Signals and Communication Technology

T. Muth

G. Ferrari (Ed.)

Sensor Networks

Where Theory Meets Practice

Springer

Springer Series on
SIGNALS AND COMMUNICATION TECHNOLOGY

SIGNALS AND COMMUNICATION TECHNOLOGY

Sensor Networks
G. Ferrari (Ed.)
ISBN 978-3-642-01340-9

Grid Enabled Remote Instrumentation
F. Davoli, N. Meyer, R. Pugliese, S. Zappatore
ISBN 978-0-387-09662-9

Usability of Speech Dialog Systems
Th. Hempel
ISBN 978-3-540-78342-8

Handover in DVB-H
X. Yang
ISBN 978-3-540-78629-0

Multimodal User Interfaces
D. Tzovaras (Ed.)
ISBN 978-3-540-78344-2

Wireless Sensor Networks and Applications
Y. Li, M.T. Thai, W. Wu (Eds.)
ISBN 978-0-387-49591-0

Passive Eye Monitoring
R.I. Hammoud (Ed.)
ISBN 978-3-540-75411-4

Digital Signal Processing
S. Engelberg
ISBN 978-1-84800-118-3

Digital Video and Audio Broadcasting Technology
W. Fischer
ISBN 978-3-540-76357-4

Satellite Communications and Navigation Systems
E. Del Re, M. Ruggieri (Eds.)
ISBN 978-0-387-47522-6

Three-Dimensional Television
H.M. Ozaktas, L. Onural (Eds.)
ISBN 978-3-540-72531-2

Foundations and Applications of Sensor Management
A.O. Hero III, D. Castañón, D. Cochran, and K. Kastella (Eds.)
ISBN 978-0-387-27892-6

Human Factors and Voice Interactive Systems, Second Edition
D. Gardner-Bonneau and H. Blanchard
ISBN 978-0-387-25482-1

Wireless Communications: 2007 CNIT Thyrrenian Symposium
S. Pupolin
ISBN 978-0-387-73824-6

Adaptive Nonlinear System Identification: The Volterra and Wiener Model Approaches
T. Ogunfunmi
ISBN 978-0-387-26328-1

Wireless Network Security
Y. Xiao, X. Shen, and D.Z. Du (Eds.)
ISBN 978-0-387-28040-0

Satellite Communications and Navigation Systems
E. Del Re and M. Ruggieri
ISBN 0-387-47522-2

Wireless Ad Hoc and Sensor Networks
A Cross-Layer Design Perspective
R. Jurdak
ISBN 0-387-39022-7

Cryptographic Algorithms on Reconfigurable Hardware
F. Rodriguez-Henriquez, N.A. Saqib, A. Díaz Pérez, and C.K. Koc
ISBN 0-387-33956-6

Multimedia Database Retrieval
A Human-Centered Approach
P. Muneesawang and L. Guan
ISBN 0-387-25627-X

Broadband Fixed Wireless Access
A System Perspective
M. Engels and F. Petre
ISBN 0-387-33956-6

Distributed Cooperative Laboratories
Networking, Instrumentation, and Measurements
F. Davoli, S. Palazzo and S. Zappatore (Eds.)
ISBN 0-387-29811-8

The Variational Bayes Method in Signal Processing
V. Šmídl and A. Quinn
ISBN 3-540-28819-8

Topics in Acoustic Echo and Noise Control
Selected Methods for the Cancellation of Acoustical Echoes, the Reduction of Background Noise, and Speech Processing
E. Hänsler and G. Schmidt (Eds.)
ISBN 3-540-33212-x

(continued after index)

Gianluigi Ferrari
Editor

Sensor Networks

Where Theory Meets Practice

Springer

Editor
Dr. Gianluigi Ferrari
Department of Information Engineering
University of Parma
Parma, Italy
gianluigi.ferrari@unipr.it

ISSN 1860-4862
ISBN 978-3-642-01340-9 e-ISBN 978-3-642-01341-6
DOI 10.1007/978-3-642-01341-6
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2009939071

Cover design: WMXDesign GmbH, Heidelberg

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To Sofia Elena
my beautiful sunshine

Preface

The idea of this book comes from the observation that sensor networks represent a topic of interest from both theoretical and practical perspectives. The title underlines that sensor networks offer the unique opportunity of clearly linking theory with practice. In fact, owing to their typical low-cost, academic researchers have the opportunity of implementing sensor network testbeds to check the validity of their theories, algorithms, protocols, etc., in reality. Likewise, a practitioner has the opportunity of understanding what are the principles behind the sensor networks under use and, thus, how to properly tune some accessible network parameters to improve the performance.

On the basis of the observations above, the book has been structured in three parts: Part I is denoted as "Theory," since the topics of its five chapters are apparently "detached" from real scenarios; Part II is denoted as "Theory and Practice," since the topics of its three chapters, altough theoretical, have a clear connection with specific practical scenarios; Part III is denoted as "Practice," since the topics of its five chapters are clearly related to practical applications.

In Part I, the first chapter, by H. V. Poor, examines ways in which interactions among nodes in wireless networks can lead to greater efficiencies in the use of wireless resources. The second chapter, by S. Sundhar Ram, V. V. Veeravalli, and A. Nedic, develops and analyzes distributed and recursive algorithms for nonlinear regression-based parameter estimation in sensor networks. The third chapter, by A. Prasath, A. Venuturumilli, A. Ranganathan, and A. A. Minai, describes a series of distributed methods for self-organized configuration of heterogeneous sensor networks, and evaluates their performance using optimal configurations obtained through an evolutionary algorithm. The fourth chapter, by A. Scaglione, Y.-W. P. Hong, and B. S. Mergen, discusses cooperative source and channel coding strategies and their performance in very dense sensor networks. The fifth and final chapter of this part, by C. Fragouli, K. Argyraki, and L. Keller, provides an overview of existing and emerging sensor network protocols that enable reliable communication in the presence of node and channel failures under energy constraints.

