the channel.

The average number of attempts before a successful transmission is

$$n_a = \sum_{n=0}^{\infty} n \ (1 - p_a)^n \ p_a \tag{14.18}$$

This evaluates to

$$n_a = \frac{1 - p_a}{p_a} \tag{14.19}$$

Figure 14.11 shows the average delay to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4;

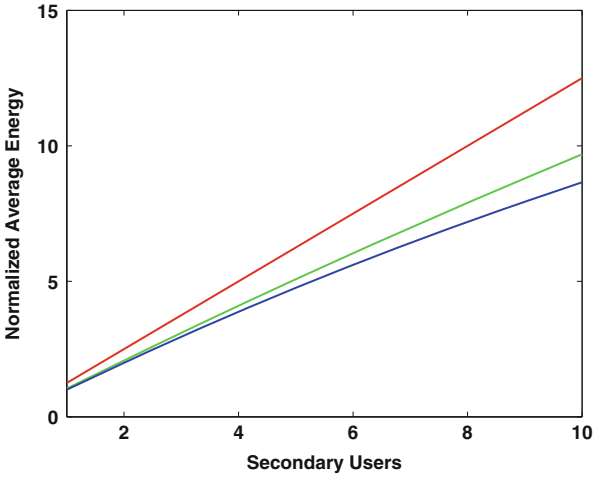



**Fig. 14.9** Average energy to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.2$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

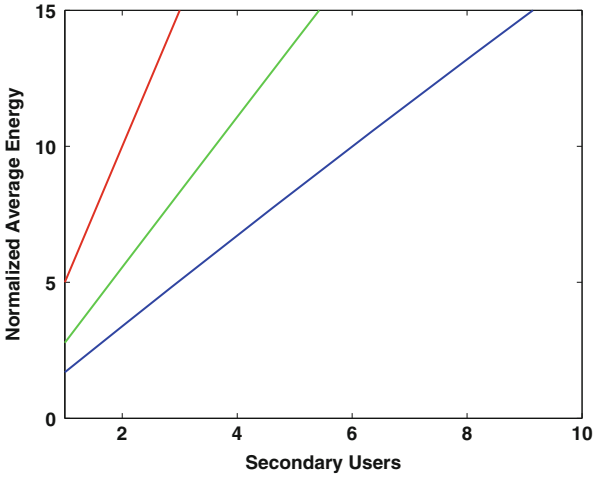



**Fig. 14.10** Average energy to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

and $M = 10$, $a = 0.2$ and $c = 0.1$. The average delay increases as the number of secondary users increases, as expected, due to increased collisions with other secondary users. Also, the average delay decreases as the number of primary users increases. This is due to the reduced chance of collision with other secondary users.

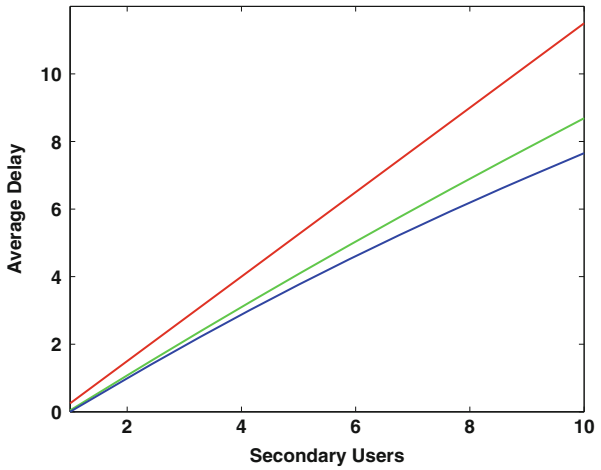



**Fig. 14.11** Average delay to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.2$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

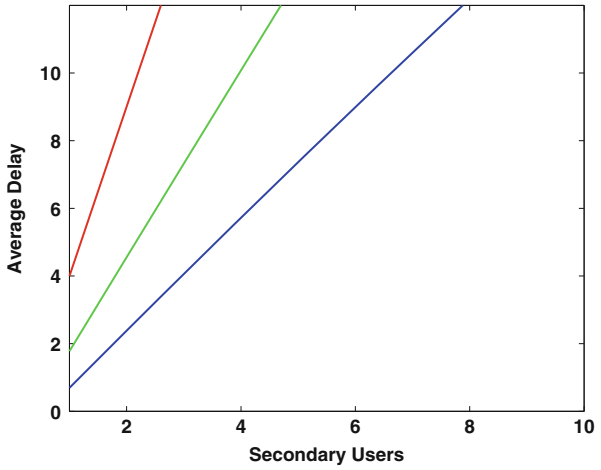



**Fig. 14.12** Average delay to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. *Red*, *green*, and *blue lines* are for $N = 1$, 2, and 4, respectively

Figure 14.12 shows the average delay to transmit a frame of a secondary user in an opportunistic SDR system for number of primary users being $N$ =1, 2, and 4; and $M = 10$, $a = 0.8$ and $c = 0.1$. As expected, the when the primary users are more active, the average delay for the secondary users increases due to the reduced number of available channels and the increased competition from the other secondary users.

## 14.4 Problems

**14.1.** Prove Eq. (14.10).

**14.2.** Prove that (14.6) reduces to (14.4) when adaptive techniques are not used.

**14.3.** Assume a fading channel having the parameters of Table 13.1. Estimate the average BER when adaptive modulation is not used versus when it is used. For this problem assume that $\check{e_j} = 0.1e_j$ and $\hat{e_j} = 10e_j$

**14.4.** In an AR radio system part of the adaptation could be to modulate the transmitter energy level so that $p_j$ is reduced when SNR is large and to increase $p_j$ for states with small SNR. How can one model average energy consumption and average BER when such a system is being used?

## References

1. Nvidia, *NVIDIA SDRF (Software Defined Radio, The modem innovation inside NVIDIA i500 and Tegra 4i*. http://www.nvidia.ca/docs/IO/116757/NVIDIA_i500_whitepaper_FINALv3.pdf
2. E.A.M. Klumperink, B. Nauta, *Software defined radio receivers exploiting noise cancelling: A tutorial review*. IEEE Communications Magazine, **52**(10), 111–117 (2014)
3. A. Goldsmith, *Wireless Communications* (Cambridge University Press, New York, 2005)

# Chapter 15
# Modeling Network Traffic

## 15.1 Introduction

Models that describe and generate telecommunication traffic are important for several reasons [1]:

*Traffic description*: Network users might be required to give a traffic description to the service provider. Based on that, the service provider decides whether the new connection can be admitted with a guaranteed quality of service (QoS) and without violating the QoS for established connections.

*System simulation*: Future networks and new equipment could be designed and the expected network performance checked.

Different models are used to describe different types of traffic. For example, voice traffic is commonly described using the on–off source or the Markov modulated Poisson process (MMPP). Studies suggest that traffic sources such as variable bit rate (VBR) video and Ethernet traffic are better represented by self-similar traffic models [2–7]. The important characteristics of a traffic source are its *average data rate*, *burstiness*, and *correlation*. The average data rate gives an indication of the expected traffic volume for a given period of time. Burstiness describes the tendency of traffic to occur in clusters. A traffic burst affects buffer occupancy and leads to network congestion and data loss. Data burstiness is manifested by the *autocorrelation function* which describes the relation between packet arrivals at different times. It was recently discovered that network traffic exhibits long-range dependence, i.e. the autocorrelation function approaches zero very slowly in comparison with the exponential decay characterizing short-range dependent traffic [2–7]. Long-range dependent traffic produces a wide range in traffic volume away from the average rate. This great variation in traffic flow also affects buffer occupancy and network congestion. In summary, high burstiness or long-term correlation leads to buffer overflow and network congestion. We begin by discussing the different models describing traffic time arrival statistics.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_15

Simple traffic models are sometimes called *point processes* since they are basically *counting processes* that count the number of packets that arrives in a time interval. These point processes sometime give the random sequence representing the time separations between packets. Several random processes are grouped together to give more complex traffic patterns. The Internet traffic archive (http://ita.ee.lbl.gov/index.html) provides data sets for network traffic and some useful software.

The other extreme for traffic modeling is to use fluid flow models. Fluid flow modeling groups the traffic into flows that are characterized by average and burst data rates. The object in these models is to investigate traffic at the *aggregate level* such as Ethernet traffic or traffic arriving at ingress and egress points of some Internet service provider (ISP). Fluid flow models do not concern themselves with the details of individual packet arrivals or departures.

The difference between point processes and fluid flow models is similar to the difference between describing an electric current in terms of the individual electrons or in terms of the current amplitude.

## 15.2 Flow Traffic Models

Flow traffic or fluid traffic models hide the details of the different traffics flowing in the network and replace them with flows that have a small set of characterizing parameters. The resulting models are easily generated, measured, or monitored.

For an end-to-end application, a *flow* has constant addressing and service requirements [19]. These requirements define a flow specification or *flowspec* which is used for bandwidth planning and service planning. *Individual flows* belonging to single sessions or applications are combined into *composite flows* that share the same path, link, or service requirements. Composite flows, in turn, are combined into *backbone flows* when the network achieves a certain level of hierarchy. Describing flows in this fashion makes it easier to combine flow characteristics and to work with a smaller set of data. For example, a core router might separate incoming data into individual flows, composite flows, and backbone flows depending on the QoS required by the users. This results in smaller number of service queues and simpler implementation of the scheduling algorithm implemented in the router. In most networks, the majority of the flows are low-performance backbone flows; there will also be some composite flows; and there will be few high-performance individual flows. The high-performance flows will influence the design of the scheduling algorithm in the switch, size, and number of the queues required since they usually have demanding delay and/or bandwidth requirements. The backbone flows will influence the buffer size required since they will usually constitute the bulk of the traffic most of the storage within the switch.

### 15.2.1 Modulated Poisson Processes

In a Markov modulated traffic model, states are introduced where the source changes its characteristic based on the state it is in. The state of the source could represent its data rate, its packet length, etc. When the Markov process represents data rate, the source can be in any of several active states and generates traffic with a rate that is determined by the state. This is commonly called MMPP. The simplest model is the on/off model and more complex models are described in the next section for video traffic.

### 15.2.2 On–Off Model

A popular model for bursty sources is the on–off model where the source switches between an active state, producing packets, and a silent state where no packets are produced. In that sense, the on–off model is a two-state MMPP. Traffic from this type of source is characterized by many variable length bursts of activity, interspersed with variable length periods of inactivity. This model is commonly used to describe constant bit rate (CBR) traffic in ATM [20–22]. Figure 15.1 shows the two-state model for the on–off source.

The source stays in the active state with probability $a$ and stays in the silent state with probability $s$. When the source is in the active state the source generates data at a rate $\lambda$ in units of bits/s or packets/s. The traffic pattern generated by this source is shown in Fig. 15.2.

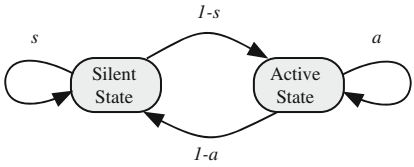



**Fig. 15.1** An on–off source model

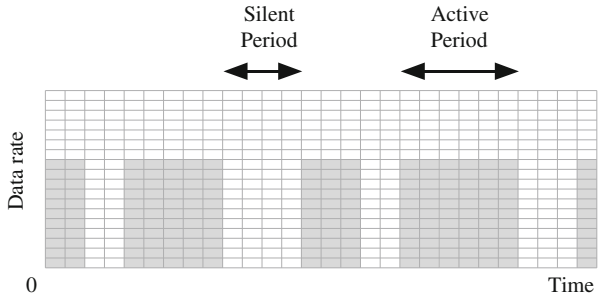



**Fig. 15.2** Packet pattern for an on–off source model

The probability that the length of the active period is $n$ time steps is given by the geometric distribution

$$A(n) = a^n(1-a) \quad n \geq 1 \tag{15.1}$$

The average duration of the active period is given by

$$T_a = \frac{a}{1-a} \text{ time steps} \tag{15.2}$$

Similarly, the probability that the length of the silent period is $n$ time steps is given by the geometric distribution

$$S(n) = s^n(1-s) \quad i \geq 1 \tag{15.3}$$

The average duration of the silent period is given by

$$T_s = \frac{s}{1-s} \text{ time steps} \tag{15.4}$$

Assume that $\lambda$ is the data rate when the source is in the active state. In that case, the average data rate is obtained as

$$\begin{aligned} \lambda_a &= \frac{\lambda \times T_a}{T_a + T_s} \\ &= \frac{\lambda}{1 + T_s/T_a} \leq \lambda \end{aligned} \tag{15.5}$$

*Example 15.1.* Assume a 64 kbit/s voice source which is modeled as an on–off source with an average duration of the active period of $T_a = 0.45$ s and average duration of the silent period is $T_s = 1.5$ s. Estimate the source parameters and the average data rate.

From (15.2), the probability the source remains in active state is

$$a = \frac{T_a}{1 + T_a} = 0.3103$$

From (15.4) the probability the source remains silent is

$$s = \frac{T_s}{1 + T_s} = 0.6$$

The average data rate is

$$\lambda_a = 14.77 \quad \text{kbps}$$

■

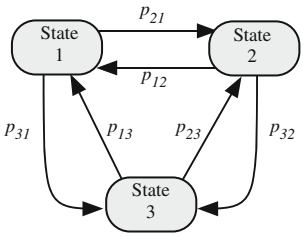



**Fig. 15.3** A three-state MMPP source model

### *15.2.3 Markov Modulated Poisson Process*

The on–off traffic source model does not describe too well the effect of multiplexing several data sources. There is only one rate when the source is active while actual sources display differing data rates when they are active. To handle this situation, more states are added to the MMPP. Figure 15.3 shows a three-state configuration which is naturally called three-state MMPP.

For an MMPP with $N$ states, we construct an $N \times N$ state transition matrix $\mathbf{P}$ whose element $p_{ij}$ represents the probability of making a transition from state $j$ to state $i$. This choice is consistent with our definition for the transition matrix of a Markov chain. Needles to say this matrix is a column stochastic matrix.

When the source is in state $i$, $1 \leq i \leq N$, the packets are produced at a rate $\lambda_i$. When the number of states is $N = 2$, we have a *switched Poisson process* (SPP) [21]. When $N = 2$ and $\lambda_1 = 0$, we have an *interrupted Poisson process* (IPP) which is also the on–off model that was discussed above.

### *15.2.4 Autoregressive Models*

Autoregressive models produce traffic with short-range dependence where the autocorrelation function decays exponentially. An autoregressive model of order $N$, denoted AR($N$), is described by

$$X(n) = \sum_{k=1}^{N} a_i \, X(n-k) + \epsilon(n) \tag{15.6}$$

where $X(n)$ is a random variable indicating the traffic rate at that time; and $\epsilon(n)$ is a random variable having small range to fit experimental data. The above formula gives a simple method for generating the next random number given the previous set of $N$ random numbers which is computationally appealing.

Alternative forms of the above expression using moving average (MA) and autoregressive moving average (ARMA) expressions were also proposed [22].

## 15.3 Continuous-Time Modeling: Poisson Traffic Description

Poisson traffic description is a model often used by many researchers due to the simplicity of the model. A characteristic of traffic is the lull period in which no packets arrive. We can think of the interarrival time between two successive packets as a random variable $T$. This r.v. is continuous for Poisson traffic. The attractive feature of Poisson traffic is that the sum of several independent Poisson processes results in a new Poisson process whose rate is the sum of the component rates [8]. Poisson traffic accurately describes user-initiated TELNET and FTP connections [4]. To study the random variable $T$ we need to study the probability $p(0)$ that no packets arrive in the period $t$. Let us start by assuming Poisson traffic with probability $p(k)$ that $k$ packets arrive in a time period $t$ which is given by

$$p(k) = \frac{(\lambda_a\, t)^k}{k!} e^{-\lambda_a\, t} \tag{15.7}$$

where $\lambda_a$ (packets/s) is the average packet arrival rate. Note that this expression for probability is valid for all values of $0 \leq k < \infty$. Of course we do not expect an infinite number of packets to arrive in a time interval $t$, but this is the expression and that is what it predicts. Note that Poisson distribution really talks about numbers. It specifies the probability of getting a number of packets $k$ in a given time period $t$.

For the interarrival time, we ask a different sort of question: what is the probability that the time separation between adjacent packets is $t$? To derive an expression for the pdf distribution for the interarrival time, we need to find the probability that no packets arrive in period $t$. Using (15.7), the probability that no packets arrive in a time period $t$ is obtained by substituting $k = 0$ in the above equation

$$p(0) = e^{-\lambda_a\, t} \tag{15.8}$$

This probability is equivalent to the event $A : T > t$ and we can write

$$\begin{aligned} p(A : T > t) &= p(0) \\ &= e^{-\lambda_a\, t} \end{aligned} \tag{15.9}$$

The event $A$ is basically the event that no packets arrived for a time period $t$. What happens after this time period is not specified. A packet might arrive or no packets arrive.

In order to find the pdf associated with the interarrival time we need to define event $B$ which is complementary to $A$ as follows:

$$B : T \leq t$$

and the probability associated with event $B$ is

$$\begin{aligned} p(B : T \leq t) &= 1 - p(A) \\ &= 1 - e^{-\lambda_a t} \end{aligned} \tag{15.10}$$

The CDF for the random variable $T$ is given from (15.10) by

$$F_T(t) = p(T \leq t) = 1 - e^{-\lambda_a t} \tag{15.11}$$

The pdf for this random variable is obtained by differentiating the above equation

$$f_T(t) = \lambda_a e^{-\lambda_a t} \tag{15.12}$$

Thus the pdf for the interarrival time of Poisson traffic follows the *exponential distribution* that was discussed in Chap. 1.

*Example 15.2.* Find the average value for the exponentially distributed interarrival time having the distribution in (15.12)

The average time between arriving packets $T_a$ is given by

$$\begin{aligned} T_a &= \int_{t=0}^{\infty} t \ \lambda_a e^{-\lambda_a t} \, dt \\ &= \frac{1}{\lambda_a} \quad \text{s} \end{aligned}$$

We see that as the rate of packet arrival decreases ($\lambda_a \ll 1$), the average time between packets increases as expected. ■

*Example 15.3.* Consider an ATM channel where a source transmits data with an average data rate of 500 kbps. Derive the corresponding Poisson distribution and find the probability that 10 cells arrive in a period of 1 ms.

Since we are talking about cells, we have to convert all the data rates from bits/s quantities into cells/s using the information we have about average packet length $A$. We start by calculating the average arrival rate which is easily done since we know the size of an ATM cell.

$$\lambda_a = \frac{500 \times 10^3}{8 \times 53} = 1.179\,2 \times 10^3 \quad \text{cells/s}$$

The probability of 10 cells arriving in the time period $t$ according to the Poisson distribution is found using (15.7):

$$p(10) = \frac{(\lambda_a t)^{10}}{(10)!} e^{-\lambda_a t} = 4.407 \times 10^{-7}$$ ■

### 15.3.1 Memoryless Property of Poisson Traffic

The memoryless property of Poisson traffic is defined using the following conditional probability expression related to the interarrival time:

$$p(T > t + \epsilon | T > t) = p(T > \epsilon) \quad \text{for all} \quad t, \epsilon > 0 \tag{15.13}$$

Basically this equation states that the probability that no packets arrive for a time $t + \epsilon$ given that no packets arrived up to time $t$ does not depend on the value of $t$. It depends only on $\epsilon$. So in effect the expression states that we knew that we waited for $t$ seconds and no packets arrived. Now we reset our clock and we ask the question: What is the probability that a packet arrives if we wait for a period $\epsilon$ seconds? The probability of this event only depends on our choice of $\epsilon$ value and will not use our prior knowledge of the period $t$.

Let us state this property using two examples of systems having the memoryless property. Assume that we are studying the interarrival times of busses instead of packets. Assume also that the time between bus arrivals is a random variable with memoryless property. We arrive at the bus station at 9:00 a.m and wait for one hour yet no buses show up. Now we know that no buses showed up for the past hour and we naturally ask the question: What are the odds that a bus will show up if we wait for five more minutes. The probability that no buses will come in the next 5 min will depend only on the wait period (5 min) and not on how long we have been waiting at the bust stop.

Another example of memoryless property is the case of an appliance (a television set for example). If the time between failures is a random variable with memoryless property, then the probability that the TV will fail after 1 hour of use is the same at any time independent of when we bought the TV or how long the TV has been used.

Obviously the time between failures in cars and airplanes has a memory property. That is why an older car breaks down more often compared to a new car or compared to an older car that is only driven on weekends in the summer months only.

Let us turn back to our interarrival time statistics. From (15.9) we could write

$$p(T > t) = e^{-\lambda_a \, t} \tag{15.14}$$

Changing the time value from $t$ to $t + \epsilon$, we get

$$p(T > t + \epsilon) = e^{-\lambda_a \, (t+\epsilon)} \tag{15.15}$$

Equation (15.13) is a conditional probability and we can write it as

$$p(A|B) = \frac{p\,(A \bigcap B)}{p(B)} \tag{15.16}$$

where the events $A$ and $B$ are defined as

$$A : T > t + \epsilon \tag{15.17}$$

$$B : T > t \tag{15.18}$$

But $A \bigcap B = A$ since $\epsilon > 0$ implies that if event $A$ took place, then event $B$ has taken place also. Thus we have

$$p(T > t + \epsilon | T > t) = \frac{p(A)}{p(B)} \tag{15.19}$$

$$= \frac{e^{-\lambda_a\ (t+\epsilon)}}{e^{-\lambda_a\ t}} \tag{15.20}$$

$$= e^{-\lambda_a \epsilon} \tag{15.21}$$

Thus we have proven that the interarrival time for the exponential distribution is memoryless.

### *15.3.2 Realistic Models for Poisson Traffic*

The Poisson distribution and the interarrival time considered in Sect. 15.3 do not offer much freedom in describing realistic traffic sources since they contain one parameter only: $\lambda$ (packets/s) that reflected the average data arrival rate. The minimum value for the interarrival time is zero. This implies that the time interval between two packet headers could be zero. An interarrival time value of zero implies two things: that our packets have zero length and the data rate could be infinity. Both of these conclusions are not realistic.

A realistic bursty source is typically described using some or all of these parameters:

| | |
|---|---|
| $\lambda_a$ | the average data rate |
| $\sigma$ | the maximum data rate expected from the source |

Since we are talking about rates in terms of packets/s, we need to make sure that the rates are in terms of packet/s. The source parameters above could be elaborated upon further, depending on our need. Section 15.2.1 discusses source with multiple data rates.

Now we ask the question: How can we write down an expression for a Poisson distribution that takes into account all of the source parameters? We have two options:

*Flow Description* This option allows us to specify the randomness of the instantaneous data rate produced by the source.

*Interarrival Time Description* This option allows us to specify the randomness of the periods between adjacent packets produced by the source.

### 15.3.3 Flow Description

We start by writing the pdf for the instantaneous data rate in the form

$$f_\Lambda(\lambda) = b\, e^{-b\lambda} \tag{15.22}$$

where $\lambda$ is the data rate and the parameter $b$ is the *shape* parameter that determines the steepness of the exponential curve.

Figure 15.4 shows the distribution where $\lambda_a$ in the figure indicates the average data rate. The distribution in (15.22) is a valid pdf since its integral equals unity.

To find the parameter $b$, we need to estimate the average data rate $\lambda_a$. The average data rate for the distribution given in (15.22) is

$$\lambda_a = \int_0^\infty \lambda\, b\, e^{-b\,\lambda}\, d\lambda = \frac{1}{b} \tag{15.23}$$

Based on this equation, we can determine the pdf for the data rate produced by the source given its average rate. If $\lambda_a$ is the average data rate of a source, then the pdf for its rate is given by

$$f_\Lambda(\lambda) = \frac{1}{\lambda_a}\, e^{-\lambda/\lambda_a} \tag{15.24}$$

Thus to describe the data rate of a source that follows the Poisson distribution, we need to specify its average data rate $\lambda_a$ only.

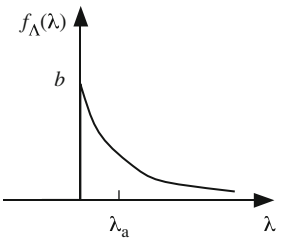



**Fig. 15.4** Exponential distribution describing instantaneous rate of a Poisson source

### 15.3.4 Interarrival Time Description

The pdf description of the interarrival time for a Poisson source follows the exponential distribution which we repeat here for convenience.

$$f_T(t) = \lambda_a e^{-\lambda_a t} \tag{15.25}$$

We mentioned that this equation is not sufficient to describe real traffic since it contains one parameter only $\lambda_a$ which describes the average data rate only. We can modify the interarrival time distribution and obtain the *biased exponential distribution* as follows.

$$f_T(t) = \begin{cases} 0 & t < a \\ b \ \exp -b\,(t-a) & t \geq a \end{cases} \tag{15.26}$$

where $a \geq 0$ is the *position parameter* (units s) and $b > 0$ is the *shape parameter* (units $s^{-1}$). Basically $a$ represents the minimum time between adjacent packets and $b$ determines how fast the exponential function decays with time. Both $a$ and $b$ will determine the average packet rate as will be explained in Example 15.4 below.

Figure 15.5 shows the distribution given by the expression in (15.26). We see from the figure that $a$ places the pdf at the desired position on the time axis and $b$ determines how fast the exponential function decays with time. The distribution in (15.26) is a valid pdf since its integral equals unity. The next section explains how to obtain the correct values for $a$ and $b$ for a typical source.

*Example 15.4.* Find the average value for the exponentially distributed interarrival time with pdf given by (15.26)

The average time separation between arriving packets $T_a$ is given by

$$\begin{aligned} T_a &= \int_{t=a}^{\infty} t\ be^{-b(t-a)}\ dt \\ &= a + \frac{1}{b} \qquad \text{s} \end{aligned}$$

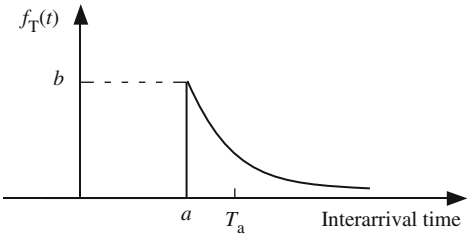



**Fig. 15.5** A biased exponential distribution with two design parameters: position parameter $a$ and shape parameter $b$

The average interarrival time $T_a$ depends on both $a$ and $b$ parameters. We see that as the shape parameter decreases ($b \ll 1$), the average time between packets increases.

On the other hand, when $b$ is large ($b \gg 1$), the exponential function will approach a delta function and the interarrival time will have its *minimum value* $T_a \approx a$.

The variance of the interarrival time for the shifted exponential distribution is given by

$$\sigma^2 = \frac{1}{b^2}$$

which depends only on the shape parameter. So large values for $b$ will result in traffic with low burstiness approaching CBR. Lower values for $b$ result in more bursty traffic. ■

### 15.3.5 Extracting Poisson Traffic Parameters

In this section we show how to find the values of the position parameter $a$ and shape parameter $b$ for a source whose average rate $\lambda_a$ and burst rate $\sigma$ are known.

The position parameter $a$ is equivalent to the minimum time between two adjacent packets. In a time period $t$, the *maximum number* of packets that could be produced by the source is given by

$$N_m = \sigma\, t \tag{15.27}$$

where it was assumed that $\sigma$ was given in units of packets/s. The minimum time between two adjacent packets was defined as $a$ and is given by

$$\alpha = \frac{t}{N_m} = \frac{1}{\sigma} \quad \text{s} \tag{15.28}$$

Problem 15.5 discusses obtaining the parameter $a$ when $\sigma$ is expressed in units of bits/s.

In a time period $t$, the *average number* of packets that could be produced by the source is given by

$$N_a = \lambda_a\, t \tag{15.29}$$

The average time between two adjacent packets is given by

$$T_a = \frac{t}{N_a} = \frac{1}{\lambda_a} \quad \text{s} \tag{15.30}$$

But from Example 15.4 we obtained an expression for the average interarrival time as

$$T_a = a + \frac{1}{b} \quad \text{s} \tag{15.31}$$

From the above two equations we are able to obtain a value for the shape parameter $b$

$$\frac{1}{\lambda_a} = a + \frac{1}{b} \tag{15.32}$$

Therefore we have

$$b = \frac{\sigma\ \lambda_a}{\sigma - \lambda_a} \quad \text{s}^{-1} \tag{15.33}$$

Problem 15.6 discusses obtaining the parameter $a$ when $\sigma$ and $\lambda_a$ are expressed in units of bits/s.

*Example 15.5.* A data source follows the Poisson distribution and has an average data rate $\lambda_a = 10^3$ packets/s and maximum burst rate of $\sigma = 3 \times 10^3$ packets/s. Estimate the exponential distribution parameters that best describe that source.

The position parameter is given from (15.28) by

$$a = \frac{1}{3 \times 10^6} = 3.33 \times 10^{-4} \quad \text{s}$$

The shape parameter $b$ is given from (15.33) by

$$b = 1500 \quad \text{s}^{-1}$$

The pdf for the interarrival time is given by

$$f_T(t) = 1500\ \exp -1000\left(t - 3.3 \times 10^{-4}\right)$$ ■

### 15.3.6 *Poisson Traffic and Queuing Analysis*

The previous subsection discussed how the biased exponential distribution parameters can be extracted given the system parameters:

$\lambda_a$ the average data rate
$\sigma$ the maximum data rate expected from the source

A Poisson source matching these given parameters has position parameter given by (15.28) and shape parameter given by (15.33). In this section we ask the question: Given a Poisson source with known parameters that feeds into a queue what is the packet arrival probability for the queue? Remember that in queuing theory the two most important descriptors are the arrival statistics and the departure statistics.

There are two cases that must be studied separately based on the values of the step size $T$ and the position parameter $a$.

## *Case When $T \leq a$*

The case $T \leq a$ implies that we are sampling our queue at a very high rate that is greater than the burst rate of the source. Therefore, when $T \leq a$ at most one packet could arrive in one time step with probability $x$ that we have to determine. We use the symbol $x$ for arrival probability since the symbols $\alpha$ is used here to describe the position parameter.