In Part II, the first chapter, by S. Palazzo, F. Cuomo, and L. Galluccio, introduces a taxonomy of the main approaches for data aggregation recently proposed in wireless sensor networks and presents a comparison between different data aggregation perspectives. The second chapter, by J. Lu, Y. Pan, S. Yamamoto, and T. Suda,

studies data dissemination resilient to large scale sensor failures by replicating data to a set of sensors forming a geographical trajectory. The third and final chapter of this part, by D. Akselrod, T. Lang, M. McDonald, and T. Kirubarajan, focuses on a problem of decision-based control of a network of sensors carrying out surveillance over a region that includes a number of moving targets, with particular application to multisensor multitarget tracking.

In Part III, the first chapter, by J. Beutel, K. Roemer, M. Ringwald, and M. Woehrle, surveys prominent examples of deployment techniques for sensor networks, identifies and classifies causes for errors and pitfalls, and presents a number of wireless sensor network specific techniques and tools for an increased understanding of the causes of such failures. The second chapter, by J.-M. Dricot, G. Bontempi, and P. De Doncker, begins with a survey of state-of-the-art techniques for localization and then proceeds to more elaborate and recent approaches, based on machine learning, automatic data classification, and sensor fusion techniques. In the third chapter, J.-H. Cui, R. Ammar, Z. Shi, Z. Zhou, S. Ibrahim, and H. Yan, investigates how to smartly utilize surface radios to enhance capability-limited underwater acoustic sensor networks. In the fourth chapter, by M. C. Vuran and A. R. Silva, theoretical and practical insights on the problem of communication through soil for wireless underground sensor networks is provided. The fifth and final chapter of this part, by D. McIlwraith and G.-Z. Yang, discusses recent advances in body sensor networks which permit pervasive sensing of detailed physiological signals from implantable, wearable, and ambient sensors.

Finally, I would like to thank those who have made the realization of this book possible. First of all, I am indebted to Dr. Cristoph Bauman, my Springer Engineering Editor, for supporting the idea of this book from the very first discussion at the beginning of 2008. Needless to say, all researchers that have contributed to this book are kindly acknowledged: without them, this would not have been possible – I hope that the fact that they are all from academia will not disproof the very idea of the book. Last, but not least, my sincere gratitude goes to a few members of our little Wireless Ad-hoc and Sensor Networks (WASN) Lab at the University of Parma, namely Marco Martalò, Paolo Medagliani, and Stefano Busanelli, for their help in the editing process and for sharing the interest in sensor networking.

Parma, Italy Gianluigi Ferrari
November 16, 2009

Contents

Contributors

David Akselrod ECE Department, McMaster University, Hamilton, ON, Canada, akselrd@mcmaster.ca

Reda Ammar Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA, reda@eng2.uconn.edu

Katerina Argyraki School of Computer and Communication Sciences, EPFL, Lausanne, Switzerland, katerina.argyraki@epfl.ch

Jan Beutel Institute for Computer Engineering and Networks Lab, ETH Zurich, Zurich, Switzerland, beutel@tik.ee.ethz.ch

Gianluca Bontempi Université Libre de Bruxelles – ULB Machine Learning Group, Bruxelles, Belgium, gbonte@ulb.ac.be

Jun-Hong Cui Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA, jcui@engr.uconn.edu

Francesca Cuomo INFOCOM Department, "La Sapienza", University of Rome, Rome, Italy, francesca.cuomo@uniroma1.it

Philippe De Doncker Université Libre de Bruxelles – OPERA Department, Wireless Communications Group, Bruxelles, Belgium, pdedonck@ulb.ac.be

Jean-Michel Dricot Université Libre de Bruxelles – OPERA Department, Wireless Communications Group, Bruxelles, Belgium, jdricot@ulb.ac.be

Christina Fragouli School of Computer and Communication Sciences, EPFL, Lausanne, Switzerland, christina.fragouli@epfl.ch

Laura Galluccio Dipartimento di Ingegneria Informatica e delle Telecomunicazioni, University of Catania, Catania, Italy, laura.galluccio@diit.unict.it

Y.-W. Peter Hong Electrical and Computer Engineering, NTHU, Hsinchu, Taiwan, ywhong@ee.nthu.edu.tw

Saleh Ibrahim Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA, saleh.ibrahim@engr.uconn.edu

Lorenzo Keller School of Computer and Communication Sciences, EPFL, Lausanne, Switzerland, lorenzo.keller@epfl.ch

Thiagalingam Kirubarajan ECE Department, McMaster University, Hamilton, ON, Canada, kiruba@mcmaster.ca

Thomas Lang General Dynamics Canada Ltd., Ottawa, ON, Canada, Tom.Lang@gdcanada.com

Jun Lu School of Information and Computer Sciences, University of California, Irvine, CA, USA, lujun@ics.uci.edu

Michael McDonald Surveillance Radar Group, Defence R&D Canada, Ottawa, ON, Canada, Mike.Mcdonald@drdc-rddc.gc.ca

Douglas McIlwraith Royal Society/Wolfson MIC Laboratory, Department of Computing, Imperial College London, London, UK, dm05@doc.ic.ac.uk

Ali A. Minai ECE Department, University of Cincinnati, Cincinnati, OH, USA, Ali.Minai@uc.edu