The number of time steps over a time period $t$ is estimated as

$$n = \frac{t}{T} \tag{15.34}$$

The average number of packets arriving over a period $t$ is given by

$$\begin{aligned} N_a &= \lambda_a \, t \\ &= \lambda_a \, n \, T \end{aligned} \tag{15.35}$$

where $\lambda_a$ was assumed to be given in units of packets/s.

From the binomial distribution, the average number of packets in one step time is

$$N_a = x \, n \tag{15.36}$$

where $x$ is the packet arrival probability in one time step.

From the above two equations, the packet arrival probability per time step is given by

$$x = \lambda_a \, T \tag{15.37}$$

We see in the above equation that as $T$ gets smaller or as the source activity is reduced (small $\lambda_a$), the arrival probability is decreased which makes sense.

*Example 15.6.* Estimate the packet arrival probability for a source with the following parameters. $\lambda_a = 50$ packets/s and $\sigma = 150$ packets/s. Assume the time step value is $T = 1$ ms.

The position parameter is

$$\alpha = \frac{1}{\sigma} = 6.7 \quad \text{ms}$$

The shape parameter is

$$\beta = \frac{\lambda_a \sigma}{\sigma - \lambda_a} = 75 \quad \text{s}^{-1}$$

The packet arrival probability is

$$x = 0.05$$ ■

### *Case When T > a*

The case $T > a$ implies that we are sampling our queue at a rate that is slower than the burst rate of the source. Therefore, when $T > a$ more than one packet could arrive in one time step and we have to find the packet arrival statistics that describe this situation.

We start our estimation of the arrival probability $x$ by determining the maximum number of packets that could arrive in one step time

$$N_m = \lceil \sigma \ T \rceil \tag{15.38}$$

The *ceiling* function was used here after assuming that the receiver will consider packets that partly arrived during one time step. If the receiver does not wait for partially arrived packets, then the *floor* function should be used.

The average number of packets arriving in the time period $T$ is

$$N_a = \lambda_a \ T \tag{15.39}$$

From the binomial distribution, the average number of packets in one step time is

$$N_a = x \ N_m \tag{15.40}$$

From the above two equations, the packet arrival probability per time step is

$$x = \frac{\lambda_a \ T}{N_m} \leq \frac{\lambda_a}{\sigma} \tag{15.41}$$

The probability that $k$ packets arrive at one time step $T$ is given by the binomial distribution

$$p(k) = \binom{N_m}{k} x^k \, (1-x)^{N_m - k} \tag{15.42}$$

*Example 15.7.* A data source follows the Poisson distribution and has an average data rate $\lambda_a = 10^3$ packets/s and maximum burst rate of $\sigma = 5 \times 10^3$ packets/s. Find the biased Poisson parameters that describe this source and find the packet arrival probabilities if the time step is chosen equal to $T = 1$ ms.

The biased Poisson parameters are

$$\begin{aligned} a &= 2 \times 10^{-4} && \text{ms} \\ b &= 1.250 \times 10^3 && \text{s}^{-1} \end{aligned}$$

The maximum number of packets that could arrive in one time step is

$$N_m = 5$$

The packet arrival probability per time step is

$$x = 0.2$$

The probability that $k$ packets arrive per time step is

$$\begin{aligned} p(0) &= 3.2768 \times 10^{-4} \\ p(1) &= 4.0960 \times 10^{-1} \\ p(2) &= 2.0480 \times 10^{-1} \\ p(3) &= 5.1200 \times 10^{-2} \\ p(4) &= 6.4000 \times 10^{-3} \\ p(5) &= 3.2000 \times 10^{-4} \end{aligned}$$

■

## 15.4 Discrete-Time Modeling: Interarrival Time for Bernoulli Traffic

Poisson traffic description applies when time is treated as continuous. Bernoulli traffic is analogous to Poisson traffic when time is discrete. Discrete-time traffic is typically described by *Bernoulli trials* that give rise to a *binomial process* in which the probability that a packet arrives at a given time step is $x$ and the probability that a packet does not arrive is $y = 1 - x$. We use the symbol $x$ for arrival probability since the symbol $a$ is used here to describe the position parameter.

We can think of Bernoulli traffic with binomial packet arrival distribution as the discrete version of Poisson traffic with exponential packet interarrival time distribution. The latter distribution could be termed a fluid flow traffic model since it deals with *flow rate* as opposed to count the number of packets that arrive in a certain time period.

The probability that $k$ packets arrive in $n$ time steps is given by the binomial distribution

$$p(k) = \binom{n}{k} x^k \; y^{n-k} \tag{15.43}$$

We can think of the interarrival time $n$ between two successive packets as a random variable $N$. This r.v. is discrete for Bernoulli traffic.

The probability $p(0)$ that no packets arrive for *at least* $n$ consecutive time steps is given by

$$p(0) = y^n$$

The above equation simply states that we did not get any packets in $n$ time steps. What happens during time step $n + 1$ is not specified in the above equation. We might get a packet or we might not. Notice that for large $n$ the probability that no packets arrive diminishes which makes sense.

The above probability is equivalent to the event $A : N > n$ and we can write

$$\begin{aligned} p(A : N > n) &= p(0) \\ &= (1 - x)^n \end{aligned} \tag{15.44}$$

This equation will help us in the next section to prove the memoryless property of Bernoulli traffic. However, we proceed here to find the pmf associated with the interarrival time.

In order to find the pmf associated with the interarrival time we need to find the probability that the interarrival time *exactly equals* $n$ time steps. In other words, our event now specifies that no packets arrived for $n$ time steps followed by a packet arrival event at the next step when the time index is $n + 1$. The desired probability is given by

$$p = x \; (1 - x)^n \tag{15.45}$$

This probability is equal to the pmf of the interarrival time and we can write the pmf as

$$p(N = n) = x \; (1 - x)^n \tag{15.46}$$

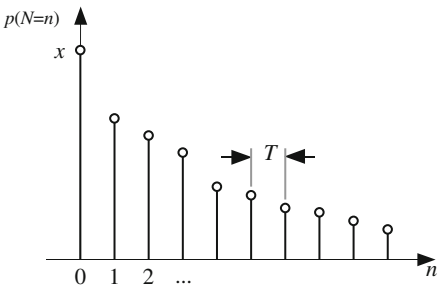



**Fig. 15.6** A simple geometric distribution with one parameter $x$, the probability that a packet arrives in a given time step

Figure 15.6 illustrates the pmf for the interarrival time of Bernoulli traffic has geometric distribution. A simple geometric distribution with one parameter $x$, the probability that a packet arrives in a given time step.

*Example 15.8.* Find the average value for the interarrival time of Bernoulli traffic which is described by (15.46).

The average number of time steps between packets $n_a$ is given by

$$n_a = \sum_{n=0}^{\infty} n\, x\, y^n$$
$$= \frac{y}{1-y} = \frac{y}{x}$$

We see that as the probability for packet arrival decreases ($x \ll 1$), the average number of steps between packets increases as expected. On the other hand, when $x$ approaches unity, the interarrival time becomes $n_a \approx 0$ as expected. This indicates that arriving packets have no empty time slots in between them. ■

*Example 15.9.* Consider an ATM channel where cells arrive with an average data rate of $\lambda_a = 2.3 \times 10^4$ cells/s and a burst rate limited only by the line rate is $\sigma = \lambda_l = 155.52$ Mbps. Derive the equivalent binomial distribution parameters and find the probability that 10 cells arrive in one sample time period of $T = 1$ ms.

The average number of ATM cells received in one time step period is given by

$$N_a = 2.3 \times 10^4 \times 10^{-3} = 23$$

which represents the average traffic produced by the source.

The maximum number of ATM cells that could be received in this time is found by estimating the duration of one ATM cell as determined by the line rate.

$$\Delta = \frac{8 \times 53}{155.52 \times 10^6} = 2.7263 \quad \mu\text{s}$$

The maximum number of ATM cells that could be received in 1 ms period is given by

$$N_m = \lceil \frac{T}{\Delta} \rceil = 367$$

The cell arrival probability per time step is given from the equation

$$x\ N_m = N_a$$

which gives

$$x = \frac{23}{367} = 0.0627$$

and of course $y = 1 - x = 0.9373$.

The probability of 10 cells arriving out of potential $N_m$ cells is

$$p(10) = \binom{N_m}{10} x^{10}\ y^{N_m - 10} = 9.3192 \times 10^{-4}$$

We note that $p(10)$ as obtained here is almost equal to $p(10)$ as obtained using the Poisson distribution in Example 15.4. ■

### 15.4.1 Memoryless Property of Bernoulli Traffic

Assume $n$ is the number of time steps between arriving packets. It is obvious that the value of $n$ shows random variations from one packet to another. Define $N$ as the random variable associated with $n$. The memoryless property of the interarrival time for Bernoulli traffic is defined using the following conditional probability expression for discrete random variables

$$p(N > n + m | N > n) = p(N > m) \quad \text{for all} \quad n, m > 0 \tag{15.47}$$

Basically this equation states that the probability that a packet arrives after a time $n + m$ given that no packets arrived up to time $n$ does not depend on the value of $n$. It depends only on $m$.

From (15.44) we could write

$$p(n) = (1 - x)^n \tag{15.48}$$

where $x$ is the packet arrival probability during on step time. Changing the time value from $n$ to $n + m$, we get

$$p(n + m) = (1 - x)^{n+m} \tag{15.49}$$

Equation (15.47) is a conditional probability and we can write it as

$$p(A|B) = \frac{p\,(A \bigcap B)}{p(B)} \tag{15.50}$$

where the events $A$ and $B$ are defined as

$$A : N > n + m \tag{15.51}$$

$$B : N > n \tag{15.52}$$

But $A \bigcap B = A$ since $m > 0$ which implies that if event $A$ took place, then event $B$ has taken place also. Thus we have

$$p(N > n + m|N > n) = \frac{p(A)}{p(B)} \tag{15.53}$$

$$= \frac{(1 - x)^{n+m}}{(1 - x)^n} \tag{15.54}$$

$$= (1 - x)^m \tag{15.55}$$

$$= p(N > m) \tag{15.56}$$

Thus we have proven that the interarrival time for Bernoulli traffic is memoryless.

### *15.4.2 Realistic Model for Bernoulli Traffic*

The interarrival time considered in Sect. 15.4 does not offer much freedom in describing realistic traffic sources since it contains one parameter only: $x$ the probability that a packet arrives at a certain time step. As was mentioned before, a realistic source is typically described using more parameters than just the average data rate.

| | |
|---|---|
| $\lambda_a$ | the average data rate |
| $\sigma$ | the maximum data rate expected from the source |

The question becomes, how can we write down an expression similar to the one given in (15.46) that takes into account all of these parameters? We can follow a similar approach as we did for Poisson traffic in Sect. 15.3.2 when we

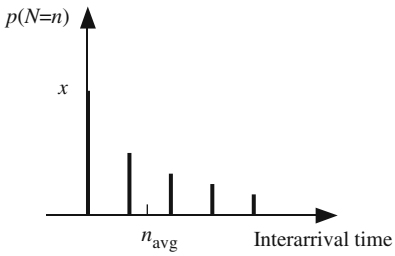



**Fig. 15.7** A biased geometric distribution with two design parameters: position parameter $\alpha$ and packet arrival probability $x$

modified the exponential distribution by shifting it by the position parameter $\alpha$. We modify (15.46) to obtain the *biased geometric distribution* as follows.

$$p(N = n) = \begin{cases} 0 & n < \alpha \\ x\,(1-x)^{n-\alpha} & n \geq \alpha \end{cases} \tag{15.57}$$

where $n$ is the number of time steps between packet arrivals, $\alpha \geq 0$ in is called the *position parameter*, in units of time steps, and $x$ is the probability that a packet arrived during one time step. Basically $a$ represents the minimum number of time steps between adjacent packets.

Figure 15.7 shows the discrete exponential distribution. We see from the figure that $\alpha$ places the pmf at the desired position on the time axis and $x$ determines how fast the exponential function decays with time. The distribution in (15.57) is a valid pmf since its sum equals unity. The values of $\alpha$ and $x$ will be derived in the next section.

*Example 15.10.* Find the average value for the geometrically distributed interarrival time given by (15.57).

The average number of time steps between packets $n_a$ is given by

$$\begin{aligned} n_a &= \sum_{n=\alpha}^{\infty} n\,x\,(1-x)^{n-\alpha} \\ &= \frac{y}{x} + \alpha \end{aligned}$$

We see that as the probability for packet arrival decreases ($x \ll 1$), the average number of steps between packets increases as expected. A high value for $\alpha$ implies slow traffic since the minimum separation between packets is large. This results in increased values for $n_a$.

The expression we obtained for the average interarrival time reduces to the expression obtained in Example 15.8 for the simple exponential distribution when the position parameter $\alpha \to 0$. ■

### 15.4.3 Extracting Bernoulli Traffic Parameters

In this section we show how to find the values of the position parameter $a$ and packet arrival probability $x$ for a source whose average rate $\lambda_a$ and burst rate $\sigma$ are known.

The position parameter $\alpha$ in (15.57) is found by studying packet arrival during a period $t$. The time step $T$ associated with this discrete arrival process is arbitrary and depends on the specifics of the system being studied.

If we study our system for a time period $t$, then the number of time steps $n$ spanned by this time period is given by

$$n = \frac{t}{T} \tag{15.58}$$

The maximum number of packets $N_m$ that could arrive during this time period depends on the burst rate $\sigma$

$$N_m = \sigma \times t \tag{15.59}$$

where $\sigma$ was assumed to be given in units of packets/s.

The minimum number of time steps between adjacent packets when the source is transmitting at the burst rate is given by

$$\alpha = \lfloor \frac{n}{N_m} \rfloor = \lfloor \frac{1}{\sigma\ T} \rfloor \tag{15.60}$$

where the floor function was used to find a conservative estimate of $\alpha$. Now we have an expression for the position parameter $a$ for the biased exponential distribution that depends on the source burst rate and the time step value.

Now we turn our attention to estimating the arrival probability $x$. Consider a period of time $t$ again which corresponds to $n$ time steps. Because we have Bernoulli traffic, we can estimate $x$ from the average number of packets received in that time period $t$ which corresponds to $n$ time steps:

$$N_a = x \times n \tag{15.61}$$

The average number of packets can also be estimated from the average data rate $\lambda_a$.

$$N_a = \lambda_a \times t \tag{15.62}$$

We can find a value of $x$ from the above two equations as

$$x = \frac{\lambda_a \times t}{n} = \lambda_a\ T \tag{15.63}$$

*Example 15.11.* An ATM data source follows the binomial distribution and has an average data rate $\lambda_a = 400$ packets/s and maximum burst rate of $\sigma = 10^3$ packets/s. Estimate the geometric distribution parameters that best describe that source if the step time value $T$ is chosen equal to 0.1 ms.

The position parameter is given by

$$\alpha = 10 \quad \text{time steps}$$

and the packet arrival probability per time step is given by

$$x = 0.04$$

Thus the pmf describing the interarrival time is given by

$$p(N = n) = 0.04 \times 0.96^{n-10}$$ ■

### 15.4.4 Bernoulli Traffic and Queuing Analysis

The previous subsection discussed how the biased geometric distribution parameters can be extracted given the system parameters:

| | |
|---|---|
| $\lambda_a$ | the average data rate |
| $\sigma$ | the maximum data rate expected from the source |

A Bernoulli source matching these given parameters has position and shape parameters:

$$a = 1/(\sigma\ T) \tag{15.64}$$

$$x = \lambda_a\ T \tag{15.65}$$

In this section we ask the question: Given a Bernoulli source with known parameters that feeds into a queue what is the packet arrival probability for the queue? Remember that in queuing theory the two most important descriptors are the arrival statistics and the departure statistics. The analysis we undertake is very similar to the one done for Poisson traffic in Sect. 15.4.

There are two cases that must be studied separately based on the values of the step size $T$ and the position parameter $a$.

### *Case When $T \leq a$*

The case $T \leq a$ implies that we are sampling our queue at a very high rate that is greater than the burst rate of the source. Therefore, when $T \leq a$ at most one packet could arrive in one time step with probability $x$ that we have to determine.

The number of time steps over a time period $t$ is estimated as

$$n = \frac{t}{T} \tag{15.66}$$

The average number of packets arriving over a period $t$ is given by

$$\begin{aligned} N_a &= \lambda_a \, t \\ &= \lambda_a \, n \, T \end{aligned} \tag{15.67}$$

From the binomial distribution, the average number of packets in one step time is

$$N_a = x \, n \tag{15.68}$$

where $x$ is the packet arrival probability in one time step.

From the above two equations, the packet arrival probability per time step is given by

$$x = \lambda_a \, T \tag{15.69}$$

We see in the above equation that as $T$ gets smaller or as the source activity is reduced (small $\lambda_a$), the arrival probability is decreased which makes sense.

### *Case When $T > a$*

The case $T > a$ implies that we are sampling our queue at a rate that is slower than the burst rate of the source. Therefore, when $T > a$ more than one packet could arrive in one time step and we have to find the packet arrival statistics that describe this situation.

We start our estimation of the arrival probability $x$ by determining the maximum number of packets that could arrive in one step time

$$N_m = \lceil \sigma \, T \rceil \tag{15.70}$$

The *ceiling* function was used here after assuming that the receiver will consider packets that partly arrived during one time step. If the receiver does not wait for partially arrived packets, then the *floor* function should be used.

The average number of packets arriving in the time period $T$ is

$$N_a(in) = \lambda_a \; T \tag{15.71}$$

From the binomial distribution, the average number of packets in one step time is

$$N_a(in) = x \; N_m \tag{15.72}$$

From the above two equations, the packet arrival probability per time step is

$$x = \frac{\lambda_a \; T}{N_m} \tag{15.73}$$

The probability that $k$ packets arrive at one time step $T$ is given by the binomial distribution

$$p(k) = \binom{N_m}{k} x^k \; (1 - x)^{N_m - k} \tag{15.74}$$

## 15.5 Self-Similar Traffic

We are familiar with the concept of periodic waveforms. A periodic signal repeats itself with *additive* translations of time. For example, the sine wave $\sin \omega t$ will have the same value if we add an integer multiple of the period $T = 2\pi/\omega$ since

$$\sin \omega t \; = \sin \omega (t + i \; T)$$

On the other hand, a self-similar signal repeats itself with *multiplicative* changes in the time scale [8–18]. Thus a self-similar waveform will have the same shape if we scale the time axis up or down. In other words, imagine we observe a certain waveform on a scope when the scope is set at 1 ms/division. We increase the resolution and set the scale to 1 $\mu$s/division. If the incoming signal is self-similar, the scope would display the same waveform we saw earlier at a coarser scale.

Self-similar traffic describes traffic on Ethernet LANs and variable-bit-rate video services [2–7]. These results were based on analysis of millions of observed packets over an Ethernet LAN and an analysis of millions of observed frame data generated by VBR video. The main characteristic of self-similar traffic is the presence of "similarly-looking" bursts at every time scale (seconds, minutes, hours) [8].

The effect of self-similarity is to introduce long range (large lag) autocorrelation into the traffic stream which is observed in practice. This phenomenon leads to periods of high traffic volumes even when the average traffic intensity is low. A switch or router accepting self-similar traffic will find that its buffers will be overwhelmed at certain times even if the expected traffic rate is low. Thus

switches with buffer sizes selected based on simulations using Poisson traffic will encounter unexpected buffer overflow and packet loss. Poisson traffic models predict exponential decrease in data loss as the buffer size increases since the probability of finding the queue in a high-occupancy state decreases exponentially. Self-similar models, on the other hand, predict stretched exponential loss curves. This is why increasing link capacity is much more effective in improving performance than increasing buffer size. The rationale being that it is better to move the data along than to attempt to store them since any buffer size selected might not be enough when self-similar traffic is encountered.

## 15.6 Self-Similarity and Random Processes

Assume we have a discrete-time random process $X(n)$ that produces the set of random variables $\{X_0, X_1, \cdots\}$. We define the aggregated random process $X^m$ as a random process whose data samples are calculated as

$$\begin{aligned} X_0^{(m)} &= \frac{1}{m}\left[X_0 + X_1 + \cdots + X_{m-1}\right] \\ X_1^{(m)} &= \frac{1}{m}\left[X_m + X_{m+1} + \cdots + X_{2m-1}\right] \\ X_2^{(m)} &= \frac{1}{m}\left[X_{2m} + X_{2m+1} + \cdots + X_{3m-1}\right] \\ &\vdots \end{aligned}$$

The random process is self-similar if satisfies the following properties.

1. The processes $X$ and $X^{(m)}$ are related by the equation

$$X^{(m)} = \frac{1}{m^{(1-H)}}\, X \tag{15.75}$$

   where $H$ is the *Hurst parameter* ($0.5 < H < 1$).
2. The means of $X$ and $X^{(m)}$ are equal

$$E[X] = E[X^{(m)}] = \mu \tag{15.76}$$

3. The autocovariance functions of $X$ and $X^{(m)}$ are equal

$$E\left[(X(n+k)-\mu)\,(X(n)-\mu)\right] = E\left[\left(X(n+k)^{(m)}-\mu\right)\left(X(n)^{(m)}-\mu\right)\right] \tag{15.77}$$

A self-similar random process exhibits long-range dependence where the autocorrelation function $r_{XX}(n)$ or the autocovariance function $c_{XX}(n)$ does vanish for large values of $n$. Distributions that have long-range dependence are sometimes called *heavy-tailed distributions*. A random process that displays no long-range dependence will have the autocorrelation and autocovariance functions vanish for low values of $n$. A typical random process that has no long-range dependence is the Brownian motion.

Typically self-similar phenomena are described using the Hurst parameter $H$ whose value lies in the range

$$0.5 < H < 1 \tag{15.78}$$

The case $H = 0.5$ describes random walk problems or Brownian motion which exhibits no self-similarity. As $H \to 1$ the degree of self-similarity increases as well as the long-range dependence.

One way to model self-similar traffic is to use pdf distributions for the interarrival time that exhibits heavy-tailed distribution as explained in the following section.

## 15.7 Heavy-Tailed Distributions

A heavy-tailed distribution gives rise to traffic that shows long-range dependence like in compressed video traffic. A distribution is heavy-tailed if it exhibits the following characteristics:

1. Its variance is high or infinite.
2. Its CDF has the property

$$1 - F(x) = p\,(X > x) \sim \frac{1}{x^{\alpha}} \quad x \to \infty \tag{15.79}$$

where $0 < \alpha < 2$ is the shape parameter and $X$ is a random variable.

## 15.8 Pareto Traffic Distribution

The Pareto distribution that we studied in Sect. 1.20 on page 19 is used here to describe realistic traffic sources that have bursty behavior. The Pareto distribution is described by the pdf

$$f\,(x) = \frac{b\,a^{b}}{x^{b+1}} \tag{15.80}$$

where $a$ is the position parameter, $b$ is the shape parameter, and the random variable $X$ has values limited in the range $a \leq x < \infty$. The Pareto distribution CDF is given by

$$F(x) = 1 - \left(\frac{a}{x}\right)^b \tag{15.81}$$

Notice that the Pareto distribution satisfies condition 2 of heavy-tailed distributions defined in Sect. 15.7.

The mean and variance for $X$ are

$$\mu = \frac{b\,a}{b-1} \tag{15.82}$$

$$\sigma^2 = \frac{b\,a^2}{(b-1)^2\,(b-2)} \tag{15.83}$$

The mean is always positive as long as $b > 1$. The variance is meaningful only when $b > 2$. The variance of the Pareto distribution could be made high by properly choosing the shape parameter $b$ to be close to 1 as the above equation indicates.

The Hurst parameter corresponding to the Pareto distribution is given by the equation

$$H = \frac{3-b}{2} \tag{15.84}$$

Table 15.1 shows the relation between the source burstiness and the two parameters $H$ and Pareto distribution shape parameter $b$.

From the table we conclude that in order to describe self-similar traffic using the Pareto distribution, we must have the shape parameter $b$ close to one. Typically $H$ is chosen within the range 0.7–0.8 which corresponds to $b$ values in the range 1.4–1.6. By proper choice of $b$, we can satisfy all the conditions defining heavy-tailed distributions defined in Sect. 15.7.

From (15.81) we can write

$$P(X > x) = 1 - F(x) = \left(\frac{a}{x}\right)^b \tag{15.85}$$

**Table 15.1** Relation between the source burstiness and the two parameters $H$ and Pareto distribution shape parameter $b$

| Traffic statistics | $H$ value | $b$ value |
|---|---|---|
| Long-range dependent | $H \to 1$ | $b \to 1$ |
| Short-range dependent | $H \to 0.5$ | $b \to 2$ |

which means that the probability that the random variable has a value greater than $x$ decreases at a rate that depends on the shape parameter $b$. If $b \approx 1$, the distribution has very large mean and variance [11].

A realistic bursty source is typically described using some or all of these parameters:

| | |
|---|---|
| $\lambda_{min}$ | the minimum data rate |
| $\lambda_a$ | the average data rate |
| $\sigma$ | the maximum data rate expected from the source |

Since we are talking about rates in terms of packets/s, we need to convert these specifications into proper packet rates. The question becomes, how can we write down an expression for a Pareto distribution that takes into account all of these parameters?

We have two options:

*Flow Description* This option allows us to specify the randomness of the instantaneous data rate produced by the source.

*Interarrival Time Description* This option allows us to specify the randomness of the periods between adjacent packets produced by the source.

### 15.8.1 Flow Description

We start by writing the pdf for the instantaneous data rate in the form

$$f_\Lambda(\lambda) = \begin{cases} 0 & \text{when } \lambda < \lambda_{min} \\ b\, a^b / \lambda^{b+1} & \text{when } \lambda \geq \lambda_{min} \end{cases} \tag{15.86}$$

where $\lambda_{min}$ is the minimum data rate, which could be zero, $a$ is the *position parameter* and $b$ is the *shape parameter* that determines the steepness of the curve.

The values of the two parameters $a$ and $b$ can be found for a source with traffic descriptors $(\lambda_{min}, \lambda_a, \sigma)$ as follows.

$$a = \lambda_{min} \tag{15.87}$$

$$b = \frac{\lambda_a}{\lambda_a - \lambda_{min}} \tag{15.88}$$

To produce a bursty source, the value of $b$ could be chosen close to 1 according to the data in Table 15.1.

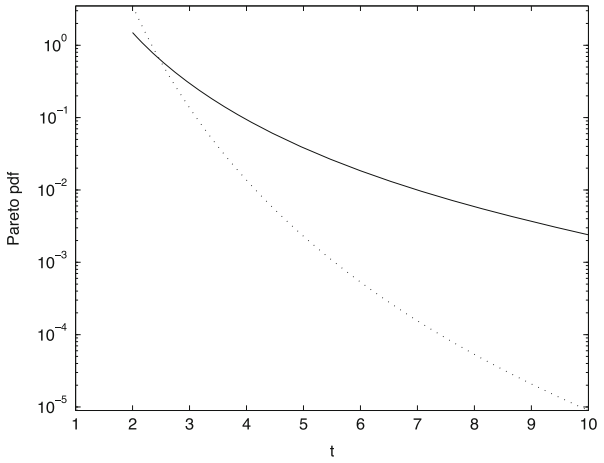



**Fig. 15.8** The pdf distribution for the case when $a = 2$ and $b = 3$ (*solid line*) and $b = 7$ (*dashed line*)

### *15.8.2 Interarrival Time Description*

The interarrival time following the Pareto distribution has a pdf that is given by

$$f_T(t) = \frac{b\, a^b}{t^{b+1}} \quad \text{with } a \leq t < \infty \tag{15.89}$$

where $a$ (units seconds) is the position parameter and $b \geq 1$ is the shape parameter. Figure 15.8 shows the pdf distribution for the case when $a = 2$ and $b = 3$ (solid line) and $b = 7$ (dashed line). For the smaller value of shape parameter $b$ the pdf becomes flatter and has higher values at larger values of $t$. This results in larger variance in the interarrival time distribution.

### *15.8.3 Extracting Pareto Interarrival Time Statistics*

A realistic source is typically described using some or all of these parameters:

| | |
|---|---|
| $\lambda_a$ | the average data rate |
| $\sigma$ | the maximum data rate expected from the source |

The question we pose here is how to find a Pareto distribution that best matches the given source parameters?

In a time period $t$, the maximum number of packets that could be produced by the source is given by

$$N_m = \sigma\ t \tag{15.90}$$

We use this estimate to calculate the minimum time between two adjacent packets as follows.

$$a = \frac{t}{N_m} = \frac{1}{\sigma} \quad \text{s} \tag{15.91}$$

The position parameter depends only on the average packet size and burst rate.

In the time period $t$, the average number of packets that could be produced by the source is given by

$$N_a = \lambda_a\ t \tag{15.92}$$

The average time between two adjacent packets is given by

$$T_a = \frac{t}{N_a} = \frac{1}{\lambda_a} \quad \text{s} \tag{15.93}$$

But from the Pareto pdf distribution, the average interarrival time is given by

$$T_a = \int_{t=a}^{\infty} t\ \frac{b\ a^b}{t^{b+1}}\ dt = \frac{b\ a}{b-1} \quad \text{s} \tag{15.94}$$

From the above two equations, we are able to obtain a value for the shape parameter $b$

$$\frac{1}{\lambda_a} = \frac{b\ a}{b-1} \tag{15.95}$$

Therefore we have

$$b = \frac{\sigma}{\sigma - \lambda_a} \tag{15.96}$$

The shape parameter depends only on the average rate $\lambda_a$ and burst rate $\sigma$. Furthermore, the shape parameter lies between the following extreme values

$$\begin{aligned} b &= 1 \quad \text{when } \sigma \gg \lambda_a \\ b &\to \infty \quad \text{when } \lambda_a \to \sigma \end{aligned}$$

The first expression applies to a fairly bursty source and the second expression applies to a CBR source where the average data rate equals the burst rate. Thus the range of the shape parameter $b$ can be expressed as

$$1 \leq b < \infty \tag{15.97}$$

Lower values of $b$ imply bursty sources and higher values of $b$ imply sources with little variations in the interarrival times since we would have a constant rate source.