Angelina Nedić Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA, angelia@illinois.edu

Sergio Palazzo Dipartimento di Ingegneria Informatica e delle Telecomunicazioni, University of Catania, Catania, Italy, sergio.palazzo@diit.unict.it

Yi Pan School of Information and Computer Sciences, University of California, Irvine, CA, USA, ypan@ics.uci.edu

Arun Prasath ECE Department, University of Cincinnati, Cincinnati, OH, USA, news4arun@gmail.com

H. Vincent Poor Princeton University, Princeton, NJ, USA, poor@princeton.edu

Aravind Ranganathan CS Department, University of Cincinnati, Cincinnati, OH, USA, rangana@email.uc.edu

Matthias Ringwald Institute for Pervasive Computing, ETH Zurich, Zurich, Switzerland, mringwal@inf.ethz.ch

Kay Römer Institute for Pervasive Computing, ETH Zurich, Zurich, Switzerland; Institute for Computer Engineering, University of Luebeck, Luebeck, Germany, roemer@inf.ethz.ch

Anna Scaglione Electrical and Computer Engineering, UC Davis, Davis, CA, USA, ascaglione@ucdavis.edu

Birsen Sirkeci Mergen Electrical and Computer Engineering, San Jose State University, San Jose, CA, USA, bsirkeci@email.sjsu.edu

Zhijie Shi Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA, zshi@engr.uconn.edu

Agnelo R. Silva Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE, USA, asilva@cse.unl.edu

Tatsuya Suda School of Information and Computer Sciences, University of California, Irvine, CA, USA, suda@ics.uci.edu

Srinivasan Sundhar Ram Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA, ssriniv5@illinois.edu

Venugopal V. Veeravalli Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA, vvv@illinois.edu

Abhinay Venuturumilli ECE Department, University of Cincinnati, Cincinnati, OH, USA; Microchip Technology, Chandler, AZ, USA, abhinay.venuturumilli@microchip.com

M. Can Vuran Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE, USA, mcvuran@cse.unl.edu

Matthias Woehrle ETH Zurich, Zurich, Switzerland, woehrle@tik.ee.ethz.ch

Satoshi Yamamoto Graduate School of Engineering, Osaka University, Osaka, Japan, satoshi@post.comm.eng.osaka-u.ac.jp

Hai Yan Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA, hai.yan@engr.uconn.edu

Guang-Zhong Yang Institute of Biomedical Engineering, Imperial College London, UK, gzy@doc.ic.ac.uk

Zhong Zhou Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA, zhong.zhou@uconn.edu

Part I
Theory

Competition and Collaboration in Wireless Sensor Networks

H. Vincent Poor

Abstract Wireless sensor networks present a number of challenges to system designers, including notably the efficient use of limited resources such as bandwidth and energy. One way that such challenges have been addressed in recent years is through an examination of interactions among nodes that can lead to greater efficiencies in the use of wireless resources. This chapter examines two types of such interactions: competition among nodes in infrastructure networks, and collaboration among nodes in ad hoc networks. In the first context, the network is viewed as an economic system, in which sensors behave as agents competing for radio resources to optimize the energy efficiency with which they transmit messages. A game theoretic formalism is used to analyze the effects of various design choices and constraints on energy efficiency. In the second context, collaborative techniques for optimizing the use of radio resources are considered. Here, the focus is primarily on distributed inference, in which distinctive features of wireless sensor networks can be exploited through collaboration among nodes to effect a tradeoff between inferential accuracy and energy consumption.

1 Introduction

Over the past 20–25 years there has been considerable interest, and a number of major research developments, in the physical layer of wireless networks. Prominent examples include multiple-antenna (MIMO) systems [6], multiuser detection [39, 41], and turbo processing [32], to name just three. These major advances in the physical layer have made a major difference in the way we think about wireless networks at the link level. Although of course there remain many interesting and important open problems at the physical layer, since roughly the turn of this century interest in the area of wireless has shifted noticeably from the performance of individual links to the interactions among the nodes of the network. There are two basic modes of such interaction, competition and collaboration. Examples of competitive

H.V. Poor (✉)
Princeton University, Princeton, NJ, USA
e-mail: poor@princeton.edu

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_1,

behavior in wireless networks include cognitive radio, in which users compete for spectrum; information theoretic security, in which an intended recipient of a message is in essence competing with an eavesdropper; and the use of game theory in the modeling, analysis and design of networks, in which sensors compete for the physical resources of a network to accomplish their own objectives. Alternatively, the use of collaboration among the nodes of a wireless network has also gained currency. Notable examples include network coding, cooperative transmission and relaying, multi-hop networking, collaborative beam-forming, and collaborative inference. All of these topics are areas that have arisen relatively recently in the community and they involve primarily node interactions rather than link layer issues.

These issues play no less a role in the design and analysis of wireless sensor networks than in other type of wireless network, and in this chapter, we will examine these two modes of node interaction by considering an illustrative example of each type that is particularly relevant to sensor networking. Both of these examples are addressing the key issue of energy efficiency, one by examining data transfer in a relatively basic setting, and the other by looking at the overarching application of the network. The first of these problem is the area of energy games in multiple access networks, in which sensors compete for wireless resources so as to optimize their individual energy efficiencies in transmitting data to a common access point, or data collection site. The other is the problem of collaborative inference, in which sensor terminals collaborate locally to perform distributed inference without the need to communicate their data to a common (possibly distant) access point. Each of these models has a role in practical sensor networks, and which is most relevant to a given system depends on the particular network deployment. Nevertheless, taken together, these examples illustrate some of the basic principles and issues arising in node-level behavior of wireless sensor networks.