*Example 15.12.* A bursty source produces data at an average rate of 5 Mbps and its maximum burst rate is 20 Mbps. Estimate the Pareto parameters that best describe that source assuming that the average packet size is 400 bits. The position parameter is

$$a = \frac{A}{\sigma} = 20 \quad \mu\text{s}$$

The average data rate is used to determine the shape parameter $b$

$$b = \frac{\sigma}{\sigma - \lambda_a} = 1.333$$ ■

### 15.8.4 Pareto Distribution and Queuing Analysis

The previous subsection discussed how the Pareto distribution parameters can be extracted given the system parameters:

$\lambda$ the source data rate
$\sigma$ the maximum data rate expected from the source

A Pareto distribution matching these given source parameters has position and shape parameters:

$$a = 1/\sigma \quad \text{s} \tag{15.98}$$

$$b = \sigma/(\sigma - \lambda_a) \tag{15.99}$$

where we assumed the rates to be given in terms of packets/s.

In this section we ask the question: Given a Pareto source with known parameters that feeds into a queue. What is the packet arrival probability? There are two cases that must be studied separately based on the values of the step size $T$ and the position parameter $a$.

#### Case When $T \leq a$

When $T \leq a$ at most one packet could arrive in one time step with probability $x$ that we have to determine.

The number of time steps over a time period $t$ is estimated as

$$n = \frac{t}{T} \tag{15.100}$$

The average number of packets in time period $t$ is given by

$$\begin{aligned} N_a &= \lambda_a \, t \\ &= \lambda_a \, n \, T \end{aligned} \tag{15.101}$$

From the binomial distribution, the average number of packets that arrive during time $t$ is given by

$$N_a = x \, n \tag{15.102}$$

where $x$ is the packet arrival probability in one time step.

From the above two equations we get

$$x = \lambda_a \, T \tag{15.103}$$

We see in the above equation that as $T$ gets smaller or as the source activity is reduced (small $\lambda_a$), the arrival probability is decreased which makes sense. The arrival probability for a Pareto distribution when $T \leq a$ is identical to the arrival probability for the Poisson distribution.

*Example 15.13.* Estimate the Pareto parameters and the packet arrival probability for a source with the following parameters. $\lambda_a = 10^3$ packets/s and $\sigma = 1.5 \times 10^4$ packets/s. Assume the time step value is $T = 0.1$ ms.

The position parameter is

$$a = 6.6667 \times 10^{-4} \quad \text{s}$$

The shape parameter is

$$b = 1.0714$$

The arrival probability is

$$x = 0.1$$

■

### *Case When T > a*

When $T > a$ more than one packet could arrive in one time step and we have to find the binomial distribution parameters that describe this situation.

We start our estimation of the arrival probability $x$ by finding the maximum number of packets that could arrive in one step time

$$N_m = \lceil \sigma \; T \rceil \tag{15.104}$$

The ceiling function was used here after assuming that the receiver will consider packets that partly arrived during one time step. If the receiver does not wait for partially arrived packets, then the floor function should be used. The average number of packets arriving in the time period $T$ is

$$N_a = \lceil \lambda_a \; T \rceil \tag{15.105}$$

From the binomial distribution, the average number of packets in one step time is

$$N_a = x \; N_m \tag{15.106}$$

From the above two equations, the packet arrival probability per time step is

$$x = \frac{N_a}{N_m} \leq \frac{\lambda_a}{\sigma} \tag{15.107}$$

The probability that $k$ packets arrive at one time step $T$ is given by the binomial distribution

$$p(k) = \binom{N_m}{k} x^k \; (1 - x)^{N_m - k} \tag{15.108}$$

*Example 15.14.* A data source follows the Pareto distribution and has an average data rate $\lambda_a = 2 \times 10^3$ packets/s and maximum burst rate of $\sigma = 5 \times 10^3$ packets/s. Find the Pareto pdf parameters that describe this source and find the packet arrival probabilities if the time step is chosen equal to $T = 2\,\text{ms}$.

The Pareto parameters are

$$a = 2 \times 10^{-4} \quad \text{ms} < T$$

The maximum number of packets that could arrive in one time step is

$$N_m = 10$$

The average number of packets that could arrive in one time step is

$$N_a = 4$$

The packet arrival probability per time step is

$$x = 0.4$$ ■

## 15.9 Traffic Data Rate Modeling with Arbitrary Source Distribution

In this section we attempt to model a traffic source that follows a general or arbitrary user-defined data rate traffic pattern. Assume that the probability mass function (pmf) of the source data rate is shown in Fig. 15.9. The number of pmf points is assumed $K$ and the time resolution is $T$. The traffic model for this source is defined by two $K$-component vectors:

$$\mathbf{v}_p = \left[ p_0 \; p_1 \cdots p_{K-1} \right]^t \tag{15.109}$$

$$\mathbf{v}_\lambda = \left[ \lambda_0 \; \lambda_1 \cdots \lambda_{K-1} \right]^t \tag{15.110}$$

where the vector $\mathbf{v}_p$ contains the pmf probabilities and the vector $\mathbf{v}_\lambda$ contains the corresponding data rate values. The peak and average data rates (packets/s) are given by

$$\sigma = \lambda_{K-1} \tag{15.111}$$

$$\lambda_a = \sum_{i=0}^{K-1} p_i \; \lambda_i \tag{15.112}$$

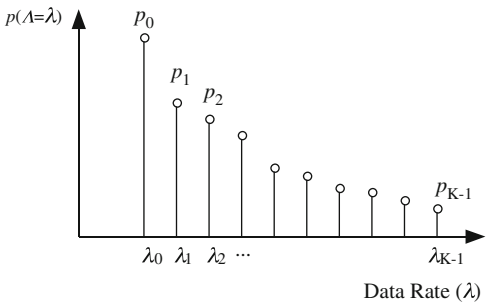



**Fig. 15.9** PMF distribution for a source with arbitrary user-specified data rate statistics

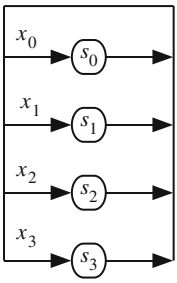



**Fig. 15.10** The state transition diagram for a traffic source that follows a particular pmf distribution

To generate traffic that obeys that general distribution we construct the source state transition diagram as shown in Fig. 15.10. State $s_i$ of the source states in Fig. 15.10 corresponds to data rate $\lambda_i$.

We need to calculate the state transition probabilities $x_i$ in the figure and see how they are related to the source probabilities $p_i$. We cannot just assume that the probabilities $x_i$ are equal to $p_i$ without some proof. From pmf definition and the figure, we can write the probability $p_i$ as

$$p_i \equiv s_i \tag{15.113}$$

The RHS of the above equation indicates that the probability that the source data rate is $\lambda_i$ given by the probability that the source state is in state $s_i$. At steady state, we can write

$$s_i = x_i \sum_{i=0}^{K-1} s_i = x_i \tag{15.114}$$

And from the above two equations we determine the state transition probabilities $x_i$ as

$$x_i = p_i \tag{15.115}$$

Although $x_i$ was proved to be equal to $p_i$, this situation will not hold true for the interarrival traffic model in Sect. 15.10.

## 15.10 Interarrival Time Traffic Modeling with Arbitrary Source Distribution

In this section we attempt to model a traffic source that follows a general or arbitrary user-defined interarrival time traffic pattern. Assume that the probability mass function (pmf) of the interarrival time is shown in Fig. 15.11. The number of

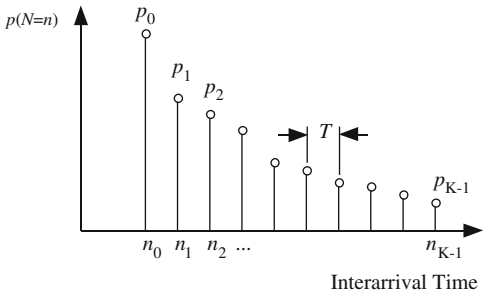



**Fig. 15.11** PMF distribution for a source with arbitrary user-specified interarrival time statistics

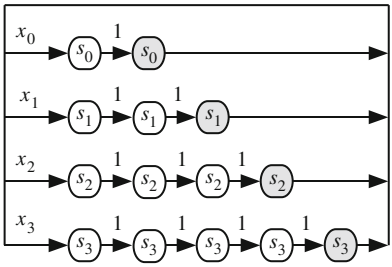



**Fig. 15.12** The state transition diagram for a traffic source that follows a particular pmf distribution

pmf points is assumed $K$ and the time resolution is $T$. The traffic model for this source is defined by two $K$-component vectors:

$$\mathbf{v}_p = \left[ p_0 \; p_1 \cdots p_{K-1} \right]^t \tag{15.116}$$

$$\mathbf{v}_n = \left[ n_0 \; n_1 \cdots n_{K-1} \right]^t \tag{15.117}$$

where the vector $\mathbf{v}_p$ contains the pmf probabilities and the vector $\mathbf{v}_n$ contains the corresponding interarrival time values. The peak and average data rates (packets/s) are given by

$$\sigma = \frac{1}{n_0 T} \tag{15.118}$$

$$\lambda_a = \frac{1}{\sum_{i=0}^{K-1} p_i \, n_i} \tag{15.119}$$

To generate traffic that obeys that general distribution we construct the source state transition diagram as shown in Fig. 15.12. We take the time step value $T$ in the Markov chain equal to the time resolution value in Fig. 15.11. Row $i$ of the source states in Fig. 15.12 corresponds to data arrival with an interarrival time value of $t_i$. The rightmost state in each row is the state where data is actually generated.

We need to calculate the state transition probabilities $x_i$ in the figure and see how they are related to the source probabilities $p_i$. From pmf definition and the figure, we can write the probability $p_i$ as

$$p_i = n_i \, s_i \tag{15.120}$$

The RHS of the above equation indicates that the probability that the interarrival time is $n_i$ given by the probability that the source state is any of the states in row $i$. Thus we determine $s_i$ as

$$s_i = \frac{p_i}{n_i} \tag{15.121}$$

At steady state, we can write

$$s_i = x_i \sum_{i=0}^{K-1} s_i \tag{15.122}$$

And from the above two equations we determine the state transition probabilities $x_i$ as

$$x_i = \frac{s_i}{\sum_{i=0}^{K-1} s_i} \tag{15.123}$$

## 15.11 Destination Statistics

Data or traffic arriving at the inputs of a switch or a router need to be routed to the desired output ports based on the information provided in the packet header and the routing table of the switch. The distribution of packets among the output ports is random and we identify three types of destination statistics as discussed in the following sections.

### *15.11.1 Uniform Traffic*

For a switch with $N$ inputs and $N$ outputs, uniform traffic implies that an incoming packet chooses a particular output port with probability $1/N$. This is true for any packet arriving at any input port. This model is referred to as the independent uniform traffic model [23]. Most studies assume uniform traffic to simplify the analysis. This assumption is true for traffic at routers or switching nodes in the network since the queuing and gradual release of the packets leads to randomization of the addressing [24]. These results apply to Ethernet LAN traffic as well as to WAN IP and ATM traffic.

### 15.11.2 Broadcast Traffic

If incoming traffic is such that an input port requests to access all output ports, we get what is called broadcast traffic. This type of traffic occurs when a site is sending data to many users at the same time or when a computer updates the databases of many computers.

Assume an $N \times N$ switch with $N$ inputs ports and $N$ output ports. We assume for simplicity that *each input port* carries two types of traffic flows: uniformly distributed traffic flow whose rate is $\lambda_u$ and broadcast traffic flow whose rate is $\lambda_b$.

The traffic flows at each input and output port are given by

$$\lambda_{\text{in}} = \lambda_u + \lambda_b \tag{15.124}$$

$$\lambda_{\text{out}} = \lambda_u + N\ \lambda_b \tag{15.125}$$

Note that each output port now has to carry more output traffic than what came in on the average because of the amplification effect of data broadcast.

The *total* traffic flows at the input and output of the switch are given by

$$f(\text{in}) = N\ \lambda_u + N\ \lambda_b \tag{15.126}$$

$$f(\text{out}) = N\ \lambda_u + N^2\ \lambda_b \tag{15.127}$$

The amount of traffic through the network increases due to data broadcast.

### 15.11.3 Hot-Spot Traffic

If incoming traffic is such that many input ports request one particular output port, we get what is called hot-spot traffic. This type of traffic occurs when a popular web site is being browsed by many users at the same time or when many computers request access to the same server. Pfister and Norton [25] models hot-spot traffic as a fixed fraction of the arrival rate or arrival probability.

Assume an $N \times N$ switch with $N$ inputs ports and $N$ output ports. We assume for simplicity that *each input port* carries two types of traffic flows: uniformly distributed traffic flow whose rate is $\lambda_u$ and hot-spot traffic flow whose rate is $\lambda_h$.

The traffic flow at each output port that is *not the destination* of the hot-spot traffic is given by

$$\lambda_r(\text{out}) = \lambda_u \tag{15.128}$$

The data rate at the output port that is *the destination* of the hot-spot traffic is begin by

$$\lambda_h(\text{out}) = \lambda_u + N\lambda_h \tag{15.129}$$

Note that hot-spot traffic effectively increases the traffic at the hot-spot port.

The overall traffic flow at the input of the switch is given by

$$f(\text{in}) = N\ \lambda_u + N\ \lambda_h \tag{15.130}$$

The overall traffic flow at the output of the switch is given by

$$f(\text{out}) = N\ \lambda_u + N\ \lambda_h \tag{15.131}$$

The amount of traffic through the network does not increase due to hot-spot traffic.

## 15.12 Packet Length Statistics

Unlike ATM, many protocols produce packets that have variable lengths. Examples of protocols that have variable length packets are IP, Frame Relay, and Ethernet. Knowledge of the packet size is essential if one wants to estimate the buffer space required to store data having a certain arrival distribution statistics.

Poisson distribution could be used to provide a simple model for packet length statistics. The probability of receiving a packet of length $A$ is given by

$$p(A) = \frac{\mu^A}{A!}e^{-\mu} \tag{15.132}$$

where $A = 1, 2, \cdots$ units of length and $\mu$ is the average packet length.

An exponential distribution could be used also in the form

$$f(A) = \frac{1}{\mu}\ e^{-A/\mu} \tag{15.133}$$

where $\mu$ is the average packet length. Alternatively, we could use the binomial distribution to describe the probability of receiving a packet of length $A$

$$p(A) = \binom{N}{A} x^A (1-x)^{N-A} \tag{15.134}$$

where $N$ is the maximum packet length and $x$ is the probability that one byte is received. We could find the value of $x$ if the average packet length $\mu$ is known:

$$x = \frac{\mu}{N} \tag{15.135}$$

If the packet length is highly irregular, then a Pareto distribution might be used in the form

$$f(A) = \frac{b\, a^b}{A^{b+1}} \tag{15.136}$$

where $a$ is minimum packet length and $b > 1$ is a shape parameter. The average packet length for this distribution is

$$\mu = \frac{b\, a}{b-1} \tag{15.137}$$

We could also use the MMPP models to describe packet length statistics as was discussed in Sect. 15.2.3. In that model, we assume a Markov chain with $N$ states such that in state $s_i$ the source produces a packet with length $A_i$. The probability of making transitions from one state to another is assumed based on experimental observations or based on model assumptions.

## 15.13 Packet Transmission Error Description

The previous sections dealt with issues related to network traffic such as data rate variation, packet length variation, and packet destination. When the packets are in transit, they are corrupted due to channel impairment or they could be totally lost due to congestion or address errors. We would like to model channel errors since these will affect the performance of the overall data transmission.

Figure 15.13 shows a model for adding errors to traffic during transmission. We have a data source that randomly generates frames as time progresses such that the interarrival time between the generated frames follows one of the distributions discussed earlier.

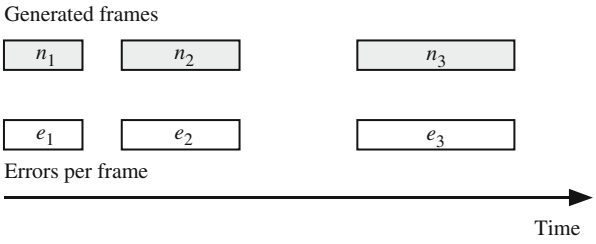



**Fig. 15.13** Time series sequence of generated data and channel errors. A received frame is in error if it is generated at the same time that an error is generated

We could even add another degree of freedom by randomly assigning different frame lengths. The number of packets per frame follows some distribution like Poisson, Bernoulli, or Pareto. The number of packets per frame is indicated in the figure by the numbers $n_1$, $n_2$, etc.

An error source also randomly generates errors with time. The number of errors per frame also follows some distribution like Poisson, Bernoulli, or Pareto. For example, a bursty error source could follow the Pareto distribution to generate lengthy error bursts. The number of packets in error per frame is indicated in the figure by the numbers $e_1$, $e_2$, etc. When the number of errors is either 0 or 1, we have a binary error source. When a Pareto distribution is used to generate the random numbers, we get bursts of errors with high probability.

*Example 15.15.* Assume an on–off data source that generates equal length frames with probability $a$ per time step. Assume for simplicity that each frame contains only one packet. The channel introduces errors in the transmitted frames such that the probability of a packet is received in error is $e$. Perform a Markov chain analysis of the system and derive its performance parameters.

The Markov chain model we use has four states:

| State | Significance |
|---|---|
| 1 | Source is idle |
| 2 | Source is retransmitting a frame that was in error |
| 3 | Frame is transmitted with no errors |
| 4 | Frame is transmitted with an error |

Figure 15.14 shows the Markov chain transition diagram and the associated transition matrix for the system is given by

$$\mathbf{P} = \begin{bmatrix} 1-a & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ a(1-e) & 1-e & 0 & 0 \\ a\,e & e & 0 & 0 \end{bmatrix}$$

The system throughput is given by

$$Th = s_3$$

The average number of lost packets per time step is given by

$$\begin{aligned} N_a(lost) &= N_a(in) - N_a(out) \\ &= a - Th \\ &= a - s_3 \end{aligned}$$

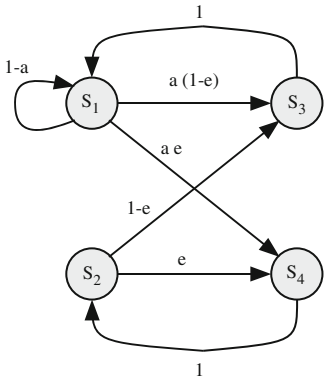



**Fig. 15.14** State -transition-rate diagram for transmitting a frame on a channel that introduces random errors

The probability that the packet will be transmitted is

$$p_a = \frac{Th}{a} = \frac{s_3}{a}$$

The packet loss probability is

$$L = \frac{N_a(lost)}{a} = 1 - \frac{s_3}{a}$$

The average number of retransmissions is given by

$$W = \frac{1 - p_a}{p_a} = \frac{a}{s_3} - 1$$

■

*Example 15.16.* Assume in the above example that $a = 0.7$ and $e = 0.1$. Obtain numerical values for the performance of the channel.

Figure 15.15 shows the variation of throughput ($Th$), delay ($W$), access probability ($p_a$), and loss probability ($L$) versus the input traffic ($a$). Two values of error probability are used $e1 = 0.1$ (solid line) and $e2 = 0.6$ (dotted line). We note that there is a maximum value on the throughput of $Th(max) = 0.5$ and that the system performance deteriorates rapidly when the error probability increases. ■

## 15.14 Problems

### *Traffic Modeling*

**15.1.** Why do we need to develop models for network traffic?

**15.2.** What is meant when we say that a source is bursty?

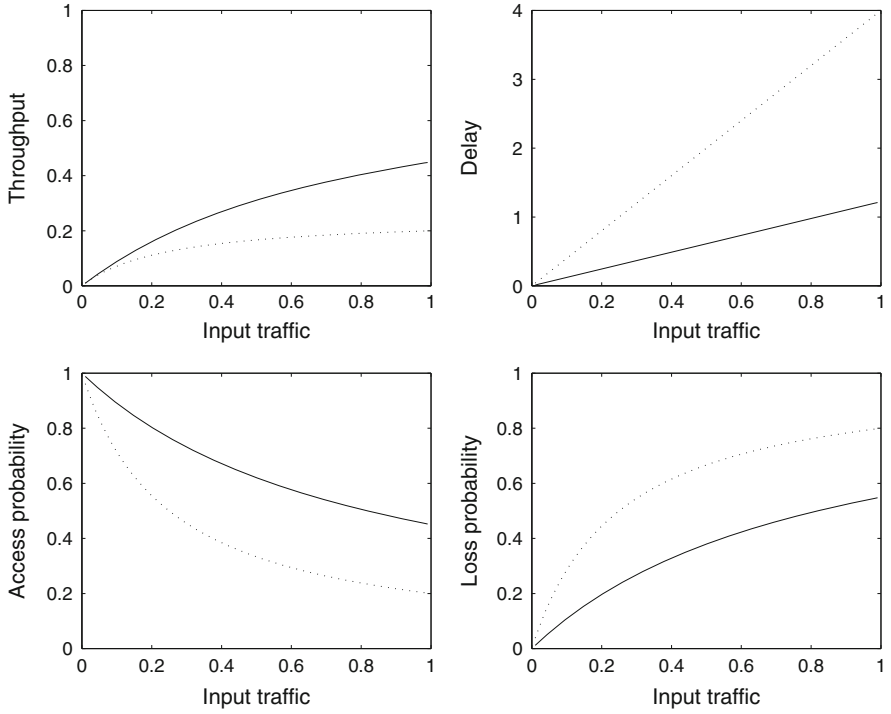



**Fig. 15.15** Throughput ($Th$), delay ($W$), access probability ($p_a$), and loss probability ($L$) versus the input traffic ($a$). Two values of error probability are used $e1 = 0.1$ (*solid line*) and $e2 = 0.6$ (*dotted line*)

**15.3.** What is meant by a point process in traffic modeling?

**15.4.** What is meant by a fluid flow model to describe network traffic?

## *Exponential Interarrival Time*

**15.5.** Obtain (15.28) on page 456 when the source maximum data rate $\sigma$ is given in units of bits/s.

**15.6.** Obtain (15.33) on page 457 where the source maximum data rate $\sigma$ and average data rate $\lambda_a$ are given in units of bits/s.

**15.7.** A Poisson packet source produces packets at the rate of 1,500 packets/s. Find the probability that a packet will be received within a 0.5 ms interval. What is the average number of packets received within this same time interval?

**15.8.** A data source has the following parameters: $\lambda_a = 200$ kbps and $\sigma = 300$ kbps. Find the pdf that describes the data rate distribution using the results of Example 15.5 on page 457.

**15.9.** Obtain the mean and variance the random variable $T$ for the exponential interarrival time whose pdf distribution is given in (15.12)

**15.10.** A radioactive material has a half-life of 1 ms. Find the average time interval between the emitted particles assuming a Poisson process. Write down an expression for the probability of detecting 5 radiated particles in a period of 0.5 ms.

**15.11.** A radioactive material has a half-life of 10 ms. An observer finds that the material did not emit a particle 20 ms, what is the probability that it will radiate a particle after 1 ms?

**15.12.** Consider the position parameter $a$ in (15.28) for a Poisson source. What are the effects of the packet length and maximum burst rate? Is a high data rate source characterized by a small or a large $a$ value?

**15.13.** Consider the position parameter $a$ in (15.33) for a Poisson source. What are the effects of the packet length, average rate, and maximum burst rate? Is a bursty source characterized by a small or a large $b$ value?

**15.14.** Obtain expressions for the position parameter $a$ and shape parameter $b$ in Eqs. (15.28) and (15.33), respectively, for exponential interarrival time distribution when the source exhibits burst rate such that $\sigma \gg \lambda_a$.

**15.15.** Obtain expressions for the position parameter $a$ and shape parameter $b$ in Eqs. (15.28) and (15.33), respectively, for exponential interarrival time distribution when the source is a CBR source such that $\sigma = \lambda_a$. Comment on your results and sketch the resulting pdf distributions.

**15.16.** A Poisson source has an interarrival time pdf distribution with the following two parameter values $a = 1$ ms and $b = 5{,}000$ packets/s. Write down an expression for the probability that the source produces 20 packets in a period of 1 ms.

**15.17.** A data source follows the Poisson distribution and has an average data rate 100 kbps and maximum burst rate of 500 kbps. Estimate the exponential distribution parameters that best describe that source assuming the average packet size is 0.5 kB.

## *Discrete Exponential Interarrival Time*

**15.18.** Because of the snow, the bus arrival times become messed up. Assume that buses should arrive on the average each 10 min. However, because of the snow, the probability that a bus arrives at this time on time is 30 %. A passenger just missed the bus, what is the probability that she will have to wait for 1 h before the next bus arrives?

**15.19.** A data source has a exponential interarrival time pdf distribution with the following two parameter values $x = 0.1$ and $b = 5000$. Write down an expression for the probability that the source produces 20 packets in a period of 20 time steps.

**15.20.** A data source follows the binomial distribution and has an average data rate 10 kbps and maximum burst rate of 100 kbps. Estimate the discrete exponential distribution parameters that best describe that source if packets are being transmitted on an ATM network operating at OC-3 (155.52 Mbps).

**15.21.** A data source follows the binomial distribution and has an average data rate 10 packets/s and maximum burst rate of 100 packets/s. Estimate the discrete exponential distribution parameters that best describe that source if packets are being transmitted on an Ethernet network operating at 10 Mbps where the average packet length is 1,024 bytes.

### *Pareto Interarrival Time*

**15.22.** Prove that the Pareto distribution is not memoryless. This implies that if a burst is received, it is likely that the burst will continue.

**15.23.** A bursty source produces data at an average rate of 5 Mbps and its maximum burst rate is 20 Mbps. Estimate the Pareto parameters that best describe that source assuming the average packet size is 400 bits.

**15.24.** Find the average interarrival time for the source in the previous problem.

### *Packet Transmission Error Description*

**15.25.** Use the results of Example 15.15 on page 486 to study the effect of channel error on data transmission. Pick some value for $a = 0.5$, say, and vary the error probability between $0.001 < e < 0.9$. Plot the system throughput and comment on your results.

**15.26.** Consider Example 15.15 and suppose that there is an upper limit on the number of retransmissions before the frame is considered lost. Obtain the resulting Markov transition diagram and the associated transition matrix.

**15.27.** Consider Example 15.15 again, and suppose that no transmissions are allowed. This could be the case for real-time data or best-effort traffic. Obtain the resulting Markov transition diagram and the associated transition matrix.

**15.28.** Consider Example 15.15 again, but this time assume that the number of errors per frame varies between 0 and 5. A forward error correction (FEC) scheme

is used where a frame is considered to be error free if it contains up to two packets in error. Obtain the resulting Markov transition diagram and the associated transition matrix.

**15.29.** Assume an adaptive FEC scheme where three levels of error correction are employed:

FEC level 1: can correct one error in received frame only.
FEC level 2: can correct up to 3 errors.
FEC level 3: can correct up to 5 errors.

When the errors in the received frame can be corrected, the next frame is transmitted using the next lower FEC level. When the errors in the received frame cannot be corrected, the frame is retransmitted using the next higher FEC level. Assume each frame to contain no more than 5 errors in it due to packet size limitations. Derive the Markov chain transition diagram and the associated transition matrix.

## References


1. D. Denterneer, V. Pronk, Traffic models for telecommunication, in *13th Symposium on Computational Statistics*, Bristol, Great Britain, pp. 269–274 (1998)
2. W. Leland, M. Taqqu, W. Willinger, D. Wilson, On the self-similar nature of Ethernet traffic. IEEE/ACM Trans. Netw. **2**, 1–15 (1994)
3. J. Beran, R. Sherman, M.S. Taqqu, W. Willinger, Variable-bit-rate video traffic and long-range dependence. IEEE/ACM Trans. Netw. **2**, 1–15 (1994)
4. V. Paxson, S. Floyd, Wide-area traffic: the failure of Poisson modeling, in *Proc. ACM Siggcomm'94*, pp. 257–268 (1994)
5. S. Molnar, A. Vidacs, *On Modeling and Shaping Self-Similar ATM Traffic, Teletraffic Contributions for the Information Age* (Elsevier Science, 1997), pp. 1409–1420, Amsterdam, Netherlands
6. Z. Sahinoglu, S. Tekinay, On multimedia networks: self-similar traffic and network performance. IEEE Communications Magazine, **37**(1), pp. 48–52 (2002)
7. M. Corvella, A. Bestavros, Self-similarity in World Wide Web traffic: evidence and possible causes. IEEE/ACM Trans. Netw. **5**, 835–846 (1996)
8. V.S. Frost, B. Melamed, Traffic modeling for telecommunications networks. IEEE Commun. Mag. **32**, 70–81 (1994)
9. M.R. Schroeder, *Number Theory in Science and Communications* (Springer, New York, 1986)
10. L.M. Sander, Fractal growth. Sci. Am. **256**, 94–100 (1987)
11. T. Hettmansperger, M. Keenan, Tailweight, statistical interference and families of distributions—a brief survey, in *Statistical Distributions in Scientific Work*, ed. by G.P. Patil et al., vol 1 (Kluwer, Boston, 1980)
12. M. Guizani, A. Rayes, *Designing ATM Switching Networks* (McGraw-Hill, New York, 1999)
13. A. Erramilli, G. Gordon, W. Willinger, Applications of fractals in engineering for realistic traffic processes. Int. Teletraffic Conf. **14**, 35–44 (1994)
14. W. Stallings, *Data and Computer Communications*, 5th edn. (Prentice Hall, New Jersey, 1998)
15. N. Likhanov, B. Tsybakov, N.D. Georganas, Analysis of an ATM buffer with self-similar ("fractal") input traffic. INFOCOM **3**, 8b.2.1–8b.2.7 (1995)
16. I. Norros, A storage model with self-similar input. Queuing Syst. **16**, 387–396 (1994)
17. J. Gordon, Pareto process as a model of self-similar packet traffic, in *IEEE Global Telecommunications Conference*, pp. 2232–2235 (1995)

18. F. Gebali, A queuing model for self-similar traffic, in *The 1st IEEE International Symposium on Signal Processing and Information Technology*, December 28–30, 2001, Cairo, Egypt
19. J.D. McCabe, *Practical Computer Network Analysis and Design* (Morgan Kaufmann, San Francisco, 1998)
20. J.M. Pitts, J.A. Schormans, *Introduction to ATM Design and Performance* (Wiley, New York, 1996)
21. H. Saito, *Teletraffic Technologies in ATM Networks* (Artech House, Boston, 1994)
22. G.E.P. Box, G.M. Jenkins, *Time Series Analysis: Forecasting and Control* (Prentice-Hall, New Jersey, 1976)
23. F.A. Tobagi, T. Kwok, F.M. Chiussi, Architecture, performance, and implementation of the tandem-banyan fast packet switch. Proc. IEEE **78**, 133–166 (1990)
24. V.J. Friesen, J.W. Wong, The effect of multiplexing, switching and other factors on the performance of broadband networks, in *Proc. IEEE INFOCOM'93*, pp. 1194–1203 (1993)
25. G.F. Pfister, V.A. Norton, Hot-spot contention and combining in multistage interconnection networks. IEEE Trans. Comput. **34**, 943–948 (1993)

# Chapter 16
# Scheduling Algorithms

## 16.1 Introduction

A scheduling algorithm must be implemented at each network router or switch to enable the sharing of the switch limited resources among the packets traveling through it. The resources being shared include available link bandwidth and buffer space. The main functions of a scheduler in the network are (1) provide required quality of service (QoS) for the different users, by making proper choices for *selecting* next packet for forwarding to the next node; (2) select next packet for *dropping* during periods of congestion when the switch buffer space is starting to get full, and (3) provide *fair sharing* of network resources among the different users.