2 Energy Games in Multiple-Access Networks

We begin by considering the situation in which wireless sensors communicate their data to an access point via an infrastructure network using a multiple access protocol, as illustrated in Fig. 1. We can think of such a network as being like an economic system, in which the sensors behave as economic agents competing for resources in order to maximize their own utilities. (This view of network behavior is sometimes termed "economorphic" networking – e.g., V. Rodriguez, 2008, personal communication.) In this case, utility is based on the transfer of data up to the access point as efficiently as possible. Because this is a multiple access network, the actions of one sensor affect the utilities of the other sensors. So, we can model this situation as a competitive, or non-cooperative, game, and we can examine the particular situation in which the utility to the sensors is measured in terms of energy efficiency – that is, in bits-per-joule.

More specifically, we can think of a game having K players, where K is the number of sensors that are competing to communicate to the access point. Each

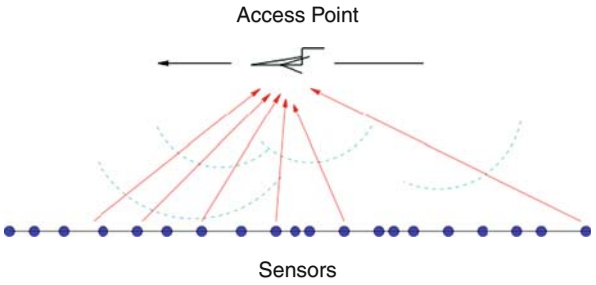


Fig. 1 Sensors transmit their data to a common access point via a shared (multiple-access) channel

sensor has a set of strategies, or actions ($\mathcal{A}_k$ for the kth sensor), and a utility function. Setting aside the set of strategies for now, as noted above the utility function of interest here is the energy efficiency, which can be measured as throughput divided by transmit power. That is, for sensor k, the utility can be written as

$$u_k = \frac{T_k}{p_k}, \tag{1}$$

where T_k denotes the sensor's throughput (or, goodput, as we are interested only in successful transmissions) and p_k denotes the sensor's transmit power. Since throughput is measured in units of bits per second, and transmit power in units of watts, or joules per second, the units of this utility are bits-per-joule, as desired.

The transmit power p_k in (1) is a very straightforward quantity, but the throughput needs to be modeled a bit more carefully in order to have an analytically tractable problem. Generally speaking, the throughput of a given sensor terminal is its transmission rate multiplied by its frame success rate; i.e., this is the goodput. Adopting a model from [15], the frame success rate can be modeled as an instantaneous function, say f, of the received signal-to-interference-plus-noise ratio (SINR) with which that users transmissions are received at the access point, post processing but before data detection. This is, of course, an idealization, since we know that the frame success rate is not always an exact function of the SINR. But this is a reasonable approximation for many modulation formats and receivers of interest. The nice thing about this model is that it allows us to abstract the physical layer into the function f. That is, we can lump the entire physical layer, except for the SINR, into f. So, f includes all the things you might think as lying in the physical layer: the modulation, the noise and channel models, the packet length, and so forth. Because the SINR for a given user contains the transmitted power of that user in its numerator, and the powers of all the other users in its denominator (recall that this is a multiple-access channel), we can see why this is a competitive game: if one user turns up its transmit power, it will positively influence its own SINR and negatively influence the SINRs of all the other users. So, this is essentially a game on SINR.

Before continuing, we note one minor problem with this model, and a corresponding solution to the problem. Namely, the frame success rate is not zero, even

when no signal is transmitted, as the receiver has some non-zero (albeit trivially small) probability of correctly guessing a transmitted packet even in the absence of observations. This means that the utility is unbounded as p_k approaches zero, and thus, any user can achieve infinite utility simply by transmitting nothing. A useful way around this problem is that suggested in [15], which is to replace the frame success rate with an approximation that is zero when zero power is transmitted. Thus, in this model, the function f will be taken to be of this latter type, in which case it is known as the *efficiency function*. A practically meaningful example is

$$f(\gamma) = \left(1 - e^{-\gamma}\right)^n \ , \tag{2}$$

where γ is the post-processing received SINR, and n is the packet length. The efficiency function of (2) approximates the frame success rate of binary phase shift keying (BPSK) operating over an additive white Gaussian noise (AWGN) channel.

Let us now examine a specific example of this game, in which we fix the access-point receiver to be a linear multiuser detector – say, a matched filter, a decorrelator (i.e., zero-forcing detector) or a minimum-mean-square-error (MMSE) detector (see, e.g., [39, 41]). The set of strategies available to each player in this game is to vary its transmitter power, up to a maximum possible value, so as to maximize its own utility; that is, this is a *power control* game. This game has been studied in [27] where it is shown that, if the function f is sigmoidal[1] then the game has a *Nash equilibrium* (i.e., an equilibrium at which no user can unilaterally improve its own utility). This equilibrium is achieved when each user chooses a transmit power to attain a certain target SINR, γ^*, which is given by the solution of the simple nonlinear scalar equation:

$$f\left(\gamma^*\right) \ = \ \gamma^* f'\left(\gamma^*\right) \ , \tag{3}$$

where f' denotes the derivative of f. So, for this problem, Nash equilibrium involves *SINR balancing*.

This game theoretic problem is in itself quite interesting. For example, the Nash equilibrium is unique, it has useful iterative properties, there are distributed algorithms, etc. But it also gives rise to a very useful analytical tool. That is, we can consider the value of utility at the Nash equilibrium to be a measure of the energy efficiency of the network. Thus, we can examine that equilibrium utility as it is affected by various network design choices. That is, the Nash equilibrium provides a design tool for comparing design choices on overall energy efficiency of the network. For example, we can compare different multiuser receiver choices by comparing the Nash equilibria arising from the use of those receivers.