### *Packet Selection Policy*

In a typical switch or router, several packets will request to access a certain output port. Because only one packet can leave, the rest must be stored in an intermediate buffer. Somehow we must find a way to decide which stored packet must be sent next. Different selection policies could be implemented for different types of queues depending on the service classes of the queues. For example, some applications have rigid real-time constraints on delay and jitter, while other adaptive applications agree to modify their behavior based on the network status. At the time of writing, most Internet applications are handled using *best-effort* packet transfer policy with no bandwidth or delay guarantees [1].



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6_16

### *Packet Dropping Policy*

The fact that packets have to be stored in each switching node of the network implies that buffer storage in a switch is a resource that must be shared among the different users or sessions. During periods of congestion, the switch buffers become full and the scheduler must also decide which packets to drop.

### *Fair Sharing Policy*

The switch resources such as available output link bandwidth and local buffer space must be shared among the switch users. Because of the different classes of service supported by the switch, an equal sharing of the resources is not the best option. Rather, the scheduler must allocate these finite resources in a *fair* manner so that each user can get its share based on its class of service. Another issue related to fair sharing is *isolation* or *protection*. This is required because not all users abide by their agreed upon data rate. When this happens, the misbehaving (nonconforming) user starts to hog the resources at the expense of other well-behaving (conforming) users.

It is obvious from the previous discussion that data scheduling is required at each node in the network for three reasons: (a) *selection* of a packet for transmission from the population of queued packets destined to a certain switch output; (b) *provide QoS* for the different types of flows going through the switch; and (c) *drop* packets when the buffer space becomes full.

## 16.2 Scheduling as an Optimization Problem

From the above discussion it is clear that the scheduling problem is an optimization problem since the scheduler distributes the system limited resources among the user traffic which impacts the offered QoS. The scheduling algorithms to be discussed in the following sections are all heuristic and have no solid proof that the proposed scheduling policy is optimal in any mathematical sense. Further, we will note that the scheduler discussed here reduces packet delay by allocating large bandwidth to the delay-sensitive traffic. This might be self-contradictory for some types of traffic which is delay-sensitive but does not require high bandwidth.

The author developed a hierarchical scheduler [2] that is based on the transportation problem optimization technique [3, 4]. The transportation problem technique allows optimizing different QoS types such as delay-sensitive traffic and bandwidth-sensitive traffic through the same switch.

Finding the optimum solution to a transportation problem is not simple and might consume a certain amount of time. However, the author developed a simple greedy

algorithm for solving the transportation problem based on computational geometric concepts. The algorithm uses only simple add/subtract operations and hence should be fast to compute [2].

## 16.3 Scheduler Location in Switches

The scheduler will be located in the switch where packets are buffered and where packets must share a resource like the switch fabric or the output links.

When a switch is capable of supporting different QoS classes, each service class will have its own dedicated buffer. At the extreme, each session or user channel will have its own dedicated buffer in each switch it encounters. Figure 16.1 shows two buffer location options in switches. The rectangles labeled "SF" indicated the switching fabric of the switch whose function is to provide connectivity between each input port and each output port. The figure on the top shows an input queuing switch where the buffers are located at each input port. The figure on the bottom shows an output queuing switch where the buffers are located at each output port. Multiple buffers are shown at each input or output port when multiple service classes are supported.

Irrespective of the buffering strategy employed, packets will contend for the shared resources and some form of scheduling algorithm will be required. The figure shows that there are three types of shared resources: (1) the buffer storage space, (2) the switch fabric (SF), and (3) the output links (i.e., available bandwidth)

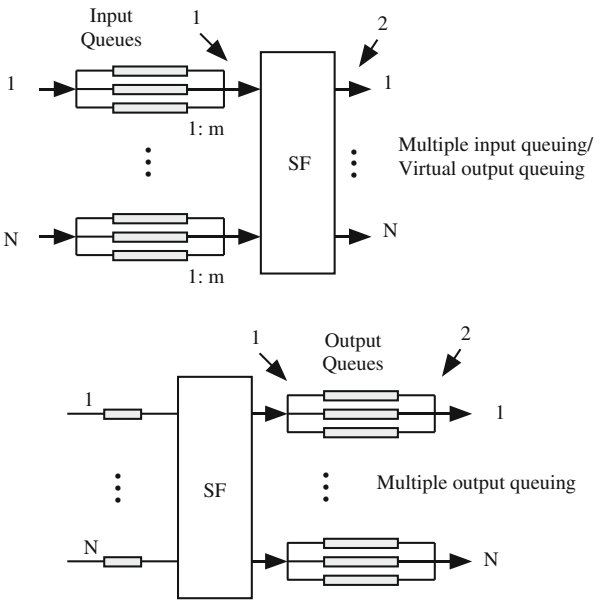



**Fig. 16.1** Virtual output queuing (VOQ) and output queuing switches with different queues for each service class. The points at which packet contention occurs are shown

For the input queuing switch, top sketch in Fig. 16.1, we can get two potential contention points. Point 1 is a potential contention point where all the packets from the different queues at an input port compete to access the switch fabric. Of course a remedy for this problem would be to modify the switch fabric to allow more than one packet to access the fabric simultaneously. At point 1 the scheduler must also determine which packets to drop when the buffers start being full. Point 2 is another potential contention point where packets from the different inputs compete to access a certain output port. At this point, the output link usually is only able to transmit one packet only except when the output link is composed of multiple channels such as in wavelength division multiplexing (WDM).

For output queuing switch, bottom sketch in Fig. 16.1, we can get two contention points also. Point 1 is a potential contention point where all the packets from the different inputs compete to access the queues of a certain output port. Point 2 is another potential contention point where packets from the different queues in each output compete to access the output link. At point 2 the scheduler must also determine which packets to drop when the buffers start being full.

## 16.4 Scheduling and Medium Access Control

From the discussion in the previous section we can tell that a scheduling algorithm is a method to allow many users or traffic flows to access the output link. In that sense the problem is access to the shared resource. Chapter 10 discusses medium access control techniques which allow packets to access a shared resource such as the communication channel. A question naturally arises whether schedulers and media access control (MAC) protocols are one and the same. The quick answer is that they are similar but not the same thing. Table 16.1 compares scheduler algorithms and MAC techniques.

**Table 16.1** Comparison between scheduling algorithms and MAC techniques

| Scheduling algorithms | MAC techniques |
|---|---|
| Designed to provide QoS guarantees | Designed to provide resource access only |
| Determine the user's share of bandwidth, buffer space allocation and packet discard decisions | Does not deal with these issues |
| Shared resource is outgoing link of a switch/router and buffer space | Shared resource is a bus, a wireless channel, or a shared memory |
| Operate at switch or router output ports | Operate at output ports or each device connected to the medium |
| Used with switches or routers | Used with buses, wireless channels, etc. |
| Physically exists inside a switch or router | Distributed among all users and MAC controllers |
| Implemented in software (at least for now) | Implemented in software and hardware |
| Operate at layers above the physical layer | Operate at the physical layer |

## 16.5 Scheduler Design Issues

There are several design issues related to schedulers. These issues are reviewed in the following subsections. In the discussion to follow we shall use the terms "user," "session," "connection," or "flow" to describe the traffic carried by the switch.

### *16.5.1 Priority*

Ideality dictates that equality is a good thing. However, real life tells us that special people are "more equal"! Alas, equality is not a good thing in computers or networks. The scheduler is only able to do a decision to select a certain user only when this user has higher priority compared to all other users. Priority assignment could be static or dynamic. A static priority assignment implies that users belonging to a certain class of service have a certain level of priority that does not change with time or with the traffic load. On the other hand, dynamic priority allocation changes the priority of a certain class of service with time or based on the traffic volume.

In addition to priority assignment, there is an *arbitration rule* that is invoked to resolve the conflicts between users with the same priority.

### *16.5.2 Degree of Aggregation*

In order to provide guaranteed QoS, the scheduler has to keep *state information* about the performance of each user or connection. The problems with such schedulers are limitations on scaling, deployment difficulties, and the requirement of mapping between application and network service parameters [5]. The large number of states that must be maintained and references slow down the scheduler and limit the number of users that can be accepted. Other schedulers aggregate several connections into classes to reduce the amount of state information and workload. This approach is more promising since it deals with large aggregates of network traffic and its per-hop behavior is configurable [6]. The differentiated services scheduler provides constant *ratios* of the QoS ratios between the service classes even when the quality level is varying. The price to be paid by aggregating traffic is loss of deterministic QoS guarantees since the state of each connection is lost. QoS guarantees for a high-level aggregation server are provided on a probabilistic basis. In other words, the scheduler loses specific information about the status of each user since it only keeps track of groups of users. This reduces the number of states that must be checked and updated. Guarantees can be provided on the QoS for groups of users, but each user cannot be guaranteed specific level of service.

### *16.5.3 Work-Conserving vs Non-Work-Conserving*

A work-conserving algorithm is idle when all of the priority queues are empty. A non-work-conserving algorithm might not transmit a packet even when the priority queues are occupied. This might happen to reduce the delay jitter for example [7]. This is nice in theory but is not implemented in practice. In general it was found that work-conserving algorithms provide lower average delay than non-work-conserving algorithm.

Examples of work-conserving algorithms include generalized processor sharing (GPS) [8], weighted fair queuing (WFQ) [9], Virtual Clock [10], Weighted Round Robin [11], Delay-Earliest-Due-Date (Delay-EDD) [12], and deficit round robin (DRR) [13].

Examples of non-work-conserving algorithms are Stop-and-Go, jitter earliest due date (jitter-EDD) [14], and rate-controlled static priorities (RCSP).

### *16.5.4 Packet Drop Policy*

Schedulers not only select which packet to serve next, but also they have to select which packet to drop when the system resources become overloaded. There are several options for dropping packets such as dropping packets that arrive after the buffer reaches a certain level of occupancy (this is known as tail dropping). Another option is to drop packets from any place in the buffer depending on their priority. As we shall see later, a third approach is to randomly select packets for dropping once the system resources become congested.

## 16.6 Rate-Based vs Credit-Based Scheduling

Scheduling methods could be classified broadly as rate-based or time-based. A rate-based scheduler selects packets based on the data rate allocated for the service class. Rate-based scheduling methods include fair queuing (FQ) [9] and [15] which is equivalent to virtual clock (VC) [10] WFQ [9], hierarchical round robin (HRR) [16], DRR [13], stop-and-go (S&G) [17–20], and RCSP [21]. Rate-based methods allow a variety of techniques for arriving at a service policy [22]. Some of the methods are fairly complex to implement since they require complex mathematical operations and require knowledge of the states of the different flows. However, they can only provide guarantees on the maximum delay or delay jitter bounds since they translate these requirements into an equivalent bandwidth. This proves to be the weak point on these algorithms since providing bandwidth guarantees to effect delay

guaranteers ignores the actual bandwidth requirements of the actual traffic stream. Needless to say, this would lead to unfair bandwidth allocation to the other services that have real QoS bandwidth requirements.

A time-based scheduler selects packets based on their time of arrival. Time-based methods include earliest-due date for delay (EDD-D) [23], earliest-due date for jitter (EDD-J) [24], and smallest response time (SRT) [25]. Scheduler-based methods require keeping track of the arrival times of the different packets and calculate the packet priority based on the packet arrival time and the deadline imposed on it. To provide end-to-end guarantees on the delay and delay jitter, the scheduler must be implemented at all the switching nodes and the incoming traffic must conform very closely with the assumed model [22].

## 16.7 Scheduler Performance Measures

A good scheduling algorithm must satisfy several of the following performance measures [26].

QoS The scheduler must be able to support different types of QoS classes with varying requirements such as bandwidth (throughput), delay, delay jitter, and loss probability [27].

Fairness: The main goal of fairness is to serve sessions in proportion to some specified value [28]. The simplest fair allocation of resources is to equally divide the available bandwidth and buffer space among all the users and to drop excess packets equally from the different queues. However, if a user does not require all of its allocated share, then the excess share should be divided equally among the other users.

Isolation or Protection: Isolation means that a misbehaving user should not adversely impact other users. The user becomes misbehaving when its packet arrival rate exceeds what is expected. Isolating the effects of misbehaving users is equivalent to *protecting* conforming users.

Simplicity: The scheduler must be easy to implement in hardware especially at high network speeds. This requires that computations to be done by the scheduler to be small in number and simple to calculate.

Scaling: The scheduler must perform well even when the number of sessions increases or when the link speed is increased. Some scheduling algorithm must keep state information about every user and must update this information very frequently. This places limitations on how many users can be supported by the scheduler and places limitations on the scheduler delay.

## 16.8 Analysis of Common Scheduling Algorithms

The remainder of this chapter discusses several scheduling algorithms that vary in performance from support of best-effort traffic with no performance guarantees to schedulers that support guaranteed services with bounds on bandwidth and delay. Models that describe the performance of each algorithm are also developed.

## 16.9 First-In/First-Out

First-in/first-out (FIFO) is also known as first-come/first-served (FCFS). The FIFO method sorts users according to their arrival time. The users that arrive early are served first [29]. In that sense, all users have the same priority and the *arbitration rule* is based on the time of arrival of the packets. This method is used naturally to store incoming packets in queues that could be associated with the input or output ports.

FIFO is simple to implement and packet insertion and deletion are particularly simple and do not require any state to be maintained for each session. However, this proves to be a disadvantage since the server has no way of distinguishing between the packets belonging to different users. Thus some users might be misbehaving and fill a large portion of the queue which increases the chance of dropping the packets of other users.

When all incoming flows are queued in a single queue, greedy users exhibiting long periods of activity (bursty behavior) will take the lion's share of the queue capacity at the expense of other flows. When the queue is filled, packets at the tail are dropped. This is the reason why this method is known as *FIFO with tail drop*. Most routers adopt this method because of its simplicity.

FIFO does not provide per-connection delay or rate guarantees since priority is based solely on arrival time. One way to provide delay bounds is to limit the size of the queue so that the maximum delay equals the time to serve a full queue. Of course, once the queue size is limited, there will be a probability that an arriving packet will be discarded if it arrives when the queue is full. To reduce the packet discard probability, the number of sessions should be limited.

### *16.9.1 Queuing Analysis of FIFO/FCFS*

Let us perform a simple queuing analysis for a FIFO buffer. We make the following simplifying assumptions.

1. The size of the buffer is $B$.
2. The maximum number of customers that could arrive at the input of the buffer at a certain time step is $m$.

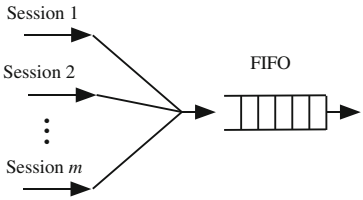



**Fig. 16.2** A FIFO buffer with $m$ flows at its input

3. The average length of packets from any user is $A$.
4. The arrival probability for any user is $a$.
5. The departure probability for the buffer is $c$.

Figure 16.2 shows the FIFO buffer where several sessions converge at the input and only one packet can leave which correspond to an $M^m/M/1/B$ queue. The transition matrix for such queue was derived in Sect. 7.7 on p. 249. Based on that we can derive expressions for the scheduling delay which corresponds to the queuing delay in that situation.

The throughput of the FCFS queue was given in Sect. 7.7, which we repeat here for convenience.

The average throughput is estimated as

$$Th = c\ (1 - a_0\ s_0) \tag{16.1}$$

where $a_0$ is the probability that no packets arrive during a time step and $s_0$ is the probability that the queue is empty.

The average lost traffic $N_a(lost)$ is given by

$$\begin{aligned} N_a(lost) &= N_a(in) - N_a(out) \\ &= N_a(in) - Th \\ &= m\ a - c\ (1 - a_0\ s_0) \end{aligned} \tag{16.2}$$

We refer the reader to Sects. 7.7.1 and 7.7.2 for a more detailed discussion of the performance figures for the $M^m/M/1/B$ queue.

## 16.10 Static Priority Scheduler

In a static priority scheduler, separate queues are assigned different priorities. Incoming data is routed to the different queues depending on their priority. The scheduler serves packets in a lower priority queue only if all the higher priority queues are empty [29–33].

The static priority scheduler is also known as the IEEE 802.1p which was discussed in Chap. 10. A queuing analysis of the static priority scheduler was performed in Sect. 10.2 on p. 338.

## 16.11 Round Robin Scheduler

The round robin scheduler serves backlogged sessions one after another in a fixed order. When the scheduler finishes transmitting a packet from session 1, say, it moves on to session 2 and checks if there are any packets waiting for transmission. After the scheduler has finished going through all sessions, it comes back to the first session, hence the name round robin. Figure 16.3 shows a round robin scheduler serving four sessions.

The main features of this scheduler are ease of implementation in hardware and protection of all sessions even best-effort sessions. A greedy or misbehaving user will not be able to transmit more than one packet per round. Hence all other users will not be penalized. The misbehaving user will only succeed in filling its own buffer and data will start getting lost. However, the long packets will be transmitted since the scheduler serves whole packets. This could affect the delay experienced by other sessions that might have short packets to transmit.

Assuming $m$ sessions, the maximum bound on delay for a queue is given by

$$W(\max) = \sum_{i=1}^{m} \frac{A_i}{C} \quad \text{s} \tag{16.3}$$

where $A_i$ is the head of the line (HOL) packet length in bits for session $i$ and $C$ (bps) is the output link rate.

The ratio of service provided for session $i$ relative to the total service provided to all sessions is equivalent to finding the ratio of number of bits moved from session $i$ relative to the total number of bits moved in one round. We define $f_i$ as that ratio which is written as

$$f_i = \frac{A_i}{\sum_{j=1}^{m} A_j} \tag{16.4}$$

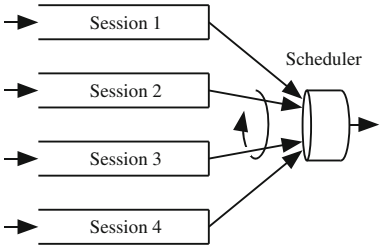



**Fig. 16.3** Round robin scheduler serving four sessions

Thus the bandwidth $C_i$ given to session $i$ relative to the total output bandwidth is simply given by

$$C_i = C \ f_i \tag{16.5}$$

In an ideal round robin algorithm, all packets have equal lengths and each session share would be

$$f_i = \frac{1}{m} \tag{16.6}$$

$$C_i = \frac{C}{m} \tag{16.7}$$

The above two equations assume that all sessions are backlogged and have packets with equal lengths. The algorithm is not fair when some sessions have variable length packets since the server will allocate more time for these sessions.

### *16.11.1 Queuing Analysis for RR*

We can study the occupancy of each queue in a round robin scheduler using the following assumptions.

1. The outgoing link capacity is $C$.
2. The number of queues or sessions is $m$.
3. The size of queue $i$ is $B_i$.
4. Time step equals $T$, the duration of one round of the scheduler.
5. The input data rate for session $i$ is $\lambda_i$.
6. The maximum burst rate for queue $i$ is $\sigma_i$.
7. The head of line packet length for queue $i$ is $A_i$.
8. The arrival probability for queue $i$ is $a_i$.
9. A maximum of $N_i$ packets could arrive during one round into queue $i$.
10. The size of queue $i$ is $B_i$.
11. The probability of departure from queue $i$ is $c_i = 1$.

Based on the above assumptions we find that we have an $M^m/M/1/B$ queue. The transition matrix for such queue was derived in Sect. 7.7 on p. 249. Based on that we can derive expressions for the scheduling delay which corresponds to the queuing delay in that situation.

The arrival probability $a_i$ can be found as follows. The duration of one round $T$ is given by

$$T = \sum_{i=1}^{m} \frac{A_i}{C} \qquad \text{s} \tag{16.8}$$

and when all sessions have packets with equal length $A$, the above expression simplifies to

$$T = \frac{m\ A}{C} \qquad \text{s} \tag{16.9}$$

The average interarrival time for session $i$ is given by

$$T_i = \frac{A_i}{\lambda_i} \qquad \text{s} \tag{16.10}$$

The probability of $k$ arrivals in one time step is given by

$$p_{i,k} = \binom{N_i}{k} a_i^k\ b_i^{N_i - k} \qquad k = 0, 1, 2, \ldots, N_i \tag{16.11}$$

where $a_i$ is the Bernoulli probability of packet arrival, $b_i = 1 - a_i$ and $N_i$ is the maximum number of packets that could arrive at the queue input as determined by the maximum burst rate $\sigma_i$

$$N_i = \lceil \frac{\sigma_i \times T}{A_i} \rceil \tag{16.12}$$

with $\lceil x \rceil$ is the smallest integer that is larger than or equal to $x$. Assuming binomial distribution, we can estimate $a_i$ from the average number of packets received on one time step:

$$a_i \times N_i = \frac{\lambda_i \times T}{A_i} \tag{16.13}$$

which gives

$$a_i = \frac{\lambda_i \times T}{N_i\ A_i} \tag{16.14}$$

Because of our choice for the step size, the queue size can only decrease by one at most at any instant with probability $c_i = 1$.

Assuming the packet buffer size $B_i$, the transition matrix for queue $i$ will be $(B_i + 1) \times (B_i + 1)$ and is given

$$\mathbf{P} = \begin{bmatrix} q & p_0 & 0 & 0 & 0 & \cdots & 0 \\ p_2 & p_1 & p_0 & 0 & 0 & \cdots & 0 \\ p_3 & p_2 & p_1 & p_0 & 0 & \cdots & 0 \\ p_4 & p_3 & p_2 & p_1 & p_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ p_{B_i} & p_{B_i-1} & p_{B_i-2} & p_{B_i-3} & p_{B_i-4} & \cdots & p_0 \\ f_1 & f_2 & f_3 & f_4 & f_5 & \cdots & f_{B_i+1} \end{bmatrix} \tag{16.15}$$

where $q = p_0 + p_1$ and

$$f_j = 1 - \sum_{k=0}^{B_i-j+1} p_k \tag{16.16}$$

Of course, if $B_i > N_i$, then the terms $p_j = 0$ whenever $N_i < j \leq p_{B_i}$.

The transition matrix helps us find performance parameters for queue $i$. For that we need to determine the equilibrium distribution vector $\mathbf{s}$. Repeating the same procedure for all other queues, we would then be able to find the performance of the round robin scheduler.

For queue $i$, the throughput is given by

$$Th = 1 - s_0 \qquad \text{packets/time step} \tag{16.17}$$

where $s_0$ is the probability that the queue is empty. The throughput in units of packets/s is simply given by

$$Th = \frac{1 - s_0}{T} \qquad \text{packets/s} \tag{16.18}$$

And the throughput in units of bps is simply given by

$$Th = \frac{(1 - s_0)\ A_i}{T} \qquad \text{bps} \tag{16.19}$$

The average queue length is given by

$$Q_a = \sum_{j=0}^{B_i} i\ s_j \tag{16.20}$$

We can invoke Little's result to estimate the *wait time*, which is the average number of time steps a packet spends in the queue before it is served, as

$$Q_a = W \times Th \tag{16.21}$$

where $W$ is the average number of time steps that a packet spends in the queue.

$$W = \frac{Q_a}{1 - s_0} \qquad \text{time steps} \tag{16.22}$$

The wait time in units of seconds is simply given by

$$W = \frac{Q_a \, T}{(1 - s_0) \, A_i} \qquad \text{s} \tag{16.23}$$

*Example 16.1.* Assume a round robin scheduler in which all sessions are identical with the following parameters:

$$\begin{array}{cc} N = 4 \text{ sessions} & C = 1 \text{ Mbps} \\ B = 5 \text{ packets} & L = 1{,}024 \text{ bits} \\ \lambda = 50 \text{ kbps} & \sigma = 500 \text{ kbps} \end{array}$$

Determine the transition matrix and determine the system performance parameters.

The duration of one round is

$$T = 4 \times \frac{1024}{5 \times 10^6} = 4.1 \qquad \text{ms}$$

Based on the data burst rate, a maximum of two packets could arrive during a time period $T$. The Bernoulli arrival probability at each queue during this time period is given by

$$a = 0.1$$

The transition matrix is $6 \times 6$

$$\mathbf{P} = \begin{bmatrix} 0.99 & 0.81 & 0 & 0 & 0 & 0 \\ 0.01 & 0.18 & 0.81 & 0 & 0 & 0 \\ 0 & 0.01 & 0.18 & 0.81 & 0 & 0 \\ 0 & 0 & 0.01 & 0.18 & 0.81 & 0 \\ 0 & 0 & 0 & 0.01 & 0.18 & 0.81 \\ 0 & 0 & 0 & 0 & 0.01 & 0.19 \end{bmatrix}$$

The equilibrium distribution vector is

$$\mathbf{s} = \begin{bmatrix} 0.9877 & 0.0122 & 0.0002 & 0 & 0 & 0 \end{bmatrix}^t$$

The throughput of the queue is

$$\begin{aligned} Th &= 1 - s_0 = 0.0124 \qquad \text{packets/time step} \\ &= 3.0864 \qquad \text{kbps} \end{aligned}$$

The average queue length is

$$Q_a = 0.0125$$

The average wait time for a queue is

$$W = 4.05 \qquad \mu s$$ ■

## 16.12 Weighted Round Robin Scheduler

The round robin scheduler discussed in the previous section treats all connections equally. When each connection has a different weight we get the weighted round robin (WRR) scheduler.

Assume session $i$ has an integer weight $w_i$ associated with it. Then in each round of service, the WRR scheduler transmits $w_i$ packets for session $i$, and so on.

The fraction of the bandwidth for session $i$ in this algorithm can be measured as the service associated with session $i$ relative to the total service in one round. The service received by the HOL packet in queue $i$ is simply the number of bits transmitted.

$$f_i = \frac{w_i \ A_i}{\sum_{j=1}^{m} w_j \ A_j} \tag{16.24}$$

In an ideal WRR algorithm, all packets have equal lengths and share would be

$$f_i = \frac{w_i}{\sum_{j=1}^{m} w_j} \tag{16.25}$$

The algorithm is not fair when some sessions have variable length packets since the server will allocate more time for these sessions.

### *16.12.1 Queuing Analysis for WRR*

We can study the occupancy of each queue in a WRR scheduler using the following assumptions.

1. The outgoing link capacity is $C$.
2. The number of queues or sessions is $m$.
3. The size of queue $i$ is $B_i$.
4. The weight associated with queue $i$ is $w_i$.
5. Time step equals $T$, the duration of one round of the scheduler.
6. The input data rate for session $i$ is $\lambda_i$.

7. The maximum burst rate for queue $i$ is $\sigma_i$.
8. The head of line packet length for queue $i$ is $A_i$.
9. The arrival probability for queue $i$ is $a_i$.
10. A maximum of $N_i$ packets could arrive during one round into queue $i$.
11. A maximum of $w_i$ packets could leave during one round from queue $i$.
12. The probability of departure from queue $i$ is $c_i = 1$.

Based on the above assumptions we find that we have an $M^m/M^m/1/B$ queue.

The arrival probability $a_i$ can be found by first selecting a time step value. We choose the time step to be equal to the duration of one round $T$

$$T = \sum_{i=1}^{m} \frac{w_i \ A_i}{C} \tag{16.26}$$

Next, we have to estimate how many packets arrive in one time step from session $i$. The average interarrival time for session $i$ is given by

$$T_i = \frac{A_i}{\lambda_i} \tag{16.27}$$

The probability of $k$ arrivals in one time step is given by

$$p_{i,k} = \binom{N_i}{k} a_i^k \ b_i^{N_i - k} \qquad k = 0, 1, 2, \ldots, N_i \tag{16.28}$$

where $a_i$ is the Bernoulli probability of packet arrival, $b_i = 1 - a_i$ and $N_i$ is the maximum number of packets that could arrive in one time step. $N_i$ is determined by the maximum burst rate $\sigma_i$ as follows

$$N_i = \lceil \sigma_i \times T \rceil \tag{16.29}$$

where the ceiling function $\lceil x \rceil$ is the smallest integer that is larger than or equal to $x$. Assuming binomial distribution, we can estimate $a_i$ from the average number of packets received on one time step:

$$a_i \times N_i = \lambda_i \times T \tag{16.30}$$

which gives

$$a_i = \frac{\lambda_i \times T}{N_i} \tag{16.31}$$

Because of our choice for the step size, the queue size can decrease by $w_i$ packets at most at any instant with probability $c_i = 1$.