Such a comparison was conducted in [27] for the case of BPSK signaling with a random code-division multiple-access (RCDMA) signaling protocol, and with flat

[1] Specifically, we assume that f is continuous, $f(0) = 0$, $\lim_{\gamma\to\infty} f(\gamma) = 1$, and that there is a value of γ such that f is convex to the left of that point and concave to its right. These conditions guarantee that the utility u_k is a quasi-concave function of p_k [27].

fading in the channels and AWGN at the access point. The access point is allowed to have multiple receive antennas. This model allows for closed-form evaluation of the utility (i.e., energy efficiency) at Nash equilibrium in the large-system asymptote when the number of terminals and system bandwidth both increase without bound, while their ratio (the system *load*, α) is held constant. Using the efficiency function of (2), the conclusion of this comparison is that significant improvement in the energy efficiency of the sensors can be achieved by proper choice of the access point receiver; in particular, the MMSE receiver offers considerable improvement over both the matched filter and the decorrelating detector. For example, at 60% load with the access point using two antennas, the MMSE detector provides twice the utility to the sensors that the decorrelator does, while the matched filter has zero utility at equilibrium. This means, with the access point using an MMSE detector, the sensors can use half the transmit power to achieve the same throughput as they would if the access point used a decorrelator. Since these two detectors have essentially the same complexity, the MMSE detector is clearly a superior design choice in this situation. Similarly, the addition of antennas at the access point can improve energy efficiency, at a rate that is approximately linear with the number of antennas. (Note that this latter conclusion is reminiscent of the similar well-known relationship between spectral efficiency and the number of antennas arising in MIMO systems [6].)

An interesting point about these conclusions is that the complexity of the receivers under consideration is incurred at the access point, whereas the energy efficiency accrues to the sensors. Typically, the sensors will be battery-powered, and hence are the most energy-needy, whereas energy efficiency and complexity at the access point are typically of less concern. Another interesting conclusion of this comparison, is that energy efficiency, represented by the utility (1), and spectral efficiency, represented by the load α, are opposing criteria. That is, increasing the increased energy efficiency generally comes at the expense of lower spectral efficiency, and vice versa. This phenomenon is consistent with information-theoretic analyses (e.g., [40]), which show that the least energy-per-bit in data transmission is achieved in many cases in the limit of zero spectral efficiency.

A natural question that arises here is why we would consider a competitive game in a situation in which there is an access point that could control the transmit power of the sensor terminals centrally. It turns out, though, that centralized solutions to this problem yield results similar to those described above, but with much greater analytical difficulty. For example, we can consider the Pareto, or socially optimal, solution, at which no user's utility can be improved without decreasing another's utility. Unlike the Nash equilibrium, which we have seen to be very easy to characterize and analyze, the Pareto solution is very difficult to determine, requiring numerical techniques. Moreover, while it is usually true that the Nash equilibrium is not generally Pareto optimal, it turns out to be quite close in this situation, as has been shown in [27]. So, even though we might think about using a centralized control here, the Nash equilibrium is much cleaner and it is almost the same as the more global, centralized equilibrium. Moreover, conclusions drawn from the analysis of the Nash equilibrium can be extrapolated to the behavior to be expected from centralized approaches where they can be implemented.

Of course, in wireless networks we are interested in more than just throughput, and in particular, we are interested in other measures of quality of service (QoS). Such measures can also be addressed through the above formulation. For example, the issue of delay QoS has been considered in this context in [25] and [26]. In [25], delay is modeled in units of packet re-transmissions, resulting in a version of the Nash game described above with SINR constraints. It is seen that, as one might expect, delay-sensitive users penalize their own utilities because of their need to contain the number of packet re-transmissions required for successful reception. But, moreover, the presence of delay-sensitive users also penalizes delay-tolerant users sharing the same network, as the increased power needed by the delay-sensitive users increases interference to the other users. These effects can be analyzed precisely, and the impact of different receiver choices on them is examined in [25]. In [26], a finite-delay model is examined by introducing queueing of packets at the terminals, resulting in a QoS model involving both (source) rate and delay. Here, an SINR-constrained game similarly arises, but the set of strategies for the terminals is expanded to include the control of transmit power and transmit rate (i.e., bandwidth). A new concept – that of the "size" of a user – arises in this formulation, and it is seen that a Nash equilibrium exists only when the total sizes of all users is less than one. This gives rise to the need for access control, which can be implemented through this notion of size, which is a measure of how much of the physical layer resources a user requires in order to achieve its specified QoS parameters. Again, a closed-form expression can be obtained for the equilibrium energy efficiency, and this is used in [26] to examine the effects on energy efficiency of rate and delay constraints.

This Nash formulation of energy-efficiency in multiple access networks has been applied to a number of other network design considerations. (An overview of many such studies is found in [24].) For example, in the area of receiver design, recent work has generalized the analysis of linear multiuser receivers described above, in several directions. One of these is the analysis of nonlinear multiuser receivers [23], such as maximum likelihood, maximum a posteriori probability, and interference canceling receivers. Using the large-system RCDMA analysis used in [27], it is seen in this work that Nash equilibria in power control games again entail SINR balancing. Interestingly, and somewhat contrary to the situation for improving spectral efficiency, it is seen in numerical results that the best of such nonlinear receivers (i.e., the maximum likelihood detector) does not necessarily give out-size gains in energy efficiency over the MMSE receiver, particularly when one considers the relative complexity of these two approaches. Another receiver design problem is that of equalizer design in systems using ultra-wideband signaling [2–4], in which rich scatter must be considered. Here, another asymptotic formalism is introduced to consider the situation in which the equalizer length must grow in proportion to the system bandwidth in order to adequately exploit the rich scattering environment. Related results are found in [1] and [20].