From the above calculations we are able to construct a transition matrix for the queue of session $i$. Having done that, we are able to obtain expressions for the queue parameters such as throughput, delay, and average queue length.

## 16.13 Max–Min Fairness Scheduling

Scheduling deals with the sharing of a resource among several users. However, not all users have the same demands on the resource. In many cases not all users require an equal share of that bandwidth. How should we divide the bandwidth among all users in a fair manner? One way to do that is to use the max–min policy.

Max–min sharing policy is an iterative technique for fair sharing of the resource. For example, assume the outgoing link capacity is $C$ and there are $m$ users. Assume user $i$ requires a bandwidth $\lambda_i$ and the users are sorted such that

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_m \tag{16.32}$$

The allocation of the bandwidth proceeds as follows.

1. Allocated the bandwidth equally among all users $C/m$.
2. If $\lambda_1 < C/m$, then give user 1 only $\lambda_1$.
3. Allocate the remaining bandwidth $C - \lambda_1$ equally among the remaining users $(C - \lambda_1)/(m - 1)$.
4. If $\lambda_2 < (C - \lambda_1)/(m - 1)$, then give user 2 only $\lambda_2$.
5. Repeat the procedure until all users have been considered.

*Example 16.2.* Assume an outgoing link is being shared among five channels. The system parameters (in units of Mbps) are as follows:

$$C = 155$$
$$\lambda_1 = 10$$
$$\lambda_2 = 20$$
$$\lambda_3 = 60$$
$$\lambda_4 = 80$$
$$\lambda_5 = 80$$

Find the rates assigned to each flow according to the max–min algorithm.

The sum of the flows due to all users is

$$10 + 20 + 60 + 80 + 80 = 250 \qquad \text{Mbps}$$

which is larger than the link capacity. The initial fair rate $f$ is given by

$$f = 155/5 = 31 \qquad \text{Mbps}$$

Flow 1 has the minimal rate and is assigned the flow

$$\lambda_1' = \min(10, 31) = 10 \qquad \text{Mbps}$$

We adjust the link capacity shared among the remaining four users as

$$C = 155 - 10 = 145 \qquad \text{Mbps}$$

The fair rate among four remaining users becomes

$$f = 145/4 = 36.25 \qquad \text{Mbps}$$

Flow 2 has minimal rate and is assigned the flow

$$\lambda_2' = \min(20, 36.25) = 20 \qquad \text{Mbps}$$

We adjust the link capacity shared among the remaining two users as

$$C = 145 - 20 = 125 \text{ Mbps}$$

The fair rate among three remaining users becomes

$$f = 125/3 = 41.7 \qquad \text{Mbps}$$

Flow 3 has minimal rate and is assigned the flow

$$\lambda_3' = \min(60, 41.7) = 41.7 \qquad \text{Mbps}$$

We adjust the link capacity shared among the remaining two users as

$$C = 125 - 41.7 = 93.3 \qquad \text{Mbps}$$

The fair rate among two remaining users becomes

$$f = 93.3/2 = 46.7 \qquad \text{Mbps}$$

The bandwidths assigned to the flows become

$$\lambda_1' = 10 \qquad \text{Mbps}$$
$$\lambda_2' = 20 \qquad \text{Mbps}$$

$$\lambda_3' = 41.7 \quad \text{Mbps}$$
$$\lambda_4' = 46.7 \quad \text{Mbps}$$
$$\lambda_5' = 46.7 \quad \text{Mbps}$$

Note that the aggregate assigned rates equal the outgoing link capacity. ■

## 16.14 Processor Sharing

Processor sharing is an ideal work-conserving scheduler that treats each flow like a fluid model. Data of a given flow is assumed to be infinitely divisible (even at level smaller than a bit), and all flows are served simultaneously. Processor sharing provides max–min fair service. However, PS is an ideal solution that is not implementable in practice. We study it only to provide the background to other practical algorithms.

Assuming we have $m(t)$ active flows at a given time $t$, the rate assigned to flow $i$ is given by

$$\lambda_i(t) = \frac{C}{m(t)} \tag{16.33}$$

where $C$ is the outgoing link capacity. Figure 16.4 shows a PS scheduler serving four queues.

*Example 16.3.* Assume a PS scheduler that serves three flows. Packets arrive at the scheduler according to the following table.

| Time | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Flow 1 | 0 | 1 | 0 | 0 | 1 |
| Flow 2 | 1 | 1 | 1 | 0 | 1 |
| Flow 3 | 0 | 0 | 1 | 1 | 1 |

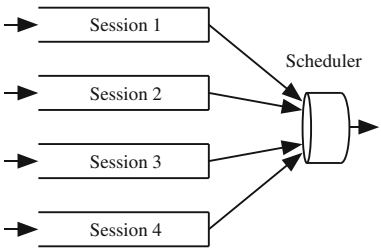



**Fig. 16.4** Processor sharing (PS) scheduling in which each queue shares an equal portion of server bandwidth in a fluid flow fashion

A "0" indicates that the flow is inactive and does not require any output bandwidth. A "1" entry indicates an active flow that requires a fair portion of the outgoing link capacity. For simplicity, we assume discrete time intervals $T_0$, $T_1$, etc. Calculate the percentage of the outgoing link capacity allocated to each active flow at the different times.

According to PS scheduling, the percentage of the bandwidth dedicated to each active flow is shown in the following table. ■

| Time | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Flow 1 | 0 | 1 | 0 | 0 | 1 |
| Flow 2 | 1 | 1 | 1 | 0 | 1 |
| Flow 3 | 0 | 0 | 1 | 1 | 1 |
| % rate | 100 | 50 | 50 | 100 | 33.3 |

## 16.15 GPS

GPS is an ideal work-conserving scheduling scheme that is not implemented in practice. It only helps as a reference to compare the performance of other more practical scheduling algorithm. GPS assumes a fluid flow traffic model where the scheduler is able to serve an infinitely small amount of data from each queue at the same time and at different rates.

The bit-by-bit GPS server works on incoming flows in a round robin fashion transmitting one bit from each flow before it moves on to the next flow. When a session is idle, the server skips over to the queue of the next session. Therefore, a single packet requires several rounds of service before it is able to move out of the scheduler but each packet is guaranteed a fair share of the outgoing link capacity.

In processor sharing all flows had the same weight and all active flows had an equal share of the outgoing link capacity. In GPS each session or flow is assigned a weight that indicates the desired share of the outgoing link capacity. Flow $i$ will have a weight $w_i \geq 1$ and the share of session $i$ out of the outgoing link bandwidth $C$ (bps) is given by

$$c_i(t) = C \times \frac{w_i}{\sum_{j \in B(t)} w_j} \tag{16.34}$$

where $B(t)$ is the set of backlogged sessions at time $t$.

The number of bits transmitted from flow $i$ in a time period $t_2 - t_1$ is given by

$$s_i = c_i \ \ (t_2 - t_1) \ \ bits \tag{16.35}$$

where $t_2 \geq t_1$.

The time $T$ required to completely transmit a packet of length $A_i$ in flow $i$ is determined from the expression

$$A_i = \int_{t=0}^{T} c_i(t)\, dt \tag{16.36}$$

This time depends on the number of backlogged sessions which could vary.

*Example 16.4.* Assume a GPS scheduler serving four flows with associated weights $w_1 = 1$, $w_2 = 2$, $w_3 = 3$, and $w_4 = 4$. The outgoing link capacity is 1 Mbps and the packet arrival pattern in the different flows is as shown below, where the numbers in

| $t$ (ms) | 0 | 10 |
|---|---|---|
| Session 1 | 3 | |
| Session 2 | | 1 |
| Session 3 | 2 | |
| Session 4 | | 7 |

the rows of each session indicate the length of the packet that arrived at that time in units of kb. Calculate the assigned bandwidth and completion times for the arriving packets.

At $t = 0$ sessions 1 and 3 are backlogged and their combined weights are $1+3 = 4$. The bandwidth assigned to sessions 1 and 3 is

$$\begin{aligned} c_1(0) &= 10^6 \times 1/4 = 0.25 \qquad \text{Mbps} \\ c_3(0) &= 10^6 \times 3/4 = 0.75 \qquad \text{Mbps} \end{aligned}$$

Assuming the system is not changed, the completion times for the backlogged packets are

$$\begin{aligned} t_1 &= 3 \times 10^3/0.25 = 12 \qquad \text{ms} \\ t_3 &= 2 \times 10^3/0.75 = 2.67 \qquad \text{ms} \end{aligned}$$

at $t = 10$ packet 3 is gone but a small portion of packet 1 is still left.

The number of bits transmitted from packet 1 of session 1 is given from (16.35) by

$$s_1 = 0.25 \times 10 = 2.5 \qquad \text{kb}$$

Thus at $t = 10$ ms there are 0.5 kb still left to be transmitted for packet $p_1(1)$.

The bandwidth assigned to sessions 1, 3, and 4 is

$$\begin{aligned} c_1(10) &= 10^6 \times 1/8 = 0.125 && \text{Mbps} \\ c_3(10) &= 10^6 \times 3/8 = 0.375 && \text{Mbps} \\ c_4(10) &= 10^6 \times 4/8 = 0.5 && \text{Mbps} \end{aligned}$$

Assuming no more sessions become backlogged, the completion times for the backlogged packets are

$$\begin{aligned} t_1 &= 0.5 \times 10^3/0.125 = 4 && \text{ms} \\ t_2 &= 10^3/0.375 \quad = 2.67 && \text{ms} \\ t_4 &= 7 \times 10^3/0.5 \quad = 14 && \text{ms} \end{aligned}$$

■

## 16.16 Fair Queuing

The Fair Queuing (FQ) algorithm proposed independently in [9] and [15] is completely equivalent to the Virtual Clock (VC) algorithm proposed by Zhang [10] in which the individual sessions are assigned separate queues. The queues are serviced in a round robin manner which prevents a source from arbitrarily increasing its share of the bandwidth. It is called "fair" because it allocates an equal share of the output bandwidth to each traffic flow or queue.

Figure 16.5 schematically shows queue serving sequence in FQ. In the figure it was assumed that the incoming flows are divided among $m$ queues. Fair queuing is used on a per-flow basis. Note, however, that the algorithm works on a packet-by-packet basis with no consideration to the separate end-to-end connections carried in each flow.

Assume flow $i$ has an arrival rate $\lambda_i$. The bandwidth allocated to flow $i$ is determined according to max–min scheduling strategy discussed in Sect. 16.13.

$$\lambda_i' = \min(\lambda_i, f) \tag{16.37}$$

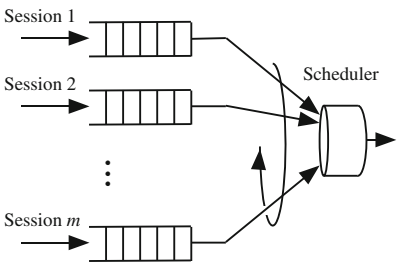



**Fig. 16.5** Fair queuing scheduling in which each queue shares an equal portion of server bandwidth in a round robin fashion

where $f$ is the *fair rate* assigned by the algorithm to each flow as follows.

$$f = \frac{C}{K} \tag{16.38}$$

where $C$ is the outgoing link rate, and $K$ is the number of backlogged sessions. $f$ is calculated such that when the switch is congested, the aggregate flow rate equals the switch capacity $C$.

Equation (16.37) indicates that $f$ is calculated recursively by removing the user with the minimal $\lambda_i$ and reducing the link capacity accordingly

$$C \leftarrow C - \lambda_{\min} \tag{16.39}$$

*Example 16.5.* Assume four sessions are being served by a fair queuing scheduler. The system parameters (in units of Mbps) are as follows:

$$C = 20$$
$$\lambda_1 = 1$$
$$\lambda_2 = 3$$
$$\lambda_3 = 8$$
$$\lambda_4 = 10$$

Find the rates assigned to each flow.

The sum of the flows due to all backlogged sessions is

$$1 + 3 + 8 + 10 = 22$$

which is larger than the link capacity. The initial fair share $f$ is given by

$$f = 20/4 = 5$$

Flow 1 has the minimal rate and is assigned the flow

$$\lambda_1' = \min(1, 5) = 1$$

We adjust the link capacity shared among the remaining three users as

$$C = 20 - 1 = 19$$

The fair share becomes

$$f = 19/3 = 6.33$$

Flow 2 has minimal rate and is assigned the flow

$$\lambda_2' = \min(3, 56.33) = 3$$

We adjust the link capacity shared among the remaining two users as

$$C = 19 - 3 = 16$$

The fair share becomes

$$f = 16/2 = 8$$

The bandwidths assigned to the flows become

$$\lambda_1' = 1$$
$$\lambda_2' = 3$$
$$\lambda_3' = 8$$
$$\lambda_4' = 8$$

The sum of all the assigned rates equals the outgoing link capacity. ■

Greedy flows that exceed the fair rate will have similar flow rate at the output equal to the fair rate assigned by the scheduler and will succeed only in filling their buffer which increases their cell loss probability.

Fair queuing is not completely satisfactory because it does not distinguish among long vs. short queues or high-priority vs. low-priority queues. Thus when bursty traffic is encountered, some queues will become full and their packets will be lost even if some of the other queues are far from full.

Switches or routers employing FQ have to classify each incoming packet to assign it to the proper queue. Furthermore, the switch or router has to perform some operations on each queue to determine its instantaneous flow $\lambda_i$.

## 16.17 Packet-by-Packet GPS

Packet-by-packet GPS (PGPS) is a packetized approximation of the GPS algorithms for fluid flow [34, 35]. This algorithm is also known as WFQ. Thus in PGPS data is served in terms of complete packets and only one packet can be served at a time.

A PGPS/WFQ server works on incoming flows in a static priority fashion based on a *timestamp* calculation. The flows or sessions are assigned separate queues based on packet header information. The scheduler scans the backlogged queues or sessions to select a packet for service. The packet with the highest priority (least timestamp) is selected and the output link capacity is dedicated to sending that

packet without sharing the resource with any other queued packets. After a packet has moved out of a FIFO queue, all packets in that queue move ahead by one position and the HOL packet enters the pool of selection from among all other HOL packets belonging to other sessions.

PGPS requires the computation of three quantities:

Virtual time $V(t)$: Indicates the share of the outgoing link capacity for each backlogged session.

Finish number $F_i$: Determines the priority of serving the packet in flow $i$. The packet with the least finish number is the one that will be served by the scheduler.

Completion time $T_i$: Determines the service time required by the packet based on the packet length and outgoing link bandwidth.

### *16.17.1 Virtual Time Calculation for PGPS/WFQ*

Assuming $B(t)$ to the set of backlogged sessions at time $t$, the *virtual time* $V(t)$ is defined using the differential equation

$$V(0) = 0 \tag{16.40}$$

$$\frac{dV(t)}{dt} = \frac{1}{\sum_{i \in B} w_i} \tag{16.41}$$

where $w_i \geq 1$ is the weight assigned to session $i$. Figure 16.6 shows the time development of $V(t)$ as packets arrive and sessions become backlogged.

When all sessions are idle, the virtual time is reset to zero

$$V(t) = 0 \qquad \text{when all sessions are idle} \tag{16.42}$$

*Example 16.6.* Assume a PGPS/WFQ scheduler serving flows of equal weights. The outgoing link capacity is 1 Mbps and all arriving packets equal lengths of 1 kb.

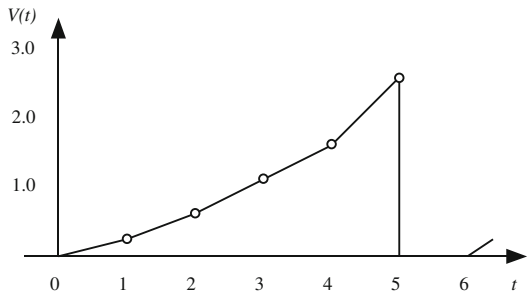



**Fig. 16.6** Development of the virtual time $V(t)$ in PGPS/WFQ scheduling

The packet arrival pattern is shown below where it is assumed that packets arrived at the idle sessions.

| $t$ (ms) | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Arrivals | 4 | | | 1 | | | 2 |

Determine the number of backlogged sessions vs. $t$ and obtain the values for $V(t)$.

A packet will require 1 ms to be transmitted.

At $t = 0$, four packets arrived and the number of backlogged sessions is $m(0) = 4$.

At $t = 1$, no packets arrive and the number of active sessions is reduced by one. Therefore, $m(1) = 3$ since one packet is guaranteed to be serviced.

At $t = 2$, no packets arrive and the number of active sessions is decremented by one $m(2) = 2$.

At $t = 3$, one packet arrives and one packet leaves which leaves $m(3) = m(2) = 2$.

At $t = 4$, no packets arrive and the number of active sessions is decremented by one.

At $t = 5$, no packets arrive and the number of active sessions is decremented by one and we get $m(5) = 0$.

At $t = 6$, two packets arrive and $m(6) = 2$.

The following table shows the development of $m(t)$.

| $t$ (ms) | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Arrivals | 4 | 0 | 0 | 1 | 0 | 0 | 2 |
| $m(t)$ | 4 | 3 | 2 | 2 | 1 | 0 | 2 |

We use (16.41) to determine the value of $V(t)$ at each time. ■

| $t$ (ms) | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Arrivals | 4 | 0 | 0 | 1 | 0 | 0 | 2 |
| m(t) | 4 | 3 | 2 | 2 | 1 | 0 | 2 |
| $dV(t)/dt$ | 1/4 | 1/3 | 1/2 | 1/2 | 1/1 | 0 | 1/2 |
| $V(t)$ | 0.0 | 0.25 | 0.58 | 1.08 | 1.58 | 2.58 | 0 |

### 16.17.2 Finish Number Calculation for PGPS/WFQ

Assuming packet $k$ has arrived at the queue for session $i$, then the finish number for that packet is calculated as

$$F_i(k) = \max\left[F_i(k-1), V(t)\right] + \frac{A_i}{w_i} \tag{16.43}$$

where $F_i(k-1)$ is the finish number for preceding packet in queue $i$ and $A_i$ is the length of the arriving packet. An empty queue will have a zero finish number associated with it.

The first term on the R.H.S. ensures that for a backlogged queue an arriving packet will have bigger finish number compared to packets already in queue $i$ so that each queue functions as FIFO.

The second term on the R.H.S. ensures that the finish number for packet $k$ of queue $i$ takes into account the size of that packet and the weight associated with the session. This ensures that sessions with lots of bits to send are slightly penalized to ensure fairness to other users that might have smaller packets to send.

Notice that a long packet or a greedy session will be characterized by large finish numbers and will receive lower service priority. On the other hand, short packets belonging to a conforming session with short queue will by characterized by small finish numbers and will be served more frequently. A greedy or bursty user traffic will only succeed in filling its buffer and losing packets while other users will still access their fair share of the link capacity.

The scheduler orders all the finish numbers for the packets at the head of all the queues. The packet with the least finish number is served first.

One last remark is worth making here. The finish number equation (16.43) guarantees that packets already in the system at time $t$ will be served *before* any packets that arrive after $t$.

When session $i$ is idle, its finish number is reset to zero

$$F_i = 0 \qquad \text{when session is idle} \tag{16.44}$$

### 16.17.3 Completion Time Calculation for PGPS/WFQ

The completion time for a packet is simple to calculate in PGPS/WFQ since the outgoing resources are completely dedicated to the selected packet.

Assuming packet $k$ in session $i$ has a length $A_k$, then the time $T$ required to transmit it is given by

$$T = \frac{A_k}{C} \qquad \text{s} \tag{16.45}$$

where $C$ is the outgoing link capacity in bps.

*Example 16.7.* Assume a PGPS/WFQ scheduler serving four flows of equal weights. The outgoing link capacity is $C = 1$ Mbps and the packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1 | 3 | | 2 | |
| Session 2 | | 1 | | 4 |
| Session 3 | 2 | | 7 | |
| Session 4 | | | | 5 |

where the numbers in the rows of each session indicate the length of the packet in units of kb. Calculate the system virtual time and finish numbers for the arriving packets of each session.

A packet of unit length (1 kb) takes 1 ms to transmit. At the start virtual time and all the finish numbers are reset to 0.

At $t = 0$ our table will be

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | 3 | | 2 | |
| Session 2 | | 1 | | 4 |
| Session 3* | (2) | | 7 | |
| Session 4 | | | | 5 |
| $dV/dt$ | 0.5 | | | |
| $V(t)$ | 0.0 | | | |

Sessions 1 and 3 are backlogged, as indicated by the asterisk (*) and the finish numbers for the arriving packets are

$$F_1(1) = \max(0, 0) + 3 = 3$$
$$F_3(1) = \max(0, 0) + 2 = 2$$

Packet $p_3(1)$ will be served first and will require 2 ms to transmit. This is indicated by the brackets round the entry for this packet.

At $t = 1$ our table will be

The finish numbers for all congested sessions (i.e., 1, 2, and 3) are given by

$$F_1(1) = 2$$
$$F_2(1) = \max(0, 0.5) + 1 = 1.5$$
$$F_3(1) = 3$$

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | 3 | | | |
| Session 2* | | 1 | | 4 |
| Session 3* | (2) | | 7 | |
| Session 4 | | | | 5 |
| $dV/dt$ | 0.5 | 0.3 | | |
| $V(t)$ | 0.0 | 0.5 | | |

Since $F_2(1)$ has the least finish number, we could have chosen the packet in session 2 for transmission. However, this would mean that we stop the transmission of packet in session 3 which is not finished yet. This is a form of *preemptive scheduling*. We choose a nonpreemptive scheduling scheme and continue transmitting packet out of session 3 as indicated by the brackets surrounding the packet of session 3.

At $t = 2$ our table will be

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | 3 | | 2 | |
| Session 2* | | (1) | | 4 |
| Session 3* | 2 | | 7 | |
| Session 4 | | | | 5 |
| $dV/dt$ | 0.5 | 0.3 | 0.3 | |
| $V(t)$ | 0.0 | 0.5 | 0.8 | |

Packet $p_2(1)$ is chosen for transmission since it has the lowest finish number among all the backlogged sessions. The finish numbers for the new packets are

$$F_1(2) = \max(3, 0.8) + 2 = 5$$

$$F_3(2) = \max(2, 0.8) + 7 = 9$$

Since finish number for packet in session 2 is 1.5, we choose this packet for transmission as indicated by the brackets surrounding that packet.

At $t = 3$ our table will be

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | (3) | | 2 | |
| Session 2* | | 1 | | 4 |
| Session 3* | 2 | | 7 | |
| Session 4* | | | | 5 |
| $dV/dt$ | 0.5 | 0.3 | 0.3 | 0.25 |
| $V(t)$ | 0.0 | 0.5 | 0.8 | 1.1 |

The finish numbers for the new packets are

$$F_2(2) = \max(1.5, 1.1) + 4 = 5.5$$
$$F_4(1) = \max(0, 1.1) + 5 = 6.1$$

Since the finish number for the HOL packet in session 1 is 3, we pick this packet for transmission. ■

## 16.18 Frame-Based Fair Queuing

Frame-based fair queuing (FFQ) belongs to the general class of rate-proportional servers (RPS) proposed in [36]. This type of schedulers are claimed to offer similar delay and fairness bounds as PGPS/WFQ but with much simpler computations of the packet priorities.

FFQ server works on incoming flows in a static priority fashion based on a *timestamp* calculation. The flows or sessions are assigned separate queues based on packet header information. The scheduler scans the backlogged queues or sessions to select a packet for service. The packet with the highest priority (least timestamp) is selected and the output link capacity is dedicated to sending that packet without sharing the resource with any other queued packets. After a packet has moved out of a FIFO queue, all packets in that queue move ahead by one position and the HOL packet enters the pool of selection from among all other HOL packets belonging to other sessions.

FFQ requires the computation of three quantities:

System potential $P(t)$: Indicates the amount of data transferred through the outgoing link up to time $t$.

Timestamp $S_i$: Determines the priority of serving the packet in flow $i$. The packet with the least timestamp is the one that will be served by the scheduler.

Completion time $T_i$: Time required by the packet to be fully transmitted. It depends on the packet length and outgoing link bandwidth.

### *16.18.1 System Potential Calculation for FFQ*

Assume that the server started serving a packet at time $t_s$. At a later time $t \geq t_s$, the *system potential* $P(t)$ is defined as

$$P(t) \leftarrow P + \frac{C(t - t_s)}{F} \tag{16.46}$$

where $C$ (bps) is the outgoing link capacity and $F$ is the *frame size* in bits. The system potential $P(t)$ measures the amount of data transferred up to time $t$ relative to the frame size $F$. The system potential is updated each time a packet starts service and when all sessions are idle, the system potential is reset to zero.

$$P(t) = 0 \qquad \text{when all sessions are idle} \tag{16.47}$$

The frame size $F$ is chosen so that at least the maximum length packet from any session can be sent during one frame period; i.e.,

$$F > A_{\max} \qquad \text{bits} \tag{16.48}$$

The *frame period* $T$ corresponding to the chosen frame size is given by

$$T = \frac{F}{C} \qquad \text{s} \tag{16.49}$$

### 16.18.2 Timestamp Calculation for FFQ

When a $k$ packet arrives at session $i$, a timestamp is associated with it according to the following formula

$$S_i(k) = \max\left[S_i(k-1), P(t)\right] + \frac{A_i}{\lambda_i} \tag{16.50}$$

where $S_i(k-1)$ is the timestamp of the previous packet in the queue, $A_i$ is the packet length in bits, and $\lambda_i$ is the reserved rate for session $i$. An empty queue will have a zero timestamp associated with it.

A long packet will be penalized by having a large timestamp value and users with higher reserved bandwidth will consistently receive lower timestamp values so as to obtain their reserved rate in a fair manner.

One issue remains to be resolved which is determining when the current frame is to be completed and a new frame is to be started. We mentioned above that a new frame starts when all sessions are idle. When one or more sessions are backlogged, a new frame is started when the accumulated bits transferred approaches the frame size $F$. To keep track of the number of bits transferred, a bit counter could be used. When a packet is selected for transmission the counter contents are updated

$$B(k) = B(k-1) + A_k \tag{16.51}$$

If $B(k) \leq F$, the current frame is continued and the packet is sent. If $B(k) > F$, a new frame is started and the packet sent during the new frame.

### 16.18.3 Completion Times Calculation for FFQ

The completion time for a packet is simple to calculate in FFQ since the outgoing resources are completely dedicated to the selected packet.

Assuming packet $k$ in session $i$ has a length $A_k$, then the time $T$ required to transmit it is

$$T = \frac{A_k}{C} \qquad \text{s} \tag{16.52}$$

where $C$ is the outgoing link capacity in bps.

*Example 16.8.* Assume an FFQ scheduler serving flows of equal weights. The outgoing link capacity is 1 Mbps and the frame size is chosen as $F = 10^4$ bits. The packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1 | 3 | | 2 | |
| Session 2 | | 1 | | 4 |
| Session 3 | 2 | | 7 | |
| Session 4 | | | | 5 |

where the numbers in the rows of each session indicate the length of the packet in units of kb. Calculate the system potential and timestamps for the arriving packets of each session.

A packet of unit length (1 kb) takes 1 ms to transmit. At the start system potential and all the timestamps are reset to 0.

At $t = 0$ our table will be

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | 3 | | 2 | |
| Session 2 | 0 | 1 | | 4 |
| Session 3* | (2) | | 7 | |
| Session 4 | 0 | | | 5 |
| $B(t)$ | 0 | | | |
| $P(t)$ | 0.0 | | | |

where the session entries at $t = 0$ are *timestamp values*. Sessions 1 and 3 are backlogged, as indicated by the asterisk (*) and the timestamp values are

$$S_1(1) = \max(0, 0) + 3 = 3$$

$$S_3(1) = \max(0, 0) + 2 = 2$$

Packet $p_3(1)$ will be served first and will require 2 ms to transmit. This is indicated by the brackets round the entry for this packet.

At $t = 1$ our table will be

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | 3 | | | |
| Session 2* | | 1 | | 4 |
| Session 3* | (2) | | 7 | |
| Session 4 | | | | 5 |
| $B(t)$ | 0 | 1 | | |
| $P(t)$ | 0.0 | 0.1 | | |

where the entry for $B(t)$ is in kbits. The timestamp for $p_2(1)$ is

$$S_2(1) = \max(0, 0.1) + 1 = 1.1$$

At $t = 2$ our table will be

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | 3 | | 2 | |
| Session 2* | | (1) | | 4 |
| Session 3* | 2 | | 7 | |
| Session 4 | | | | 5 |
| $B(t)$ | 0 | 1 | 2 | |
| $P(t)$ | 0.0 | 0.1 | 0.2 | |

Packet $p_2(1)$ is chosen for transmission since it has the lowest timestamp among all the backlogged sessions. The timestamps for the new packets are

$$S_1(2) = \max(3, 0.2) + 2 = 5$$
$$S_3(2) = \max(2, 0.2) + 7 = 9$$

At $t = 3$ our table will be

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1* | (3) | | 2 | |
| Session 2* | | 1 | | 4 |
| Session 3* | 2 | | 7 | |
| Session 4* | | | | 5 |
| $B(t)$ | 0 | 1 | 2 | 3 |
| $P(t)$ | 0.0 | 0.1 | 0.2 | 0.3 |

The timestamps for the new packets are

$$S_2(2) = \max(1.1, 0.3) + 4 = 5.1$$
$$S_4(1) = \max(0, 0.3) + 5 = 5.3$$ ■

## 16.19 Core-Stateless Fair Queuing

The problem with the schedulers discussed so far is the need to maintain a separate state for each flow at all routers in the path of the packets. Such schedulers allow the system to provide firm QoS guarantees for each flow. However, they are complex to implement and their performance is limited by the number of flows that can be supported [37].

Core-stateless fair queuing (CSFQ) attempts to simplify matters by dividing the routers in the network into two categories: edge routers and core routers as shown in Fig. 16.7 [37].