Transmitter design for energy efficient data transmission has also been considered via the Nash game formalism. One problem of this nature is that of carrier loading for multi-carrier systems [21], in which the game-players' strategies include

not only the choice of total transmit power, but also how that power can be distributed among a set of possible carrier signals. Here, assuming the access point uses matched filter receivers, Nash equilibria do not always exist. When they do exist (which numerical results indicate happens with very high probability), sensors choose a single best carrier on which to transmit and then use SINR balancing. That is, only one carrier is used by each sensor. This is contrary to what one would normally do for spectral efficiency, which calls for carrier loading based on waterfilling [14]. Another transmitter design problem is that of adaptive modulation, considered in [22], in which the sensors choose their modulation indices for quadrature amplitude modulation. In this case, Nash equilibria call for transmitters to choose the lowest-order modulation that will support QoS requirements, and then use SINR balancing. Again, this design runs contrary to spectrally-efficient design, which would call for using the highest supportable modulation index. A further problem of this type is that in which sensors in a CDMA setting can choose their transmitter waveforms, which has been considered in [8], and in related work in [10]. Here, when loads are less than unity, transmitters choose orthogonal waveforms, as one might expect, and so the interesting aspects of this problem occur for overloaded systems. Buzzi and Poor [8] also considers interference canceling receivers, which do achieve substantial improvement over linear systems in the particular case of overloaded systems. (An overview of adaptive waveform adaptation based on game-theoretic analysis is found in [9].)

In summary of this approach, we see that capturing competition among sensor terminals through a game-theoretic analysis provides a very useful tool for examining the issue of energy efficiency in multiple-access wireless sensor networks. Before moving to consideration of a collaborative model, it is of interest to mention two other economorphic approaches, in which ideas about modes of competition from economics have been used for network design. The first of these is the use of *auctions* in relay systems, in which multiple sensors contend for the services of a relay in forwarding their messages to the access point [19]. A second is the use of multi-period *coalition games* to incentivize sensor terminals located near the access point in a multi-hop setting to forward messages of more distant terminals [17]. (An alternative game-theoretic analysis of multi-hop networks is found in [5].) More generally, these problems fit within the context of cross-layer design of wireless networks, and a number of related issues in this area are discussed in [11].

3 Collaborative Inference

Collaborative interactions among nodes are perhaps even more natural in wireless sensor networks than competitive ones are, because of two salient features of such networks. In particular, all sensor terminals in a wireless sensor network share a common application, namely inference (i.e., detection, estimation, mapping, etc.); and the information collected at different sensors is often correlated since closely spaced sensors will likely have highly correlated measurements of the physical phenomenon they are sensing. A consequence of these features is that the goals

of individual sensors can be subsumed to the goals of the overall network (a view me might call "sociomorphic"). Additionally, even more so than in other types of wireless networks, resources are often severely limited in wireless sensor networks because sensors are typically deployed in places where humans do not want to go or do not often go. So, batteries cannot be recharged and replaced easily in a sensor network. Collaboration among sensors addresses these features of wireless sensor networks by having sensors work together to make inferences while conserving the radio resources of the network. These considerations lead us to examine the problem of collaborative inference in wireless sensor networks, and in particular we will consider collaborative learning.

By way of background, we first review briefly the classical supervised learning problem, in which we start with a set of d-dimensional real-vector inputs (or observations) X and corresponding real-valued outputs (or targets) Y. The goal of inference is to predict Y from observation of X, and to accomplish this we would like to design a function g to map values of X into predictions of Y satisfying some criterion, for example, minimizing the mean-square error, $E\{|Y - g(X)|^2\}$. Of course, if we know the joint distribution of X and Y, the solution to this problem lies in the province of classical Bayesian inference; in the particular case of minimizing the mean-square error, the proper choice of g is the conditional mean of Y given X; i.e., $E\{Y|X\}$ (see, e.g., [31]). However, in many applications, and certainly in sensor networking where sensors are often distributed in unknown territory, the joint distribution between X and Y is unknown. So, we would like to use learning to construct a suitable function g from a set of examples (or exemplars), say $\{(x_i, y_i)\}_{i=1}^n$, which are realizations of the inputs paired with the corresponding correct outputs. This is the classical supervised learning problem, which has been very well studied (see, e.g., [18]).

Now, we turn specifically to learning in wireless sensor networks. Supposing that sensors are spatially distributed, we can think of a wireless sensor network as a distributed sampling device with a wireless interface, as illustrated in Fig. 2. And, as such, we can consider a learning problem in which each sensor measures some subset of the set of exemplars $\{(x_i, y_i)\}_{i=1}^n$, so that the exemplars are distributed around the network. In classical learning theory it is assumed that all of the exemplars can be processed jointly, which in our case would correspond to the situation in which all sensors transmit their exemplars to a central location to be processed. A wealth of methodology is available for this centralized problem. One such technique is smoothness-penalized least-squares function fitting within a reproducing kernel Hilbert space (RKHS) [38], which fits the function g by solving the problem

$$\min_{g \in \mathcal{H}_K} \left\{ \sum_{i=1}^{n} [g(x_i) - y_i]^2 \; + \; \lambda \parallel g \parallel_{\mathcal{H}_K}^2 \right\}, \tag{4}$$

where $\mathcal{H}_K$ denotes the RKHS associated with a suitably chosen reproducing kernel $K(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, $\| \cdot \|_{\mathcal{H}_K}^2$ denotes the norm in $\mathcal{H}_K$, and λ controls the smoothness penalty.

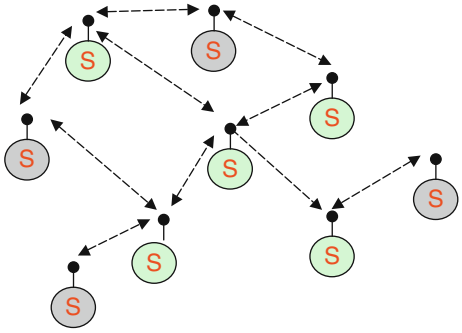


Fig. 2 Sensors share their data with their neighbors

Here, we wish to address the question of what can be done when the exemplars are not collected together, but rather are distributed throughout the network in the above fashion. So, we assume that energy and bandwidth constraints preclude the sensors from transferring their exemplars to a central collection point for processing, and thus they must cope with the data locally. A justification for this is the usual one for distributed networks; i.e., that local communication is efficient relative to global communication, and certainly in wireless networks that is true. Communication among neighbors requires less energy and bandwidth than does communication with a distant access point. This gives us the seed of a model, in which we have the data distributed according to a physical sensor network that gives rise to a neighborhood structure. These considerations lead to a fairly general model that applies to this situation and many others, and that shapes the nature of collaboration in such networks.

In particular, we consider a model in which there are a number, m, of learning agents which in this setting are sensors, and a number, n, of training examples. As illustrated in Fig. 3, the topology of the network can be captured in a bipartite graph in which the vertices in one set of vertices represent the m learning agents, the vertices in the other set of vertices represent the n training examples, and an edge extends from a learning agent vertex to a training example vertex if the corresponding learning agent has access to the corresponding exemplar. This model provides a general framework for examining distributed inference; it is quite general, encompassing fully centralized processing, decentralized processing with a public data-base, fully decentralized processing, etc., depending on the number of vertices in each part of the graph, and on the connectivity. For example, Fig. 4 illustrates the situation of most interest for sensor networks, in which the edges are determined by the physical topology of the sensor network. (See [35] for further details, and other examples.) But generally, to analyze this model we do not need to be concerned with its particular topology. We can simply think about a general bipartite graph consisting of learning agents connected to a training database, and examine inference in that context.

A natural approach to consider in such a setting is *local learning*, in which each learning agent learns according to kernel regression (4) applied to those exemplars

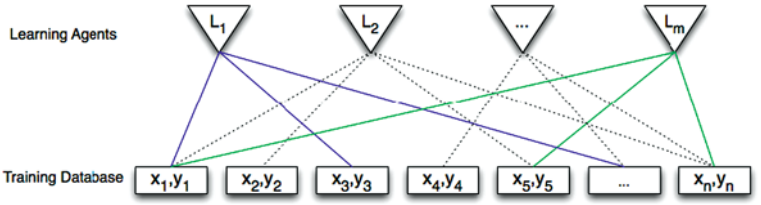


Fig. 3 A bipartite graph model for distributed learning

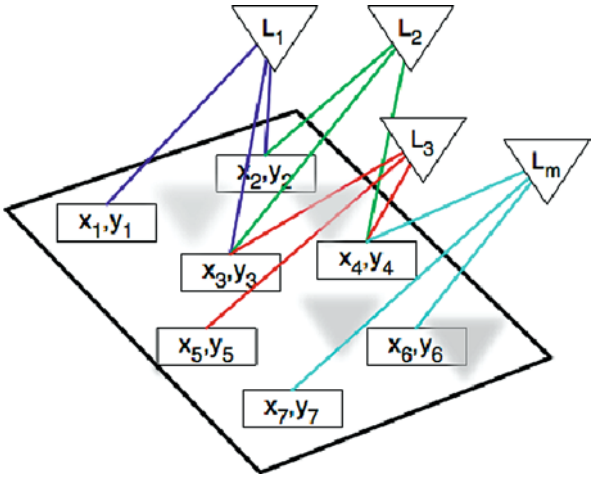


Fig. 4 Distributed learning model for a sensor network sensing a spatio-temporal field

to which it is directly connected; i.e., for local learning at a given learning agent, the summation in (4) is taken only over those exemplars in the training database to which that agent is connected via an edge. In this way, each learning agent creates a local model for what is going on in the entire field being sensed. A problem with local learning is that it is provable locally incoherent; that is, if two learning agents examine the same datum, their estimates may not agree on that datum. This is an undesirable property, and it can be shown that it generally leads to a penalty in terms of overall fitting error. So, a natural question that arises is whether there is some way for the agents to collaborate to eliminate this local incoherence, while retaining the efficiency of locality, i.e., without having to communicate more broadly than their neighborhoods. One way to do this is to begin with local learning, in which each of the learning agents performs a kernel regression locally. But rather than stopping there, after performing local regression, each agent then updates all of the exemplars to which it has access with its own estimates of the targets. This process can be further iterated over time in addition to iterating over sensors by adding an inertia-penalty to the regression.