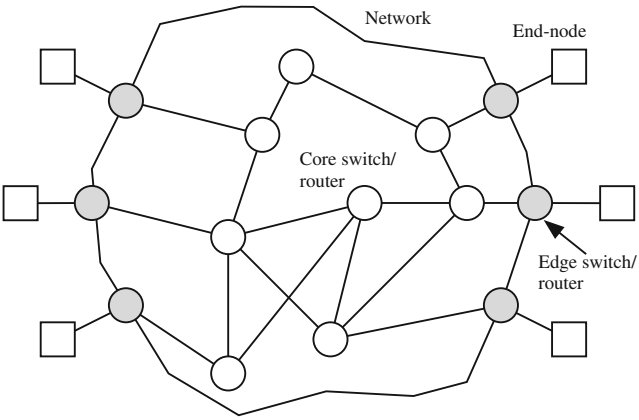



**Fig. 16.7** Routers in core-stateless fair queuing (CSFQ) are divided into edge routers (*grey circles*) and core routers (*empty circles*). End nodes (*squares*) are connected to edge routers

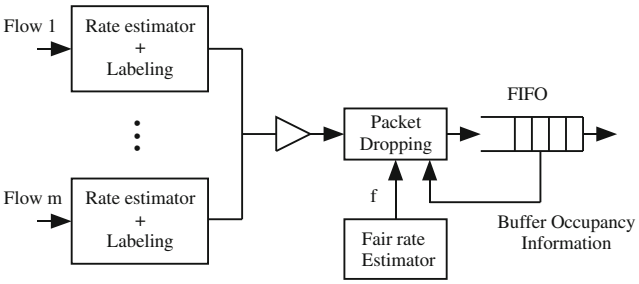



**Fig. 16.8** The architecture of an edge router implementing CSFQ scheduling

Edge routers maintain a per-flow state and label incoming packets accordingly. They also regulate the incoming flows such that flow $i$ receives a fair service rate $\lambda_i$ determined by

$$\lambda_i = \min[\lambda_i, f] \tag{16.53}$$

where $\lambda_i$ is the arrival rate for flow $i$ and $f$ is the fair share rate determined in the following section.

Figure 16.8 shows the functions performed by each edge router. The rate of each incoming flow is estimated based on the timing of arriving packets. The estimated arrival rates for all incoming traffic are used to obtain an estimated value for the fair share $f$. The edge router also decides whether to accept or drop the arriving packet based on the arrival rate and the fair share estimate as discussed below. The figure shows that the packet drop probability depends both on the arrival rate, estimated fair share, and the state of the FIFO buffer occupancy.

Core routers do not maintain a per-flow state but use the per-flow label in each packet to decide whether to drop an incoming packet or to accept it. The probability that a packet is dropped is calculated by each core router based on the packet label and on the estimated fair rate at the core router. Accepted packets are placed in a simple FIFO buffer for transmission.

Figure 16.9 shows the functions performed by each core router. The rate of each incoming flow is extracted from the header of arriving packets. The arrival rates for all incoming traffic are used to obtain an estimated value for the fair share $f$. The core router also decides whether to accept or drop the arriving packet based on the arrival rate and the fair share estimate as discussed below. The figure shows that the packet drop probability depends both on the arrival rate, estimated fair share, and the state of the FIFO buffer occupancy.

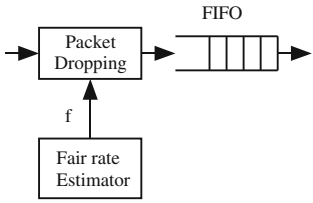



**Fig. 16.9** The architecture of a core router implementing CSFQ scheduling

### *16.19.1 Determination of Packet Arrival Rate $\lambda_i$*

Central to CSFQ is obtaining an accurate estimate of the average packet arrival rate for each flow. The arrival rate $\lambda_i$ is estimated in an iterative manner each time a new packet arrives:

$$\lambda_i(k) = (1-\alpha)\,\frac{A_i(k)}{T_i(k)} + \alpha\,\lambda_i(k-1) \tag{16.54}$$

$$\alpha = \exp -T_i(k)/K \tag{16.55}$$

where $T_i(k)$ is the packet interarrival time for flow $i$ at time $k$, $K$ is some constant, and $A_i(k)$ is the length of the arriving packet.

A conforming user will have a high value for the interarrival time $T_i(k)$. This will result in an exponentially low value for $\alpha$ which increases the assigned rate $\lambda_i(k)$.

### *16.19.2 Determination of Fair Rate $f$*

The fair share $f$ is determined based on whether the link is congested or uncongested. The update operation is performed at regular intervals of time determined by the system administrator.

The link is congested when the total arrival rate exceeds the outgoing link capacity $C$

$$C \le \sum_{i=1}^{m} \min(\lambda_i, f) \qquad \text{link congested} \tag{16.56}$$

and in that case the fair share is set equal to its old value

$$f(k) = f(k-1) \tag{16.57}$$

The link is uncongested when the total arrival rate is smaller than the outgoing link capacity $C$

$$C > \sum_{i=1}^{m} \min(\lambda_i, f) \qquad \text{link uncongested} \tag{16.58}$$

and in that case the fair share is determined as follows

$$f(k) = f(k-1)\frac{C}{\max(\lambda_i)} \tag{16.59}$$

The estimated fair share also is slightly modified based on the level of occupancy of the FIFO buffer but this is a minor detail that the reader could check in reference [37].

### *16.19.3 Bit Dropping Probability $p_b$*

When the traffic rate for flow $i$ is such that $\lambda_i < f$, then that session is conforming and no bit dropping is necessary. The packets of that flow can pass through the network.

A misbehaving session $i$ is characterized by $\lambda_i > f$ and the packet must be dropped with a drop probability given by

$$p_i = \max\left(0, 1 - \frac{f}{\lambda_i}\right) \tag{16.60}$$

## 16.20 Random Early Detection

The schedulers we discussed in the previous sections emphasized techniques for selecting the next packet for service. Selecting which packet to drop when the buffer overflows was simple. The last packet to arrive is dropped when the buffer is full. This type of packet drop is called *tail drop* strategy. To achieve max–min buffer sharing, the scheduler must assign a buffer to each service class or each user.

Random early drop detection (RED) belongs to the class of *early drop schedulers* where a packet is dropped even when the buffer is not full. There are several plausible reasons why early dropping of packets is beneficial:

1. In tail drop schedulers misbehaving users occupy precious buffer space and packets belonging to conforming users would be dropped if they arrive when the buffer is full.
2. Dropping packets from some sessions or classes sends a message to end-points to reduce their rate before the network becomes congested and more packets would then be lost.

In RED the switch calculates the average queue size at each time step. Based on this estimate, the switch decides whether to drop the packet or label it with a probability that is a function of the queue size [38]. The switch calculates the average queue size $Q_a$ using a low-pass filter. Calculating an average queue length based on filtering is better than using the instantaneous queue length since this allows small temporary traffic bursts to go through unharmed.

The average queue size is compared to two thresholds $Q_{min}$ and $Q_{max}$.

1. When $Q_a < Q_{min}$, packets are not marked.
2. When $Q_a > Q_{max}$, all packets are marked.
3. When $Q_{min} \leq Q_a \leq Q_{max}$, packets are marked with a probability $p_a$ given by

$$p_a = \frac{p_b}{1 - Q\ p_b} \tag{16.61}$$

where $Q$ is the number of packets received since the last marked packet and $p_b$ is given by

$$p_b = \frac{Q_a - Q_{min}}{Q_{max} - Q_{min}} \tag{16.62}$$

One of the problems of RED is the amount of operations that have to be done each time step on each packet. This is one reason why FIFO with tail drop method is still in use.

Another related algorithm that is similar to RED is flow random early drop (FRED) where the switch drops each arriving packet with a fixed drop probability $p_d$ when the queue size exceeds a certain drop threshold. FRED is classified as a *early random drop* algorithm where arriving packets are randomly dropped. The reasoning being that misbehaving users send more packets and will have a higher probability that their packets are dropped. However, it has been shown that this drop policy is not very successful [10].

## 16.21 Packet Drop Options

We discussed above two advantages for dropping packets in a router. Packets are dropped to reduce network congestion and to improve its efficiency [39]. There are several options for *selecting* the next packet to drop and for determining the *times* when the drop policies are implemented. Below we discuss some of the packet drop policies that could be implemented alone or in combinations, depending on the transmission and scheduler protocols being used.

The simplest packet drop policy is to drop incoming packets when the shared buffer or queue is full. This drop policy does not offer protection against misbehaving users. A full buffer most probably has been caused by a misbehaving user and the packets dropped might belong to conforming users. Of course such simple policy does not require maintaining any state information. There is only one

state to maintain here which is the level of occupancy of the buffer. Attempting to protect conforming users requires defining state variables for each user indicating the amount of packets present in the system and belonging to each session. During periods of congestion, users that have high buffer occupancy states are eligible to have their packets dropped. While this offers protection against misbehaving users, the system must maintain many state variables.

An intermediate solution is to group or aggregate the users into classes of service and maintain state information for these classes only. This solution offers some protection and isolation and also reduces the amount of state variables required.

If the scheduler implements PGPS/WFQ or FFQ algorithms, then a simple packet drop strategy is to drop the packet with the highest finish number or timestamp value. The reasoning for this is that large values for these parameters indicate either long packets or many packets belonging to this session.

Sometimes the packet header contains information that can help with the packet drop policy. For example in ATM the AAL layer contains information about missing cells. When a switch or router detects that a connection has a missing cell, it drops all subsequent cells until the last cell of the frame [39]. This frees the network from supporting a cell stream that will be discarded by the receiver since the frame will be retransmitted anyway.

## 16.22 Problems

### *Scheduler Functions*

**16.1.** Explain the main functions performed by a scheduler.

**16.2.** What are the main switch resources shared among the different users?

**16.3.** What are the scheduler performance measures most important to the following applications: electronic mail, file transfer, web browsing, voice communications, and one-way video?

**16.4.** Which is the important QoS parameter for best-effort applications and CBR traffic?

**16.5.** Explain the functions performed by the scheduler at the different contention points for input, output, and shared buffer switches.

**16.6.** One solution to solve the HOL problem in input queued switches is to use VOQ. Explain how VOQ works then explain how the scheduler will work in such a scheme.

**16.7.** Explain what is meant by fairness and what is meant by protection from a scheduler perspective.

**16.8.** It was explained that to provide deterministic QoS guarantees the scheduler must maintain separate queues for the separate sessions. Explain how this can be done in input, output, and shared buffer switches. Discuss the pros and cons of each scheme from the point of view of the implementation of the scheduler in each case.

## *Scheduler Performance Measures*

**16.9.** Explain what is meant by protection in terms of outgoing link bandwidth utilization.

**16.10.** Explain what is meant by protection in terms of switch buffer utilization.

**16.11.** Investigate how a scheduler might be able to reduce delay jitter for the different sessions.

## *Scheduler Classifications*

**16.12.** Explain the difference between work-conserving and non-work-conserving schedulers.

**16.13.** Explain what is meant by degree of aggregation and the advantages and disadvantages of this strategy.

**16.14.** Explain the different packet drop policies that could be used by schedulers.

## *Max–Min Fairness*

**16.15.** Explain max–min fairness as it applies to outgoing link bandwidth.

**16.16.** Explain max–min fairness as it applies to outgoing shared buffer space in a switch.

**16.17.** Assume an outgoing link is being shared among six channels. The system parameters (in units of Mbps) are as follows:

$$C = 622$$
$$\lambda_1 = 200$$
$$\lambda_2 = 30$$
$$\lambda_3 = 100$$
$$\lambda_4 = 50$$

$$\lambda_5 = 180$$

$$\lambda_6 = 180$$

Find the rates assigned to each flow according to the max–min algorithm.

**16.18.** Assume a 1,000 byte buffer is being shared among five sessions. The buffer requirements (in units of bytes) for each session are as follows:

$$B_1 = 250$$

$$B_2 = 250$$

$$B_3 = 300$$

$$B_4 = 400$$

$$B_5 = 150$$

Find the buffer space assigned to each flow according to the max–min algorithm.

## *FIFO (or FCFS) Scheduling*

**16.19.** Assume a FIFO scheduler where the output link rate is $C$ and the arrival rate for each session is $\lambda$, the number of arriving sessions is $m$ and all flows have equal packet lengths $L$. Find the performance of the system assuming a fluid flow model.

**16.20.** Assume a FIFO scheduler where there are $m$ users accessing the buffer but one of the users has an arrival probability that is different from that of the other users. Derive the transition matrix for such a system.

**16.21.** Assume a FIFO scheduler where there are $m$ users accessing the buffer but one of the users produces packets with different length compared to those of the other users. Derive an expression for the average length of the queue and the average queuing delay of such a system.

## *Static Priority Scheduling*

**16.22.** Consider queue $i$ in the static priority scheduler. Assume the arrival probability for this queue $a_i$ and its priority is $i$, where lower values of $i$ indicate higher priority. Write down the transition matrix for this queue and comment on methods to find its steady state distribution vector.

**16.23.** Repeat Problem 16.22 for the case when all queues have the same size $B$ and have the same arrival probability $a$.

**16.24.** Consider a static-priority protocol serving four users where the packet arrival probabilities for all users are equal (i.e., $a_i = 0.3$ for all $1 \leq i \leq 4$) and all users have the same buffer size (i.e., $B_i = 8$ for all $1 \leq i \leq 4$). Estimate the performance of each user.

**16.25.** Consider a static-priority protocol serving four users where the packet arrival probabilities for all users are equal (i.e., $a_i = 0.6$ for all $1 \leq i \leq 4$) and all users have the same buffer size (i.e., $B_i = 4$ for all $1 \leq i \leq 4$). Estimate the performance of each user.

**16.26.** Repeat Problem 16.25 when the probability of the queue being empty becomes high $e = 0.9$

## *Round Robin Scheduler*

**16.27.** Assume a round robin scheduler in which all packets have equal lengths. Obtain expressions for the maximum scheduler delay and the fraction of the bandwidth assigned to any session.

**16.28.** Assume a round robin scheduler in which all queues have identical traffic characteristics (arrival probability, packet length, etc.). Obtain the transition matrix for one queue and obtain expressions for its performance parameters: queue length, throughput, and loss probability.

**16.29.** Assume a round robin scheduler in which all sessions are identical with the following parameters:

$$m = 8 \text{ sessions } C = 10 \text{ Mbps}$$
$$B = 8 \text{ packets } L = 512 \text{ bits}$$
$$\lambda = 100 \text{ kbps } \sigma = 500 \text{ kbps}$$

Determine the transition matrix and determine the system performance parameters.

## *GPS*

**16.30.** Assume a GPS scheduler serving four flows with associated weights $w_1 = 4$, $w_2 = 2$, $w_3 = 3$, and $w_4 = 1$. The outgoing link capacity is 10 Mbps and the packet arrival pattern in the different flows is as shown below.
where the numbers in the rows of each session indicate the length of the packet that arrived at that time in units of kb. Calculate the assigned bandwidth and completion times for the arriving packets.

| $t$ (ms) | 0 | 1 | 2 |
|---|---|---|---|
| Session 1 | 3 | | 2 |
| Session 2 | | 1 | 3 |
| Session 3 | 2 | | |
| Session 4 | | 7 | |

**16.31.** What is the longest delay bound experienced by the packet in session $i$ in GPS?

## *Fair Queuing*

**16.32.** Assume four sessions are being served by a fair queuing scheduler. The system parameters (in units of Mbps) are as follows:

$$C = 40$$
$$\lambda_1 = 6$$
$$\lambda_2 = 2$$
$$\lambda_3 = 20$$
$$\lambda_4 = 16$$

Find the rates assigned to each flow.

## *Packet-by-Packet GPS/Weighted Fair Queuing*

**16.33.** Assume a PGPS/WFQ scheduler serving flows of equal weights. The outgoing link capacity is 1 Mbps and all arriving packets equal lengths of 1 kb. The packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Arrivals | 2 | | | 2 | | 4 | |

Determine the number of backlogged sessions vs. $t$ and obtain the values for $V(t)$.

**16.34.** Assume a PGPS/WFQ scheduler serving flows of equal weights. The outgoing link capacity is 1 Mbps and the packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1 | 4 | | 2 | |
| Session 2 | | 2 | | 3 |
| Session 3 | 1 | | 5 | |
| Session 4 | | | | 4 |

where the numbers in the rows of each session indicate the length of the packet in units of kb. Calculate the system virtual time and finish numbers for the arriving packets of each session.

**16.35.** Assume a PGPS/WFQ scheduler serving flows of equal weights. The outgoing link capacity is 1 Mbps and the packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1 | | | 2 | |
| Session 2 | 3 | 2 | 1 | |
| Session 3 | | 1 | | 2 |
| Session 4 | 1 | | 5 | |

where the numbers in the rows of each session indicate the length of the packet in units of kb. Calculate the system virtual time and finish numbers for the arriving packets of each session.

**16.36.** Assume a PGPS/WFQ scheduler serving flows of equal weights. The outgoing link capacity is 2 Mbps and the packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1 | 7 | | 2 | |
| Session 2 | | 3 | | 2 |
| Session 3 | 3 | | 7 | |
| Session 4 | | | | 8 |

where the numbers in the rows of each session indicate the length of the packet in units of kb. Calculate the system virtual time and finish numbers for the arriving packets of each session.

## *FFQ*

**16.37.** Assume an FFQ scheduler serving flows of equal weights. The outgoing link capacity is 1 Mbps and the frame size is chosen as $F = 10^4$ bits. The packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1 | 1 | | 2 | |
| Session 2 | | 1 | | 4 |
| Session 3 | 2 | | 7 | |
| Session 4 | | | | 5 |

where the numbers in the rows of each session indicate the length of the packet in units of kb. Calculate the system potential and timestamps for the arriving packets of each session.

**16.38.** Assume an FFQ scheduler serving flows of equal weights. The outgoing link capacity is 1 Mbps and the frame size is chosen as $F = 10^4$ bits. The packet arrival pattern is shown below.

| $t$ (ms) | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Session 1 | 2 | | 2 | |
| Session 2 | | 1 | | 4 |
| Session 3 | 1 | | 7 | |
| Session 4 | | | | 5 |

where the numbers in the rows of each session indicate the length of the packet in units of kb. Calculate the system potential and timestamps for the arriving packets of each session.

## References

1. N. McKeown, T.E. Anderson, A quantitative comparison of iterative scheduling algorithms for input-queued switches. Comput. Netw. ISDN Syst. **30**, 2309–2326 (1998)
2. T. Nasser, F. Gebali, A new scheduling technique for packet switching networks using the transportation problem optimization model, in *1st International Symposium on Signal Processing and Information Technology (ISSPIT 2001)*, pp. 360–363, Cairo, Egypt, Dec 2001
3. G.B. Dantzig, *Linear Programming* (Springer, New York, 1985)
4. S.S. Rao, *Engineering Optimization* (Princeton University Press, Princeton, 1996)
5. J.A. cobb, M.G. Gouda, A. El-Nahas, Time-shift scheduling—fair scheduling of flows in high-speed networks. IEEE/ACM Trans. Netw. **6**(3), 274–285 (1998)

6. K. Nichols, S. Blake, F. Baker, D.L. Blake, Definition of the differentiated services field (DS Field) in the IPv4 and IPv6 headers. Dec 1999, IETF RFC 2474
7. D. Stiliadis, A. Varma, Latency-rate servers: A general model for analysis of traffic scheduling algorithms. IEEE/ACM Trans. Netw. **6**(5), 611–624 (1995), p. 41
8. A.K. Parekh, R.G. Gallager, A generalized processor sharing approach to flow control - the single node case, in *Proceedings of IEEE INFOCOM*, vol. 2, pp. 915–924, May 1992
9. A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queuing algorithm, in *SIGCOM'89*, pp. 3–12
10. L. Zhang, Virtual clock: A new traffic control algorithm for packet switching networks. ACM Trans. Comput. Syst. **9**, 101–124 (1991)
11. M. Datevenis, S. Sidiropoulos, C. Courcoubetis, Weighted round-robin cell multiplexing in a general-purpose ATM switch chip. IEEE J. Sel. Areas Commun. **9**, 1265–1279 (1991)
12. D. Ferrari, D. Verma, A scheme for real-time channel establishment in wide-area networks. IEEE J. Sel. Areas Commun. **8**, 368–379 (1990)
13. M. Shreedhar, G. Varghese, Efficient fair queuing using deficit round robin, in *SIGCOMM'95*, Sep 1995
14. D.C. Verma, H. Zhang, D. Ferrari, Delay jitter control for real-time communication in a packet switching network, in *Proceedings of Tricomm'91*, pp. 35–43, Chapel Hill, North Carolina, Apr 1991
15. J. Nagle, On packet switches with infinite storage. IEEE Trans. Commun. **35**, 435–438 (1987)
16. C.R. Kalmanek, H. Kanakia, S. Keshav, Rate controlled servers for very high-speed networks, in *Conference on Global Communications (GLOBECOM)*, pp. 12–20, San Diego, CA, USA, Dec 2–5, 1990
17. S.J. Golestani, A framing strategy for congestion management. IEEE J. Sel. Areas Commun. **9**(7), 1064–1077 (1991)
18. S.J. Golestani, Congestion-free transmission of real-time traffic in packet networks, in *Proceedings of IEEE INFOCOM'90*, San Francisco, California, USA, pp. 527–536, June 1990
19. S.J. Golestani, A Stop-and-go queuing framework for congestion management, in *Proceedings of the ACM Symposium on Communications Architectures and Protocols (SIGCOMM'90)*, ACM Press, Philadelpia, PA, USA, vol. 20, pp. 8–18, Sep 24–27, 1990
20. S.J. Golestani, Duration-limited statistical multiplexing of delay-sensitive traffic in packet networks, in *Proceedings of the ACM Symposium on Communications Architectures and Protocols (SIGCOMM'91)*, ACM Press, pp. 323–332, Zurich, Switzerland, Sep 3–6, 1991
21. H. Zhang, D. Ferrari, Rate-controlled static-priority queueing", *Conference on Computer Communications (IEEE INFOCOM)*, San Francisco, CA, USA, pp. 227–236, Mar 28–Apr 1, 1993
22. M. Aras, J.F. Kurose, D.S. Reeves, H. Schulzrinne, Real-time communication in packet-switched networks. Proc. IEEE **82**(1), 122–139 (1994)
23. D. Ferrari, D.C. Verma, Scheme for real-time channel establishment in wide-area networks. IEEE J. Sel. Areas Commun. **8**(3), 368–379 (1990)
24. D.C. Verma, H. Zhang, D. Ferrari, Delay jitter control for real-time communications in a packet switching network, in *Proceedings of TRICOMM'91*, pp. 35–46, Chapel Hill, North Carolina, Apr 1991
25. D.D. Kandlur, K.G. Shin, D. Ferrari, Real-time communication in multi-hop networks, in *11th International Conference on Distributed Computing Systems (ICDCS)*, pp. 300–307, Arlington, TX, May 1991
26. H. Fattah, Frame-based fair queuing: analysis and design. Thesis, Department of Electrical and Computer Engineering, p. 111, University of Victoria, Victoria, B.C., 2000
27. D. Ferrari, Client requirements for real-time communication services. IEEE Commun. Mag. **28**(11), (1990) also RFC 1193
28. S.J. Glestani, Congestion-free transmission of real-time traffic in packet networks, in *ACM SIGCOMM'90*, pp. 527–542, June 1990
29. R. Guerin, V. Peris, Quality of service in packet networks: basic mechanisms and directions, in *Proceedings of INFOCOM'97*, vol. 31, pp. 169–189, 1999

30. F. Elguibaly, Analysis and design of arbitration protocols. IEEE Trans. Comput. **38**(2), 1168–1175 (1989)
31. L. Bic, A.C. Shaw, *The Logical Design of Operating Systems*, pp. 275–280 (1995), Prentice-Hall, New Jersey, USA
32. J. Crowcroft, J. Oechslin, Differentiated end-to-en internet services using a weighted proportional fair sharing TCP. ACM SIGCOMM **28**(3), 53–67 (1998)
33. D. Ferrari, D. Verma, A scheme for real-time channel establishment in wide-area networks. IEEE J. Sel. Areas Commun. **8**, 368–379 (1990)
34. S. Floyd, V. Jacobson, Link-sharing and resource management models for packet networks. IEEE/ACM Trans. Netw. **3**(4), 365–386 (1995)
35. M. Grossglauser, S. Keshav, D. Tse, RCBR: A simple and efficient service for multiple time-scale traffic, in *Proceedings of ACM SIGCOMM'95*, 1995
36. D. Stiliadis, A. Varma, Efficient fair queuing algorithms for packet-switched networks. IEEE/ACM Trans. Netw. **6**(2), 175–185 (1998)
37. I. Stoica, S. Shenkar, H. Zhang, Core-stateless fair queuing: A scalable architecture to approximate fair bandwidth allocation in high speed networks, in *Proceedings of SIGCOMM'98*, pp. 118–130
38. S. Floyd, V. Jackobson, Random early detection for congestion avoidance. IEEE/ACM Trans. Netw. **1**, 397–413 (1993)
39. S. Keshav, *An Engineering Approach to Computer Networking* (Addison Wesley, Reading, MA, 1997)

# Appendix A
# Series and Useful Formulas

## A.1 Series and Useful Formulas

$$\sum_{i=0}^{n-1}(a + i\,d) = \frac{n}{2}\,(a + l)$$

where $l = a + (n - 1)d$

## A.2 Geometric Series

$$\sum_{i=0}^{n-1} a\,r^i = \frac{a(1 - r^n)}{1 - r}$$

$$\sum_{i=0}^{\infty} a\,r^i = \frac{a}{1 - r}$$

where $r \neq 1$ in the above two equations.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6

## A.3 Arithmetic-Geometric Series

$$\sum_{i=0}^{n-1} (a + i\,d)\, r^i = \frac{a(1 - r^n)}{1 - r} + \frac{rd\left[1 - n r^{n-1} + (n-1)\, r^n\right])}{(1-r)^2}$$

where $r \neq 1$ in the above two equations. If $-1 < r < 1$, the series converge and we get

$$\sum_{i=0}^{\infty} (a + i\,d)\, r^i = \frac{a}{1 - r} + \frac{rd}{(1-r)^2}$$

$$\sum_{i=0}^{\infty} i^2\, r^i = \frac{r + r^2}{(1-r)^3}$$

## A.4 Sums of Powers of Positive Integers

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \tag{A.1}$$

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \tag{A.2}$$

$$\sum_{i=1}^{n} i^3 = \frac{n^2(n+1)^2}{4} \tag{A.3}$$

## A.5 Binomial Series

$$\sum_{i=0}^{n} \binom{n}{i} a^{n-i} b^i = (a + b)^n$$

special cases

$$\sum_{i=0}^{n} \binom{n}{i} = 2^n$$

$$\sum_{i=0}^{n}(-1)^i\binom{n}{i}=0$$

$$\sum_{i=0}^{n}\binom{n}{i}^2=\binom{2n}{n}$$

$$(1+x)^q=1+qx+\frac{q(q-1)}{2!}x^2+\frac{q(q-1)(q-2)}{3!}x^3\cdots$$

$$(1+x)^{-1}=1-x+x^2-x^3+x^4-\cdots$$

where $x<1$ in the above equations.

### *A.5.1 Properties of Binomial Coefficients*

$$\binom{n}{i}=\binom{n}{n-i}$$

$$\binom{n}{i}=\frac{n}{i}\binom{n-1}{i-1}$$

$$\binom{n}{i}+\binom{n}{i-1}=\binom{n+1}{i}$$

$$\binom{n}{n}=\binom{n}{0}=1$$

$$n!\sim\sqrt{2\pi}n^{n+\frac{1}{2}}e^{-n}\qquad\text{Stirling's formula}$$

## A.6 Other Useful Series and Formulas

$$\sum_{i=0}^{n}i\binom{n}{i}a^{i-1}b^{n-i}=na(a+b)^{n-1}$$

$$\sum_{i=0}^{n}i^2\binom{n}{i}a^ib^{n-i}=n(n-1)a^2(a+b)^{n-2}+na(a+b)^{n-1}$$

$$\sum_{i=1}^{n} \binom{n-1}{i-1} \frac{1}{i} a^i (1-a)^{n-i} = \frac{1}{n} \left[1 - (1-a)^n\right]$$

$$\sum_{i=0}^{\infty} \frac{a^i}{i!} = e^a$$

$$\sum_{i=0}^{\infty} (i+1) \frac{a^i}{i!} = e^a (a+1)$$

$$\lim_{n\to\infty} (1 \pm \frac{x}{n})^n = e^{\pm x}$$

$$\lim_{n\to\infty} x^n \binom{n}{i} = 0 \qquad |x| < 1$$

$$\binom{n}{i} + \binom{n}{i-1} = \binom{n+1}{i}$$

$$\lim_{n\to\infty} \left(1 - \frac{a}{n}\right)^n = e^{-a}$$

$$\lim_{n\to\infty} \binom{n}{i} a^i (1-a)^{n-i} = \frac{a^i e^{-a}}{i!} \qquad a < 1$$

The last equation is used to derive the Poisson distribution from the binomial distribution.

## A.7 Integrals of Exponential Functions

$$\int e^{cx} \, \mathrm{d}x = \frac{1}{c} e^{cx}$$

$$\int x e^{cx^2} \, \mathrm{d}x = \frac{1}{2c} e^{cx^2}$$

$$\int_0^{\infty} e^{-ax} \, \mathrm{d}x = \frac{1}{a}$$

$$\int_0^{\infty} e^{-ax^2} \, \mathrm{d}x = \sqrt{\frac{\pi}{a}}, \qquad a > 0$$

$$\int_0^{\infty} x^n e^{-ax^b} \, \mathrm{d}x = \frac{1}{b} a^{-(n+1)/b} \, \Gamma\left(\frac{n+1}{b}\right)$$

# Appendix B
# Solving Difference Equations

## B.1 Introduction

Difference equations describe discrete-time systems just like differential equations describe continuous-time systems. We encounter difference equations in many fields in telecommunications, digital signal processing, electromagnetics, civil engineering, etc.

In Markov chains, and many queuing models, we often get a special structure for the state transition matrix that produces a recurrence relation between the system states. This appendix is based on the results provided in [1] and [2]. We start first by exploring simple approaches for simple situations, then we deal with the more general situation.

## B.2 First-Order Form

Assume we have the simple recurrence equation

$$s_i = a\, s_{i-1} + b \tag{B.1}$$

where $a$ and $b$ are given. Our task is to find values for the unknown $s_i$ for all values of $i = 0, 1, \cdots$ that satisfy the above equation. This is a first-order form since each sample depends on the immediate past value only.

Since this is a linear relationship, we assume that the solution for $s_i$ is composed of two components, a constant component $c$ and a variable component $v$ that depends on $i$. Thus, we write the trial solution for $s_i$ as

$$s_i = v_i + c \tag{B.2}$$



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6

Substitute this into our recursion to get

$$v_i + c = a\,(v_{i-1} + c) + b \tag{B.3}$$

We can group the constant parts and the variable parts together to get

$$c = a\,c + b \tag{B.4}$$

$$v_i = a\,v_{i-1} \tag{B.5}$$

and the value of the constant component of $s$ is

$$c = \frac{b}{1-a} \tag{B.6}$$

Assume a solution for $v_i$ in (B.5) of the form

$$v_i = \lambda^i \tag{B.7}$$

Substitute this solution in the recursion formula (B.5) for $v_i$ to get

$$\lambda^i = a\,\lambda^{i-1} \tag{B.8}$$

which gives

$$\lambda = a \tag{B.9}$$

and $v_i$ is given by

$$v_i = \lambda^i = a^i \tag{B.10}$$

Thus $s_i$ is given from (B.2) as

$$s_i = a^i + \frac{b}{1-a} \tag{B.11}$$

This is the desired solution to the difference equations.

## B.3 Second-Order Form

Assume we have the simple recurrence equation

$$s_{i+1} + a\,s_i + b\,s_{i-1} = 0 \tag{B.12}$$

This is a second-order form since each sample depends on the two most recent past samples. Assume a solution for $s_i$ of the form

$$s_i = \lambda^i \tag{B.13}$$

The recursion formula gives

$$\lambda^2 + a\,\lambda + b = 0 \tag{B.14}$$

There are two possible solutions (roots) for $\lambda$, which we denote as $\alpha$ and $\beta$, and there are three possible situations.

### B.3.1 Real and Different Roots $\alpha \neq \beta$

When the two roots are real and different, $s_i$ becomes a linear combination of these two solutions

$$s_i = A\alpha^i + B\beta^i \tag{B.15}$$

where $A$ and $B$ are constants. The values of $A$ and $B$ are determined from any restrictions on the solutions for $s_i$ such as given initial conditions. For example, if $s_i$ represent the different components of the distribution vector in a Markov chain, then the sum of all the components must equal unity.

### B.3.2 Real and Equal Roots $\alpha = \beta$

When the two roots are real and equal, $s_i$ is given by

$$s_i = (A + iB)\,\alpha^i \tag{B.16}$$

### B.3.3 Complex Conjugate Roots

In that case we have

$$\alpha = \gamma + j\theta \tag{B.17}$$

$$\beta = \gamma - j\theta \tag{B.18}$$

$s_i$ is given by

$$s_i = \gamma^i\,[A\cos(i\ \theta) + B\sin(i\ \theta)] \tag{B.19}$$

## B.4 General Approach

Consider the $N$-order difference equations given by

$$s_i = \sum_{k=0}^{N} a_k \; s_{i-k}, \qquad i > 0 \tag{B.20}$$

where we assumed $s_i = 0$ when $i < 0$.

We define the one-sided z-transform [3] of $s_i$ as

$$S(z) = \sum_{i=0}^{\infty} s_i \; z^{-i} \tag{B.21}$$

Now take the z-transform of both sides of (B.20) to obtain

$$S(z) - s_0 = \sum_{k=0}^{N} a_k \; z^{-k} \times \left[ \sum_{i=0}^{\infty} s_{i-k} \; z^{-(i-k)} \right] \tag{B.22}$$

We assume that $a_k = 0$ when $k > N$ and we also assume that $s_i = 0$ when $i < 0$. Based on these two assumptions, we can change the upper limit for the summation over $k$ and we can change the variable of summation of the term in square brackets as follows.

$$S(z) - s_0 = \sum_{k=0}^{\infty} a_k \; z^{-k} \times \left[ \sum_{m=0}^{\infty} s_m \; z^{-m} \right] \tag{B.23}$$

where we introduced the new variable $m = i - k$. Define the z-transform of the coefficients $a_k$ as

$$A(z) = \sum_{k=0}^{\infty} a_k \; z^{-k} \tag{B.24}$$

Thus we get

$$\begin{aligned} S(z) - s_0 &= A(z) \times \left[ \sum_{m=0}^{\infty} s_m \; z^{-m} \right] \\ &= A(z) \times S(z) \end{aligned} \tag{B.25}$$

Notice that the two summations on the RHS are now independent. Thus we finally get

$$S(z) = \frac{s_0}{1 - A(z)} \tag{B.26}$$

We can write the above equation in the form

$$S(z) = \frac{s_0}{\mathcal{D}(z)} \tag{B.27}$$

where the denominator polynomial is

$$\mathcal{D}(z) = 1 - A(z) \tag{B.28}$$

MATLAB allows us to find the inverse z-transform of $S(z)$ using the command

```
RESIDUE(a,b)
```

where $a$ and $b$ are the coefficients of the nominator and denominator polynomials $A(z)$ and $B(z)$, respectively, in descending powers of $z^{-1}$.

The function `RESIDUE` returns the column vectors $r$, $p$, and $c$ which give the residues, poles, and direct terms, respectively.

The solution for $s_i$ is given by the expression

$$s_i = c_i + \sum_{j=1}^{m} r_j (p_j)^{(i-1)} \qquad i > 0 \tag{B.29}$$

where $m$ is the number of elements in $r$ or $p$ vectors.

## References

1. J.R. Norris, *Markov Chains* (Cambridge University Press, New York, 1997)
2. V.K. Ingle, J.G. Proakis, *Digital Signal Processing Using MATLAB* (Brooks/Cole, Pacific Grove, 2000)
3. A. Antoniou, *Digital Filters: Analysis, Design, and Applications* (McGraw-Hill, New York, 1993)

# Appendix C
# Finding s($n$) Using the Z-Transform

When the transition matrix $\mathbf{P}$ of a Markov chain is not diagonalizable, we could use the z-transform technique to find the value of the distribution vector at any time instance $\mathbf{s}(n)$. An alternative, and more appealing technique, is to use the Jordan Canonic form (JCF). However, we will explain the z-transform here. The z-transform of the distribution vector $\mathbf{s}$ is given by

$$\mathbf{S}(z) = \sum_{n=0}^{\infty} \mathbf{s}(n)\, z^{-n} \tag{C.1}$$

We express $\mathbf{s}(n)$ in the above equation in terms of the transition matrix $\mathbf{P}$ using the relation

$$\mathbf{s}(n) = \mathbf{P}^n\, \mathbf{s}(0) \tag{C.2}$$

Alternatively, $\mathbf{s}(n)$ can be written as

$$\mathbf{s}(n) = \mathbf{P}\, \mathbf{s}(n-1) \tag{C.3}$$

From (C.1), the z-transform of $\mathbf{s}(n)$ can be written as

$$\mathbf{S}(z) = \mathbf{s}(0) + \sum_{n=1}^{\infty} \mathbf{P}\, \mathbf{s}(n-1) z^{-n} \tag{C.4}$$

Thus we have

$$\mathbf{S}(z) - \mathbf{s}(0) = z^{-1}\mathbf{P} \sum_{n=1}^{\infty} \mathbf{s}(n-1)\, z^{-(n-1)} \tag{C.5}$$

Changing the index of summation, we get



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6

$$\mathbf{S}(z) - \mathbf{s}(0) = z^{-1}\,\mathbf{PS}(z) \tag{C.6}$$

we can thus obtain an expression for the z-transform of the distribution vector as

$$\mathbf{S}(z) = \left(\mathbf{I} - z^{-1}\mathbf{P}\right)^{-1}\mathbf{s}(0) \tag{C.7}$$

Denoting the transform pair

$$\mathbf{S}(z) \Leftrightarrow \mathbf{s}(n) \tag{C.8}$$

then we can write (C.7), using (C.2), in the form

$$\left(\mathbf{I} - z^{-1}\mathbf{P}\right)^{-1}\mathbf{s}\,(0) \Leftrightarrow \mathbf{P}^n\mathbf{s}\,(0) \tag{C.9}$$

Since $\mathbf{s}\,(0)$ is arbitrary, we have the z-transform pair

$$\mathbf{P}^n \Leftrightarrow \left(\mathbf{I}{-}z^{-1}\mathbf{P}\right)^{-1} \tag{C.10}$$

*WWW:* We have defined the MATLAB function `invz(B,a)` which accepts the matrix $B$ whose elements are the nominator polynomials of $\left(\mathbf{I} - z^{-1}\mathbf{P}\right)^{-1}$ and the polynomial $a$ which is the denominator polynomial of $\left(\mathbf{I} - z^{-1}\mathbf{P}\right)^{-1}$. The function returns the residue matrices $r$ that correspond to each pole of the denominator.

The following two examples illustrate the use of this function.

*Example C.1.* Consider the Markov chain matrix

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.8 & 0.4 \\ 0.5 & 0 & 0.3 \\ 0 & 0.2 & 0.3 \end{bmatrix}$$

Use the z-transform technique to find a general expression for the distribution vector at step $n$ and find its value when $n = 3$ and 100 assuming an initial distribution vector $\mathbf{s}(0) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^t$

First we must form

$$\mathbf{I} - z^{-1}\mathbf{P} = \begin{bmatrix} 1 - 0.5z^{-1} & -0.8z^{-1} & -0.4z^{-1} \\ -0.5z^{-1} & 1 & -0.3z^{-1} \\ 0 & -0.2z^{-1} & 1 - 0.3z^{-1} \end{bmatrix}$$

we have to determine the inverse of this matrix using determinants or any other technique to obtain

$$\left(\mathbf{I}\ -z^{-1}\mathbf{P}\right)^{-1} = \frac{1}{D}\times$$
$$\begin{bmatrix} 1-0.3z^{-1}-0.06z^{-2} & 0.8z^{-1}-0.16z^{-2} & 0.4z^{-1}+0.24z^{-2} \\ 0.5z^{-1}-0.15z^{-2} & 1-0.8z^{-1}+0.15z^{-2} & 0.3z^{-1}+0.05z^{-2} \\ 0.10z^{-2} & 0.2z^{-1}-0.10z^{-2} & 1-0.5z^{-1}-0.40z^{-2} \end{bmatrix}$$

where

$$D = \left(1-z^{-1}\right)\left(1+0.45z^{-1}\right)\left(1-0.25z^{-1}\right)$$

We notice that the poles of this matrix are the eigenvalues of $\mathbf{P}$. We now have to find the inverse z-transform of $\left(\mathbf{I}-z^{-1}\mathbf{P}\right)^{-1}$ which can be done on a term-by-term basis [1] or by using the MATLAB function `invz(B,a)` [1].

$$\mathbf{P}^n = \begin{bmatrix} 0.59 & 0.59 & 0.59 \\ 0.32 & 0.32 & 0.32 \\ 0.09 & 0.09 & 0.09 \end{bmatrix} +$$
$$(-0.45)^n \begin{bmatrix} 0.27 & -0.51 & 0.06 \\ -0.37 & 0.70 & -0.08 \\ 0.10 & -0.19 & 0.02 \end{bmatrix} +$$
$$(0.25)^n \begin{bmatrix} 0.14 & -0.08 & -0.64 \\ 0.05 & -0.02 & -0.24 \\ -0.19 & 0.10 & 0.88 \end{bmatrix} \qquad n = 0, 1, \ldots$$

We notice that the matrix corresponding to the pole $z^{-1} = 1$ is column stochastic and all its columns are equal. For the two matrices corresponding to the poles $z^{-1} = -0.45$ and 0.25, the sum of each column is exactly zero.

$\mathbf{s}(3)$ is given from the above equation by substituting $n = 3$ in the expression for $\mathbf{P}^n$.

$$\mathbf{s}(3) = \mathbf{P}^3\mathbf{s}(0) = \mathbf{s}(3) = \begin{bmatrix} 0.57 \\ 0.36 \\ 0.08 \end{bmatrix}^t$$

$\mathbf{s}(100)$ is given by

$$\mathbf{s}(100) = \mathbf{P}^{100}\mathbf{s}(0) = \begin{bmatrix} 0.59 \\ 0.32 \\ 0.09 \end{bmatrix}^t$$

■

*Example C.2.* Consider the Markov chain matrix

$$\mathbf{P} = \begin{bmatrix} 0.2 & 0.4 & 0.4 \\ 0.8 & 0.1 & 0.4 \\ 0 & 0.5 & 0.2 \end{bmatrix}$$

Use the z-transform technique to find a general expression for the distribution vector at step $n$ and find its value when $n = 7$ for an initial distribution vector $\mathbf{s}(0) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^t$

First we must form

$$\mathbf{I} - z^{-1}\mathbf{P} = \begin{bmatrix} 1 - 0.2z^{-1} & -0.4z^{-1} & -0.4z^{-1} \\ -0.8z^{-1} & 1 - 0.1z^{-1} & -0.4z^{-1} \\ 0 & -0.5z^{-1} & 1 - 0.2z^{-1} \end{bmatrix}$$

we have to determine the inverse of this matrix using determinants or any other technique to obtain

$$\left(\mathbf{I} - z^{-1}\mathbf{P}\right)^{-1} = \frac{1}{D} \times \begin{bmatrix} 1 - 0.3z^{-1} - 0.18z^{-2} & 0.4z^{-1} + 0.12z^{-2} & 0.4z^{-1} + 0.12z^{-2} \\ 0.8z^{-1} - 0.16z^{-2} & 1 - 0.4z^{-1} + 0.40z^{-2} & 0.4z^{-1} + 0.24z^{-2} \\ 0.40z^{-2} & 0.5z^{-1} - 0.10z^{-2} & 1 - 0.3z^{-1} - 0.30z^{-2} \end{bmatrix}$$

where

$$D = \left(1 - z^{-1}\right)\left(1 + 0.3z^{-1}\right)\left(1 + 0.2z^{-1}\right)$$

We notice that the poles of this matrix are the eigenvalues of $\mathbf{P}$. We now have to find the inverse z-transform of $\left(\mathbf{I} - z^{-1}\mathbf{P}\right)^{-1}$ which can be done on a term-by-term basis [1] or by using the function `invz(B,a)`.

$$\mathbf{P}^n = \begin{bmatrix} 0.33 & 0.33 & 0.33 \\ 0.41 & 0.41 & 0.41 \\ 0.26 & 0.26 & 0.26 \end{bmatrix} + (-0.3)^n \begin{bmatrix} 0.0 & 0.0 & 0.0 \\ -3.08 & 1.92 & 0.92 \\ 3.08 & -1.92 & -0.92 \end{bmatrix} + (-0.2)^n \begin{bmatrix} 0.67 & 0.67 & 0.67 \\ 2.67 & 2.67 & 2.67 \\ -3.33 & -3.33 & -3.33 \end{bmatrix} \qquad n = 0, 1, \ldots$$

We notice that the matrix corresponding to the pole $z^{-1} = 1$ is column stochastic and all its columns are equal. For the two matrices corresponding to the poles $z^{-1} = -0.3$ and 0.2, the sum of each column is exactly zero.

$\mathbf{s}(7)$ is given from the above equation as

$$\mathbf{s}(7) = \mathbf{P}^7\mathbf{s}(0)$$

or

$$\mathbf{s}(7) = \begin{bmatrix} 0.33 \\ 0.41 \\ 0.26 \end{bmatrix}$$

■

## C.1 Problems

**C.1.** Use the z-transform to find the distribution vector at any time instant $n$ for the transition matrix

$$\begin{bmatrix} 0.2 & 0.4 & 0.4 \\ 0.8 & 0.1 & 0.4 \\ 0 & 0.5 & 0.2 \end{bmatrix}$$

**C.2.** Use the z-transform to find the distribution vector at any time instant $n$ for the transition matrix

$$\begin{bmatrix} 0.5 & 0.3 \\ 0.5 & 0.7 \end{bmatrix}$$

**C.3.** Use the z-transform to find the distribution vector at any time instant $n$ for the transition matrix

$$\begin{bmatrix} 0.2 & 0.3 & 0.5 & 0.6 \\ 0.3 & 0.1 & 0.2 & 0.1 \\ 0.5 & 0.1 & 0.1 & 0.2 \\ 0 & 0.5 & 0.2 & 0.1 \end{bmatrix}$$

**C.4.** Use the z-transform to find the distribution vector at any time instant $n$ for the transition matrix

$$\begin{bmatrix} 0.1 & 0.4 & 0.1 \\ 0.7 & 0 & 0.3 \\ 0.2 & 0.6 & 0.6 \end{bmatrix}$$

## Reference

1. V.K. Ingle, J.G. Proakis, *Digital Signal Processing Using MATLAB* (Brooks/Cole, Pacific Grove, CA, 2000)

# Appendix D
# Vectors and Matrices

## D.1 Introduction

The objective of this appendix is to briefly review the main topics related to matrices since we encounter them in most of our work on queuing theory. There are excellent books dealing with this subject and we refer the reader to them for a more comprehensive treatment. Perhaps one of the best books on matrices is [1]. This is book is not only easy to read, but the author's writing style and insights make the topic actually enjoyable. The reader that wants to read a comprehensive book, albeit somewhat dry, could consult [2].

## D.2 Scalars

A scalar is a real or complex number. We denote scalars in this book by lowercase letters or Greek letters. Sometimes, but not too often, we use uppercase letters to denote scalars.

## D.3 Vectors

A vector is an ordered collection of scalars $v_1, v_2, \cdots$, which are its *components*. We use bold lowercase letters to indicate vectors. A subscript on a vector will denote a particular component of the vector. Thus the scalar $v_2$ denotes the second component of the vector $\mathbf{v}$. Usually we say $\mathbf{v}$ is a 3-vector to signify that $\mathbf{v}$ has three components.

A vector $\mathbf{v}$ could have its component values change with the time index $n$. At time $n$, that vector will be denoted by $\mathbf{v}(n)$.



F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6

As an example, a vector that has only three components is written as:

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \tag{D.1}$$

To conserve space, we usually write a column vector in the following form

$$\mathbf{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^t \tag{D.2}$$

where the superscript $t$ indicates that the vector is to be *transposed* by arranging the components horizontally instead of vertically.

A vector is, by default, a *column* vector. A *row* vector $\mathbf{r}$ could be obtained by transposing $\mathbf{v}$

$$\mathbf{r} = \mathbf{v}^t = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \tag{D.3}$$

## D.4 Arithmetic Operations with Vectors

Two vectors can be added if they have the same number of components

$$\mathbf{v} + \mathbf{w} = \begin{bmatrix} v_1 + w_1 \\ v_2 + w_2 \\ v_3 + w_3 \end{bmatrix} = \begin{bmatrix} (v_1 + w_1) & (v_2 + w_2) & (v_3 + w_3) \end{bmatrix}^t \tag{D.4}$$

A vector can be multiplied by a scalar if all the vector components are multiplied by the same scalar as shown

$$a\ \mathbf{v} = \begin{bmatrix} a v_1 & a v_2 & a v_3 \end{bmatrix}^t \tag{D.5}$$

A row vector can multiply a column vector as long as the two vectors have the same number of components.

$$\mathbf{r}\ \mathbf{c} = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} \tag{D.6}$$

$$= r_1\ c_1 + r_2\ c_2 + r_3\ c_3 \tag{D.7}$$

The *dot product* of a vector is just a scalar that is defined as

$$\mathbf{v} \cdot \mathbf{w} = v_1\ w_1 + v_2\ w_2 + v_3\ w_3 \tag{D.8}$$

where corresponding components are multiplied together.

Two vectors $\mathbf{x}$ and $\mathbf{y}$ are said to be *orthogonal* if their dot product vanishes.

$$\mathbf{x} \cdot \mathbf{y} = 0$$

## D.5 Linear Independence of Vectors

The two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ are said to be *linearly independent* or simply *independent* when

$$a_1\mathbf{x}_1 + a_2\mathbf{x}_2 = \mathbf{0} \tag{D.9}$$

is true if and only if

$$a_1 = 0$$
$$a_2 = 0$$

## D.6 Matrices

Assume we are given this set of linear equations

$$\begin{aligned} a_{11}\, x_1 + a_{12}\, x_2 + a_{13}\, x_3 &= b_1 \\ a_{21}\, x_1 + a_{22}\, x_2 + a_{23}\, x_3 &= b_2 \\ a_{31}\, x_1 + a_{32}\, x_2 + a_{33}\, x_3 &= b_3 \end{aligned} \tag{D.10}$$

we can write the equations in the form

$$\begin{bmatrix} a_{11}\ a_{12}\ a_{13} \\ a_{21}\ a_{22}\ a_{23} \\ a_{31}\ a_{32}\ a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{D.11}$$

where the first equation is obtained by multiplying the first row and the vector $\mathbf{x}$ and the second equation is obtained by multiplying the second row and the vector $\mathbf{x}$, and so on. A concise form for writing the system of linear equations in (D.10) is to use vectors and matrices

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{D.12}$$

The *coefficient matrix* of this system of equations is given by the array of numbers

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{D.13}$$

A matrix with $m$ rows and $n$ columns is called a matrix of order $m \times n$ or simply an $m \times n$ matrix. When $m = n$ we have a square matrix of order $n$. The elements $a_{ii}$ constitute the *main diagonal* of $\mathbf{A}$.

## D.7 Matrix Addition

If $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{B}$ is an $m \times n$ matrix, then we can add them to produce an $m \times n$ matrix $\mathbf{C}$

$$\mathbf{C} = \mathbf{A} + \mathbf{B} \tag{D.14}$$

where the elements of $\mathbf{C}$ are given by

$$c_{ij} = a_{ij} + b_{ij} \tag{D.15}$$

with $1 \leq i \leq m$ and $1 \leq j \leq n$.

## D.8 Matrix Multiplication

If $\mathbf{A}$ is an $l \times m$ matrix and $\mathbf{B}$ is an $m \times n$ matrix, then we can multiply them to produce an $l \times n$ matrix $\mathbf{C}$

$$\mathbf{C} = \mathbf{A}\,\mathbf{B} \tag{D.16}$$

where the elements of $\mathbf{C}$ are given by

$$c_{ij} = \sum_{k=1}^{m} a_{ik}\, b_{kj} \tag{D.17}$$

with $1 \leq i \leq l$ and $1 \leq j \leq n$.

## D.9 Inverse of a Matrix

Given the matrix $\mathbf{A}$, its *left inverse* is defined by the equation

$$\mathbf{LA} = \mathbf{I} \tag{D.18}$$

The *right inverse* is defined by the equation

$$\mathbf{AR} = \mathbf{I} \tag{D.19}$$

When $\mathbf{A}$ is square, the left and the right inverses are equal and we denote the *inverse* as $\mathbf{A}^{-1}$ which satisfies the equations

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I} \tag{D.20}$$

The inverse of the matrix is found by treating the above equation as a system of simultaneous equations in the unknown matrix $\mathbf{A}^{-1}$. The matrix $\mathbf{A}^{-1}$ is first written as

$$\mathbf{A}^{-1} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_n \end{bmatrix} \tag{D.21}$$

where $n$ is the dimension of $\mathbf{A}$. The $i^{\text{th}}$ column $\mathbf{a}_i$ of $\mathbf{A}^{-1}$ is treated as an unknown vector that is to be found from the equation

$$\mathbf{A}\,\mathbf{a}_i = \begin{bmatrix} 0 \cdots 0\ 1\ 0 \cdots 0 \end{bmatrix}^t \tag{D.22}$$

where the vector on the RHS has 1 in the $i^{\text{th}}$ location. Gauss elimination is a useful technique for finding the inverse of the matrix.

Not all matrices have inverses. A square matrix has an inverse only if its rank equals the number of rows (rank is explained Sect. D.11).

## D.10 Nullspace of a Matrix

Given an $m \times n$ matrix $\mathbf{A}$, a nonzero $n$-vector $\mathbf{x}$ is said to be a nullvector for $\mathbf{A}$ when

$$\mathbf{Ax} = \mathbf{0} \tag{D.23}$$

All possible solutions of the above equation form the *nullspace* of $\mathbf{A}$, which we denote by *null*($\mathbf{A}$). If $n$ is the number of all possible and independent solutions, then we have

$$n = \text{null}\,(\mathbf{A}) \tag{D.24}$$

Finding the nullvectors $\mathbf{x}$ is done by solving (D.23) as a system of homogeneous linear equations.

MATLAB offers the function `null` to find all the nullvectors of a given matrix. The function `null(A)` produces the null space basis vectors. The function `null(A,'r')` produces the nullvectors in rational format for presentation purposes. For example

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix}$$

$$\text{null}(A,'r') = \begin{bmatrix} -2 & -3 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

## D.11 The Rank of a Matrix

The maximum number of linearly independent rows, or columns of a matrix **A** is the rank of the matrix. This number $r$ is denoted by

$$r = \text{rank}\,(\mathbf{A}) \tag{D.25}$$

A matrix has only one rank regardless of the number of rows or columns. An $m \times n$ matrix (where $m \leq n$) is said to be *full rank* when $\text{rank}(\mathbf{A}) = m$.

The rank $r$ of the matrix equals the number of pivots of the matrix when it is transformed to its echelon form (see Sect. D.19.3 for definition of echelon form).

## D.12 Eigenvalues and Eigenvectors

Eigenvalues and eigenvectors apply only to *square* matrices. Consider the special situation when we have

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \tag{D.26}$$

The number $\lambda$ is called the *eigenvalue* and $\mathbf{x}$ is called the *eigenvector*. Math packages help us find all possible eigenvalues and the corresponding eigenvectors of a given matrix.

We can combine all the eigenvectors into the eigenvector matrix $\mathbf{X}$ whose columns are the eigenvectors and we could write

$$\mathbf{A}\,\mathbf{X} = \mathbf{X}\,\mathbf{D} \tag{D.27}$$

where

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{bmatrix} \tag{D.28}$$

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \tag{D.29}$$

When the inverse $\mathbf{X}^{-1}$ exists, we can diagonalize $\mathbf{A}$ in the form

$$\mathbf{A} = \mathbf{X}\,\mathbf{D}\,\mathbf{X}^{-1} \tag{D.30}$$

## D.13 Diagonalizing a Matrix

We say that the square matrix $\mathbf{A}$ is diagonalized when it can be written in the form (D.30). A matrix that has no repeated eigenvalues can always be diagonalized. If some eigenvalues are repeated, then the matrix might or might not be diagonalizable.

A general rule for matrix diagonalization is as follows: A matrix is diagonalizable only when its JCF is diagonal. Section 3.14.1 on page 105 discusses the JCF of a matrix.

## D.14 Triangularizing a Matrix

Sometimes it is required to change a given matrix $\mathbf{A}$ to an upper triangular matrix. The resulting matrix is useful in many applications such as

1. Solving a system of linear equations.
2. Finding the eigenvector of a matrix given an eigenvalue.
3. Finding the inverse of a matrix.

Householder or Givens techniques can be used to triangularize the matrix. We illustrate Givens technique here only. The idea is to apply a series of plane rotation, or Givens rotation, matrices on the matrix in question in order to create zeros below the main diagonal. We start with the first column and create zeros below the element $a_{11}$. Next, we start with the second column and create zeros below the element $a_{22}$ and so on. Each created zero does not disturb the previously created zeros. Furthermore, the plane rotation matrix is very stable and does not require careful choice of the pivot element. Section D.19.7 discusses the Givens plane rotation matrices.

Assume, as an example, a $5 \times 5$ matrix $\mathbf{A}$ and we are interested in eliminating element $a_{42}$, we choose the Givens rotation matrix $\mathbf{G}_{42}$

$$\mathbf{G}_{42} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c & 0 & s & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -s & 0 & c & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{D.31}$$

where $c = \cos\theta$ and $s = \sin\theta$. Premultiplying a matrix $\mathbf{A}$ by $\mathbf{G}_{42}$ modifies only rows 2 and 4. All other rows are left unchanged. The elements in rows 2 and 4 become

$$a_{2j} = \quad c\, a_{2j} + s\, a_{4j} \tag{D.32}$$

$$a_{4j} = -s\, a_{2j} + c\, a_{4j} \tag{D.33}$$

The new element $a_{42}$ is eliminated from the above equation if we have

$$\tan\theta = \frac{a_{42}}{a_{22}} \tag{D.34}$$

The following MATLAB code illustrates how an input matrix is converted to an upper triangular matrix.

```
%File: givens.m
%The program accepts a matrix performs a series of
%Givens rotations to transform the matrix into an
%upper-triangular matrix.
%The new matrix is printed after each zero is created.
%
%Input matrix to be triangularized
A=[-0.6  0.2  0.0
    0.1 -0.5  0.6
    0.5  0.3 -0.6]
n=3;
%iterate for first n-1 columns
for j=1:n-1
  %cancel all subdiagonal elements
  %in column j
  for i=j+1:n
    %calculate theta
    theta=atan2(q(i,j),q(j,j))
    for k=j:n
      temp_x= A(j,k)*cos(theta)+A(i,k)*sin(theta);
      temp_y=-A(j,k)*sin(theta)+A(i,k)*cos(theta);
```

```
        %update new elements in rows i and j
        A(j,k) = temp_x;
        A(i,k) = temp_y;
      end
      A %print q after each iteration.
    end
  end
```

An example of using the Givens rotation technique is explained in the next section.

## D.15 Linear Equations

A system of linear equations has the form

$$\mathbf{Ax} = \mathbf{b} \tag{D.35}$$

where the *coefficient matrix* $\mathbf{A}$ is a given $n \times n$ nonsingular matrix so that the system possesses a unique solution. The vector $\mathbf{b}$ is also given. The unknown vector $\mathbf{x}$ is to be found. The system is said to be *homogeneous* when $\mathbf{b}$ is zero, otherwise, the system is said to be *nonhomogeneous*. Before we discuss methods for obtaining the solution to the above equation, we define first some *elementary row operations* [2, 4]:

1. Interchange of two rows.
2. Multiplication of a row by a nonzero constant.
3. Addition of a constant multiple of one row to another row.

Each of the above elementary row operations is implemented by multiplying the matrix from the left by an appropriate matrix $\mathbf{E}$ called an *elementary matrix*. The three elementary matrices corresponding to the above three elementary row operations are illustrated below for $4 \times 4$ matrices [6].

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ a & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{D.36}$$

There are two classes of numerical methods for finding a solution. The *direct* methods guarantee to find a solution in one step. This is the recommended approach for general situations and for small values of $n$. The *iterative* methods start with an assumed solution then try to refine this assumption until the succeeding estimates of the solution converge to within a certain error limit. This approach is useful for large values of $n$ and for sparse matrices where $\mathbf{A}$ has a large number of zeros.

The advantage of iterative solutions is that they practically eliminate arithmetic roundoff errors and produce results with accuracy close to the machine precision [3]. We discuss the two approaches in the following sections. We refer the reader to Appendix D for a discussion of the techniques used by MATLAB to numerically find the solution.

In the following section we review the useful techniques for solving systems of linear equations.

### *D.15.1 Gauss Elimination*

Gauss elimination solves a system of linear equations by transforming **A** into upper triangular form using elementary row operations. The solution is then found using back-substitution. To create a zero at position (2,1) we need to multiply row 2 by $a_{21}/a_{11}$ then subtract this row from row 1. This is equivalent to a row operation matrix of the form

$$\mathbf{E}_{21} = \begin{bmatrix} 1 & 0 & 0 \\ e & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ with } e = -\frac{a_{21}}{a_{11}} \tag{D.37}$$

The reader could verify that this matrix performs the desired operation and creates a zero at position (2,1).

We illustrate this using an example of a $3 \times 3$ system for a matrix with rank 2:

$$\begin{bmatrix} -0.5 & 0.3 & 0.2 \\ 0.3 & -0.4 & 0.5 \\ 0.2 & 0.1 & -0.7 \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{D.38}$$

Step 1 Create a zero in the (2,1) position using the elementary matrix $\mathbf{E}_{21}$

$$\mathbf{E}_{21} = \begin{bmatrix} 1 & 0 & 0 \\ r & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ with } e = -\frac{a_{21}}{a_{11}}$$

which gives

$$\mathbf{T}_1 = \mathbf{E}_{21}\mathbf{A} = \begin{bmatrix} -0.5 & 0.3 & 0.2 \\ 0 & -0.22 & 0.62 \\ 0.2 & 0.1 & -0.7 \end{bmatrix}$$

Step 2 Create a zero in the (31) position using the elementary matrix $\mathbf{E}_{31}$

$$\mathbf{E}_{31} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ r & 0 & 1 \end{bmatrix} \text{ with } r = -\frac{a_{31}}{a_{11}}$$

which gives

$$\mathbf{T}_2 = \mathbf{E}_{31}\mathbf{T}_1 = \begin{bmatrix} -0.5 & 0.3 & 0.2 \\ 0 & -0.22 & 0.62 \\ 0 & 0.22 & -0.62 \end{bmatrix}$$

Step 3 Create a zero in the (3, 2) position using the elementary matrix $\mathbf{E}_{32}$

$$\mathbf{E}_{32} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & e & 1 \end{bmatrix} \text{ with } e = -\frac{a_{32}}{a_{22}}$$

which gives

$$\mathbf{T}_3 = \mathbf{E}_{32}\mathbf{T}_2 = \begin{bmatrix} -0.5 & 0.3 & 0.2 \\ 0 & -0.22 & 0.62 \\ 0 & 0 & 0 \end{bmatrix}$$

Note that the triangularization operation produced a row of zeros, indicating that the rank of this matrix is indeed 2.

Step 5 Solve for the components of $\mathbf{s}$ assuming $s_3 = 1$

$$\mathbf{s} = \begin{bmatrix} 2.0909 & 2.8182 & 1 \end{bmatrix}$$

After normalization, the true state vector becomes.

$$\mathbf{s} = \begin{bmatrix} 0.3538 & 0.4769 & 0.1692 \end{bmatrix}$$

### *D.15.2 Gauss–Jordan Elimination*

Gauss–Jordan elimination is a powerful method for solving a system of linear equations of the form

$$\mathbf{Ax} = \mathbf{b} \tag{D.39}$$

where $\mathbf{A}$ is a square matrix of dimension $n \times n$ and $\mathbf{x}$ and $\mathbf{b}$ are column vectors with $n$ components each. The above equation can be written in the form

$$\mathbf{Ax} = \mathbf{Ib} \tag{D.40}$$

where $\mathbf{I}$ is the $n \times n$ unit matrix.

We can find the unknown vector $\mathbf{x}$ by multiplying by the inverse of matrix $\mathbf{A}$ to get the solution

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \tag{D.41}$$

It is not recommended to find the inverse as mentioned above. Gauss elimination and Gauss–Jordan elimination techniques are designed to find the solution without the need to find the inverse of the matrix. This solution in the above equation can be written in the form

$$\mathbf{Ix} = \mathbf{A}^{-1}\mathbf{b} \tag{D.42}$$

Comparing (D.40)–(D.42) we conclude that we find our solution if somehow we were able to convert the matrix $\mathbf{A}$ to $\mathbf{I}$ using repeated row operation. Gauss–Jordan elimination does exactly that by constructing the augmented matrix $\left[\mathbf{A}\ \mathbf{I}\right]$ and converting it to the matrix $\left[\mathbf{I}\ \mathbf{A}^{-1}\right]$.

*Example D.1.* Solve the following set of linear equations.

$$\begin{aligned} 2x_1 \quad -x_2 \quad\quad\ \ &= 0 \\ -x_1 +2x_2 \quad -x_3 &= 3 \\ -x_2 +2x_3 &= 2 \end{aligned}$$

We have

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} 0 & 3 & 2 \end{bmatrix}^t$$

The augmented matrix is

$$\left[\mathbf{A}\ \mathbf{I}\right] = \begin{bmatrix} 2 & -1 & 0 & 1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 1 & 0 \\ 0 & -1 & 2 & 0 & 0 & 1 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 2 & -1 & 0 & 1 & 0 & 0 \\ 0 & 3 & -2 & 1 & 2 & 0 \\ 0 & -1 & 2 & 0 & 0 & 1 \end{bmatrix} \qquad \text{row } 1 + 2 \text{ row } 2$$

$$\rightarrow \begin{bmatrix} 2 & -1 & 0 & 1 & 0 & 0 \\ 0 & 3 & -2 & 1 & 2 & 0 \\ 0 & 0 & 4 & 1 & 2 & 3 \end{bmatrix} \qquad \text{row } 2 + 3 \text{ row } 3$$

So far we have changed our system matrix $\mathbf{A}$ into an upper triangular matrix. Gauss elimination would be exactly that and our solution could be found by forward substitution.

Gauss–Jordan elimination continues by eliminating all elements not on the main diagonal using row operations.

$$[A\ I] \rightarrow \begin{bmatrix} 6 & 0 & -2 & 4 & 2 & 0 \\ 0 & 3 & -2 & 1 & 2 & 0 \\ 0 & 0 & 4 & 1 & 2 & 3 \end{bmatrix} \qquad 3 \text{ row } 1 + \text{row } 2$$

$$\rightarrow \begin{bmatrix} 12 & 0 & 0 & 9 & 6 & 3 \\ 0 & 3 & -2 & 1 & 2 & 0 \\ 0 & 0 & 4 & 1 & 2 & 3 \end{bmatrix} \qquad 2 \text{ row } 1 + \text{row } 3$$

$$\rightarrow \begin{bmatrix} 12 & 0 & 0 & 9 & 6 & 3 \\ 0 & 6 & 0 & 3 & 6 & 3 \\ 0 & 0 & 4 & 1 & 2 & 3 \end{bmatrix} \qquad 2 \text{ row } 2 + \text{row } 3$$

Now we can simplify to get the unit matrix as follows

$$\left[\mathbf{I}\ \mathbf{A}^{-1}\right] = \begin{bmatrix} 1 & 0 & 0 & \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 1 & 0 & \frac{1}{2} & 1 & \frac{1}{2} \\ 0 & 0 & 1 & \frac{1}{4} & \frac{1}{2} & \frac{3}{4} \end{bmatrix}$$

As a result, we now know the matrix $\mathbf{A}^{-1}$ and the solution to our system of equations is given by

$$\begin{aligned} \mathbf{x} &= \mathbf{A}^{-1}\mathbf{b} \\ &= \begin{bmatrix} 2 & 4 & 3 \end{bmatrix}^t \end{aligned}$$

### D.15.3 Row Echelon Form and Reduced Row Echelon Form

The row echelon forms tells us the important information about a system of linear equations such as whether the system has a unique solution, no solution or an infinity of solutions.

We start this section with a definition of a *pivot*. A pivot is the fist nonzero element in each row of a matrix. A matrix is said to be in *row echelon form* if it has the following properties [7]

1. Rows of all zeros appear at the bottom of the matrix.
2. Each pivot has the value 1.
3. Each pivot occurs in a column that is strictly to the right of the pivot above it.

A matrix is said to be in *reduced row echelon form* if it satisfies one additional property.

4. Each pivot is the only nonzero entry in its column.

MATLAB offers the function `rref` to find the reduced row echelon form of a matrix.

Consider the system of equations

$$\begin{aligned} x_1 + 2x_2 + 3x_3 &= 1 \\ 2x_1 + 3x_2 + 4x_3 &= 4 \\ 3x_1 + 3x_2 + 5x_3 &= 3 \end{aligned}$$

The augmented matrix is given by

$$\left[\mathbf{A}\ \mathbf{b}\right] = \begin{bmatrix} 1 & 2 & 3 & 1 \\ 2 & 3 & 4 & 4 \\ 3 & 3 & 5 & 3 \end{bmatrix}$$

The reduced echelon form is given by

$$\begin{aligned} \mathbf{R} &= \text{rref}(\mathbf{A}\ ,\ \mathbf{b}) \\ &= \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 1 & -3 \end{bmatrix} \end{aligned}$$

Thus the solution to our system of equations is

$$\mathbf{x} = \begin{bmatrix} 1 & 4 & -3 \end{bmatrix}^t$$

Let us now see the reduced echelon form when the system has no solution.

$$\begin{aligned} 3x_1 + 2x_2 + x_3 &= 1 \\ 2x_1 + x_2 + x_3 &= 0 \\ 6x_1 + 2x_2 + 4x_3 &= 6 \end{aligned}$$

The augmented matrix is given by

$$\left[\mathbf{A}\ \mathbf{b}\right] = \begin{bmatrix} 3 & 2 & 1 & 3 \\ 2 & 1 & 1 & 0 \\ 6 & 2 & 4 & 6 \end{bmatrix}$$

The reduced echelon form is given by

$$\begin{aligned} \mathbf{R} &= \text{rref}(\mathbf{A}\ ,\ \mathbf{b}) \\ &= \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

The last equation implies that

$$0x_1 + 0x_2 + 0x_3 = 1$$

There are no values of $x_1$, $x_2$, and $x_3$ that satisfy this equation and hence the system has no solution.

The following system has an infinite number of solutions.

$$\begin{aligned} 3x_1 + 2x_2 + x_3 &= 3 \\ 2x_1 + x_2 + x_3 &= 0 \\ 5x_1 + 3x_2 + 2x_3 &= 3 \end{aligned}$$

The augmented matrix is given by

$$\left[\mathbf{A}\ \mathbf{b}\right] = \begin{bmatrix} 3 & 2 & 1 & 3 \\ 2 & 1 & 1 & 0 \\ 5 & 3 & 2 & 3 \end{bmatrix}$$

The reduced echelon form is given by

$$\begin{aligned} \mathbf{R} &= \text{rref}(\mathbf{A}\ ,\ \mathbf{b}) \\ &= \begin{bmatrix} 1 & 0 & 1 & -3 \\ 0 & 1 & -1 & 6 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

Thus we have two equations for three unknowns and we could have infinity solutions.

## D.16 Direct Techniques for Solving Systems of Linear Equations

Direct techniques for solving systems of linear equations are usually the first one attempted to obtain the solution. These techniques are appropriate for small matrices where computational errors will be small. In the next section we discuss iterative techniques which are useful for large matrices.

### *D.16.1 Givens Rotations*

Givens rotations operation performs a number of orthogonal similarity transformations on a matrix to make it an upper triangular matrix. Another equivalent technique is the Householder transformation but we will illustrate the Givens technique here. The technique is very stable and does not suffer from the pivot problems of Gauss elimination. We start with the equilibrium steady-state Markov chain equation

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{D.43}$$

and write it in the form

$$(\mathbf{P} - \mathbf{I})\ \mathbf{s} = \mathbf{0} \tag{D.44}$$

$$\mathbf{A}\,\mathbf{s} = \mathbf{0} \tag{D.45}$$

where $\mathbf{0}$ is a zero column vector. Thus, finding $\mathbf{s}$ amounts to solving a homogeneous system of $n$ linear equations in $n$ unknowns. The system has a nontrivial solution if the determinant of $\mathbf{A}$ is zero. This is assured here since the determinant of the matrix is equal to the product of its eigenvalues. Here $\mathbf{A}$ has a zero eigenvalue which results in a zero determinant and this guarantees finding a nontrivial solution.

Applying a series of Givens rotations [5] will transform $\mathbf{A}$ into an upper triangular matrix $\mathbf{T}$ such that

$$\mathbf{T}\,\mathbf{s} = \mathbf{0} \tag{D.46}$$

We are now able to do back-substitution to find all the elements of $\mathbf{s}$. The last row of $\mathbf{T}$ could be made identically zero since the rank of $\mathbf{T}$ is $n - 1$. We start our back-substitution by ignoring the last row of $\mathbf{T}$ and assuming an arbitrary value for $s_n = 1$ then proceed to evaluate $s_{n-1}$ and so on. There still remains one equation that must be satisfied

$$\sum_{i=1}^{n} s_i = 1 \tag{D.47}$$

Let us assume that the sum of the components that we obtained for the vector **s** gives

$$\sum_{i=1}^{n} s_i = b \tag{D.48}$$

then we must divide each value of **s** by $b$ to get the true normalized vector that we desire.

*Example D.2.* Use Givens method to find the equilibrium state vector state **s** for the Markov chain with transition matrix given by

$$\mathbf{P} = \begin{bmatrix} 0.4 & 0.2 & 0 \\ 0.1 & 0.5 & 0.6 \\ 0.5 & 0.3 & 0.4 \end{bmatrix}$$

Step 1 Obtain the matrix $\mathbf{A} = \mathbf{P} - \mathbf{I}$

$$\mathbf{A} = \begin{bmatrix} -0.6 & 0.2 & 0.0 \\ 0.1 & -0.5 & 0.6 \\ 0.5 & 0.3 & -0.6 \end{bmatrix}$$

Step 2 Create a zero in the (2, 1) position using the Givens rotation matrix $\mathbf{G}_{21}$

$$\mathbf{G}_{21} = \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ with } \theta = \tan^{-1} \frac{0.1}{-0.6}$$

which gives

$$\mathbf{T}_1 = \mathbf{G}_{21}\mathbf{A} = \begin{bmatrix} 0.6083 & -0.2795 & 0.0986 \\ 0 & 0.4603 & -0.5918 \\ 0.5 & 0.3 & -0.6 \end{bmatrix}$$

Step 3 Create a zero in the (3, 1) position using the Givens rotation matrix $\mathbf{G}_{31}$

$$\mathbf{G}_{31} = \begin{bmatrix} c & 0 & s \\ 0 & 1 & 0 \\ -s & 0 & c \end{bmatrix} \text{ with } \theta = \tan^{-1} \frac{0.5}{0.6083}$$

which gives

$$\mathbf{T}_2 = \mathbf{G}_{31}\mathbf{T}_1 = \begin{bmatrix} 0.7874 & -0.0254 & -0.3048 \\ 0 & 0.4603 & -0.5918 \\ 0 & 0.4092 & -0.5261 \end{bmatrix}$$

Step 4 Create a zero in the (3, 2) position using the Givens rotation matrix $\mathbf{G}_{32}$

$$\mathbf{G}_{32} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{bmatrix} \text{ with } \theta = \tan^{-1} \frac{0.4092}{0.4603}$$

which gives

$$\mathbf{T}_3 = \mathbf{G}_{32}\mathbf{T}_2 = \begin{bmatrix} 0.7874 & -0.0254 & -0.3048 \\ 0 & 0.6159 & -0.7919 \\ 0 & 0 & 0 \end{bmatrix}$$

Step 5 Solve for the components of $\mathbf{s}$ assuming $s_3 = 1$

$$\mathbf{s} = \begin{bmatrix} 0.3871 & 1.2857 & 1 \end{bmatrix}$$

After normalization, the true state vector becomes.

$$\mathbf{s} = \begin{bmatrix} 0.1448 & 0.4810 & 0.3741 \end{bmatrix}$$

## D.17 Iterative Techniques

Iterative techniques continually refine the estimate of the solution while at the same time suppressing computation noise. Thus, the answer could be accurate within the machine precision. Iterative techniques are used when the system matrix is large and sparse. This is a typical situation in Markov chains and queuing theory.

### *D.17.1 Jacobi Iterations*

We start with the steady-state Markov chain equation

$$\mathbf{P}\,\mathbf{s} = \mathbf{s} \tag{D.49}$$

The matrix $\mathbf{P}$ is broken down as the sum of three components

$$\mathbf{P} = \mathbf{L} + \mathbf{D} + \mathbf{U} \tag{D.50}$$

where **L** is the lower triangular part of **P**, **D** is the diagonal part of **P**, and **U** is the upper triangular part of **P**. Thus, our steady-state equation can be written as

$$(\mathbf{L} + \mathbf{D} + \mathbf{U})\ \mathbf{s} = \mathbf{s} \tag{D.51}$$

We get Jacobi iterations if we write the above equation as

$$\mathbf{D}\,\mathbf{s} = (\mathbf{I} - \mathbf{L} - \mathbf{U})\ \mathbf{s} \tag{D.52}$$

where **I** is the unit matrix and it is assumed that **P** has nonzero diagonal elements. The technique starts with an assumed solution $\mathbf{s}^0$ then iterates to improve the guess using the iterations

$$\mathbf{s}^{k+1} = \mathbf{D}^{-1}(\mathbf{I} - \mathbf{L} - \mathbf{U})\ \mathbf{s}^k \tag{D.53}$$

Each component of **s** is updated according to the equation

$$s_i^{k+1} = \frac{1}{p_{ii}} \left( s_i^k - \sum_{j=0}^{i-1} p_{ij}\ s_j^k - \sum_{j=i+1}^{n-1} p_{ij}\ s_j^k \right) \tag{D.54}$$

### *D.17.2 Gauss–Seidel Iterations*

We get Gauss–Seidel iterations if we write (D.51) as

$$(\mathbf{D} + \mathbf{L})\ \mathbf{s} = (\mathbf{I} - \mathbf{U})\ \mathbf{s} \tag{D.55}$$

The technique starts with an assumed solution $\mathbf{s}^0$ then iterates to improve the guess using the iterations

$$\mathbf{s}^{k+1} = (\mathbf{D} + \mathbf{L})^{-1}\ (\mathbf{I} - \mathbf{U})\ \mathbf{s}^k \tag{D.56}$$

Each component of **s** is updated according to the equation

$$s_i^{k+1} = \frac{1}{p_{ii}} \left( s_i^k - \sum_{j=0}^{i-1} p_{ij}\ s_j^{k+1} - \sum_{j=i+1}^{n-1} p_{ij}\ s_j^k \right) \tag{D.57}$$

Notice that we make use of previously updated components of **s** to update the next components as evident by the iteration index associated with the first summation term (compare that with Jacobi iterations).

### *D.17.3 Successive Overrelaxation Iterations*

We explain successive overrelaxation (SOR) technique by rewriting a slightly modified version of (D.57)

$$s_i^{k+1} = s_i^k + \frac{1}{p_{ii}} \left( s_i^k - \sum_{j=0}^{i-1} p_{ij}\ s_j^{k+1} - \sum_{j=i}^{n-1} p_{ij}\ s_j^k \right) \tag{D.58}$$

We can think of the above equation as updating the value of $s_i^k$ by adding the term in brackets. We multiply that term by a *relaxation parameter* $\omega$ to get the *SOR* iterations

$$s_i^{k+1} = s_i^k + \frac{\omega}{p_{ii}} \left( s_i^k - \sum_{j=0}^{i-1} p_{ij}\ s_j^{k+1} - \sum_{j=i}^{n-1} p_{ij}\ s_j^k \right) \tag{D.59}$$

when $\omega = 1$, we get Gauss–Seidel iterations again of course.

## D.18 Similarity Transformation

Assume there exists a square matrix $\mathbf{M}$ whose inverse is $\mathbf{M}^{-1}$. Then the two square matrices $\mathbf{A}$ and $\mathbf{B} = \mathbf{M}^{-1}\mathbf{A}\mathbf{M}$ are said to be similar. We say that $\mathbf{B}$ is obtained from $\mathbf{A}$ by a *similarity transformation*. Similar matrices have the property that they both have the same eigenvalues. To prove that assume $\mathbf{x}$ is the eigenvector of $\mathbf{A}$

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{x} \tag{D.60}$$

We can write the above equation in the form

$$\mathbf{A}\ \mathbf{M}\mathbf{M}^{-1}\ \mathbf{x} = \lambda \mathbf{x} \tag{D.61}$$

Premultiply both sides of this equation by $\mathbf{M}^{-1}$ we obtain

$$\mathbf{M}^{-1} A\ \mathbf{M}\mathbf{M}^{-1}\ x = \lambda \mathbf{M}^{-1} x \tag{D.62}$$

But $\mathbf{B} = \mathbf{M}^{-1}\mathbf{A}\mathbf{M}$, by definition, and we have

$$\mathbf{B}\left(\mathbf{M}^{-1} x\right) = \lambda\left(\mathbf{M}^{-1} x\right) \tag{D.63}$$

Thus, we proved that the eigenvectors and eigenvalues of $\mathbf{B}$ are $\mathbf{M}^{-1}\mathbf{x}$ and $\lambda$, respectively.

Thus we can say that the two matrices **A** and **B** are similar when their eigenvalues are the same and their eigenvectors are related through the matrix **M**.

## D.19 Special Matrices

The following is an alphabetical collection of special matrices that were encountered in this book.

### *D.19.1 Circulant Matrix*

A square circulant matrix has the form

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \tag{D.64}$$

Premultiplying a matrix by a circulant matrix results in circularly shifting all the rows down by one row and the last row will become the first row.

An $m \times m$ circulant matrix **C** has the following two interesting properties:

Repeated multiplication $m$-times produces the identity matrix

$$\mathbf{C}^k = \mathbf{I} \tag{D.65}$$

where $k$ is an integer multiple of $m$.

Eigenvalues of the matrix all lie on the unit circle

$$\lambda_i = \exp\left(j2\pi \times \frac{k_i}{m}\right) \qquad k_i = 1,\ 2,\ \cdots,\ m \tag{D.66}$$

where $j = \sqrt{-1}$ and

$$|\lambda_i| = 1 \tag{D.67}$$

### D.19.2 Diagonal Matrix

A diagonal matrix has $a_{ij} = 0$ whenever $i \neq j$ and $a_{ii} = d_i$. A $3 \times 3$ diagonal matrix $\mathbf{D}$ is given by

$$\mathbf{D} = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} \tag{D.68}$$

An alternative way of defining $\mathbf{D}$ is

$$\mathbf{D} = \text{diag}\begin{pmatrix} d_1 & d_2 & d_3 \end{pmatrix} \tag{D.69}$$

or

$$\mathbf{D} = \text{diag}\,(\mathbf{d}) \tag{D.70}$$

where $\mathbf{d}$ is the vector of diagonal entries of $\mathbf{D}$.

### D.19.3 Echelon Matrix

An $m \times n$ matrix $\mathbf{A}$ can be transformed using elementary row operations into an upper triangular matrix $\mathbf{U}$, where elementary row operations include

1. Exchanging two rows.
2. Multiplication of a row by a nonzero constant.
3. Addition of a constant multiple of a row to another row.

Such operations are used in Givens rotations and Gauss elimination to solve the system of linear equations

$$\mathbf{Ax} = \mathbf{b} \tag{D.71}$$

by transforming it into

$$\mathbf{Ux} = \mathbf{c} \tag{D.72}$$

then doing back-substitution to find the unknown vector **x**. The matrix **U** that results is in echelon form. An example of a $4 \times 6$ echelon matrix is

$$\mathbf{U} = \begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix} \tag{D.73}$$

Notice that the number of leading zeros in each row must increase. The number of pivots[1] equals the rank $r$ of the matrix.

### D.19.4 Identity Matrix

An identity matrix has $a_{ij} = 0$ whenever $i \neq j$ and $a_{ii} = 1$. A $3 \times 3$ identity matrix **I** is given by

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{D.74}$$

### D.19.5 Nonnegative Matrix

An $m \times n$ matrix **A** is nonnegative when

$$a_{ij} \geq 0 \tag{D.75}$$

for all $1 \leq i \leq m$ and $1 \leq j \leq n$

### D.19.6 Orthogonal Matrix

An orthogonal matrix has the property

$$\mathbf{A}\mathbf{A}^t = \mathbf{I} \tag{D.76}$$

The inverse of **A** is trivially computed as $\mathbf{A}^{-1} = \mathbf{A}^t$. The inverse of an orthogonal matrix equals its transpose. Thus if **G** is an orthogonal matrix, then we have by definition

$$\mathbf{G}^t\mathbf{G} = \mathbf{G}^{-1}\mathbf{G} = \mathbf{I} \tag{D.77}$$

[1] A pivot is the leading nonzero element in each row.

It easy to prove that the Givens matrix is orthogonal. A matrix $\mathbf{A}$ is invertible if there is a matrix $\mathbf{A}^{-1}$ such that

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I} \tag{D.78}$$

### D.19.7 Plane Rotation (Givens) Matrix

A $5 \times 5$ plane rotation (or Givens) matrix is one that looks like the identity matrix except for elements that lie in the locations $pp$, $pq$, $qp$, and $qq$. Such a matrix is labeled $\mathbf{G}_{pq}$. For example, the matrix $\mathbf{G}_{42}$ takes the form

$$\mathbf{G}_{42} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c & 0 & s & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -s & 0 & c & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{D.79}$$

where $c = \cos\theta$ and $s = \sin\theta$. This matrix is orthogonal. Premultiplying a matrix $\mathbf{A}$ by $\mathbf{G}_{pq}$ modifies only rows $p$ and $q$. All other rows are left unchanged. The elements in rows $p$ and $q$ become

$$a_{pk} = ca_{pk} + sa_{qk} \tag{D.80}$$

$$a_{qk} = -sa_{pk} + ca_{qk} \tag{D.81}$$

### D.19.8 Stochastic (Markov) Matrix

A column stochastic matrix $\mathbf{P}$ has the following properties.

1. $a_{ij} \geq 0$ for all values of $i$ and $j$.
2. The sum of each column is exactly 1 (i.e., $\sum_{j=1}^{m} p_{ij} = 1$).

Such a matrix is termed *column stochastic* matrix or *Markov* matrix. This matrix has two important properties. First, all eigenvalues are in the range $-1 \leq \lambda \leq 1$. Second, at least one eigenvalue is $\lambda = 1$.

### D.19.9 Substochastic Matrix

A column substochastic matrix $\mathbf{V}$ has the following properties.

1. $a_{ij} \geq 0$ for all values of $i$ and $j$.
2. The sum of each column is less than 1 (i.e. $\sum_{j=1}^{m} p_{ij} < 1$).

Such a matrix is termed *column substochastic* matrix. This matrix has the important property that all eigenvalues are in the range $-1 < \lambda < 1$.

### D.19.10 Tridiagonal Matrix

A tridiagonal matrix is both upper and lower Hessenberg. That means that the only nonzero elements exist only on the main diagonal and the adjacent upper and lower subdiagonals. A $5 \times 5$ tridiagonal matrix $\mathbf{A}$ is given by

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix} \tag{D.82}$$

### D.19.11 Upper Hessenberg Matrix

An Upper Hessenberg matrix has $h_{ij} = 0$ whenever $j < i - 1$. A $5 \times 5$ upper Hessenberg matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \\ h_{21} & h_{22} & h_{23} & h_{24} & h_{25} \\ 0 & h_{32} & h_{33} & h_{34} & h_{35} \\ 0 & 0 & h_{43} & h_{44} & h_{45} \\ 0 & 0 & 0 & h_{54} & h_{55} \end{bmatrix} \tag{D.83}$$

A matrix is a lower Hessenberg matrix if its transpose is an upper Hessenberg matrix.

## References

1. G. Strang, *Introduction to Linear Algebra* (Wellesley-Cambridge Press, Wellesley, MA, 1993)
2. F.E. Hohn, *Elementary Matrix Algebra* (MacMillan, New York, 1964)
3. W.H. Press, S.T. Teukolsky, W.T. Vetterling, B.P. Fhannery, *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge University Press, Cambridge, 1992)
4. E. Kreyszig, *Advanced Engineering Mathematics* (Wiley, New York, 1988)
5. I. Jacques, C. Judd, *Numerical Analysis* (Chapman and Hall, New York, 1987)
6. R.E. Blahut, *Fast Algorithms for Digital Signal Processing* (Addison-Wesley, Reading, MA, 1985)
7. D. Arnold, MATLAB and RREF. http://online.redwoods.cc.ca.us/instruct/darnold/LinAlg/rref/context-rref-s.pdf

# Index





F. Gebali, *Analysis of Computer Networks*, DOI 10.1007/978-3-319-15657-6