Such a message-passing algorithm is proposed and analyzed in [35] for the MMSE regression problem, where it has been shown to have very favorable properties. In particular, the algorithm converges in norm to a relaxation of the centralized

kernel estimator as we iterate in time, and the limit is locally coherent. Moreover, the iterates lie in the span of a finite number of functions so that the estimation problem being solved at each iteration is actually parameter estimation rather than function estimation, which is very efficient computationally. Also, the global fitting error improves with every update. These properties hold for any bipartite graph of the form described above, regardless of whether it is connected or not. But, under a connectivity assumption and the assumption of independent and identically distributed exemplars, the local estimates can all be made to converge in RKHS norm to the conditional mean by proper programming of the penalty terms. Further description of this algorithm, together with some illustrative numerical examples, is found in [35].

The message-passing approach to collaborative inference of [35] provides a general formalism for designing and analyzing algorithms for collaborative inference in a distributed sensing environment. However, there are other ways in which the constraints of sensor networks can be addressed. For example, another approach to distributed learning is considered in [33], in which, rather than collaborating, sensors communicate directly back to an access point via links with very limited capacity, as illustrated in Fig. 5. Questions then arise regarding whether consistent learning of regression or classification functions can take place in this setting. These questions are answered in [33] largely in the affirmative, in that consistent regression is possible with the transmission of only $\log_2 3$ bits per sensor per decision, while consistent classification is possible with transmission of only a single bit per sensor per decision. A further problem of interest is judgment aggregation, considered in [36], which is related to the problem of local coherence mentioned earlier. This is a problem that arises in many applications beyond sensor networking, including the combination of judgments of individuals in panels of experts. A review of these and related issues in this general area is found in [34].

Another way in which wireless sensors can collaborate locally to save radio resources is via collaborative, or distributed, beam-forming [29], in which sensors that cluster together physically can collaborate to form a common message and then to form a beam on the access point. A number of techniques for implementing this type of beam-forming have been considered recently (see, e.g., [7, 12, 13, 16, 30, 37]), and [28] provides a recent survey of developments in this area.

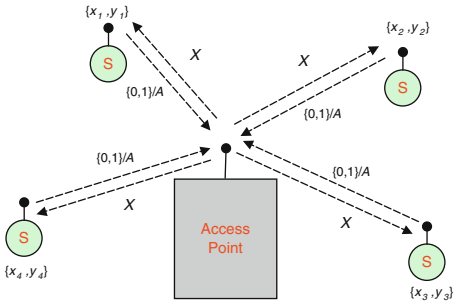


Fig. 5 Distributed learning model with limited capacity links to an access point

4 Conclusions

As can be seen from the above discussion, the two problems discussed here are quite rich in themselves. However, they are only representative of a large class of emerging research problems in which node interaction is the primary consideration. In discussing these issues, this chapter has focused principally on the work of the author and co-workers. However, there are many other important issues and interesting contributions in these areas, which can be found in the literature cited in the references, and in particular in the review papers [2, 9, 24, 28, 30, 34]. Readers interested in learning more about these subjects are referred to these works.

References

1. Bacci G, Luise M, Poor HV (2008) Energy efficient power control is (almost) equivalent for DS-CDMA and TH-UWB. Electr Lett 44(8):555–556.
2. Bacci G, Luise M, Poor HV (2008) Game theory and power control in ultrawideband networks. Phys Commun 1(1):21–39.
3. Bacci G, Luise M, Poor HV (2008) Performance of rake receivers in IR-UWB networks using energy-efficient power control. IEEE Trans Wireless Commun 7(6):2289–2299.
4. Bacci G, Luise M, Poor HV, Tulino A (2007) Energy-efficient power control in impulse radio UWB wireless networks. IEEE J Selected Topics Signal Process 1(3):508–520.
5. Betz S, Poor HV (2008) Energy efficiency in multi-hop CDMA networks: a game theoretic analysis considering operating costs. IEEE Trans Signal Process 56(10):5181–5190.
6. Biglieri E, Calderbank AR, Constantinides AG, Goldsmith A, Paulraj A, Poor HV (2007) MIMO Wireless Communications. Cambridge University Press, Cambridge, UK.
7. Brown DR, III, Poor HV (2008) Time-slotted round-trip carrier synchronization for distributed beamforming. IEEE Trans Signal Process 56(11):5630–5643.
8. Buzzi S, Poor HV (2008) Joint receiver and transmitter optimization for energy-efficient multiple-access communications. IEEE J Selected Areas Commun 26(3):459–472.
9. Buzzi S, Poor HV, Saturnino D (2009) Non-cooperative waveform adaptation games in multiple access communications. IEEE Signal Process Mag 26(5): 64–76.
10. Buzzi S, Saturnino D, Poor HV (2009) Adaptive cross-layer distributed energy-efficient resource allocation algorithms for wireless data networks. EURASIP J Advances Signal Process, Article ID 532607:14.
11. Comaniciu C, Mandayam N, Poor HV (2005) Wireless Networks: Multiuser Detection in Cross-Layer Design. Springer, New York.
12. Dong L, Petropulu A, Poor HV (2008) A cross-layer approach to collaborative beamforming for wireless ad hoc networks. IEEE Trans Signal Process 56(7):2981–2993.
13. Fan Y, Adinoyi A, Thompson JS, Yanikomeroglu H, Poor HV (2009) A simple distributed antenna processing scheme for cooperative diversity. IEEE Trans Commun 57(3):626–629.
14. Goldsmith A (2005) Wireless Communications. Cambridge University Press, Cambridge, UK.
15. Goodman DJ, Mandayam N (2000) Power control for wireless data. IEEE Pers Commun 7(2):48–54.
16. Han Z, Poor HV (2007) Lifetime improvement in wireless sensor networks via collaborative beamforming and cooperative transmission. IET Microw Antennas Propag 1(6):1103–1110.
17. Han Z, Poor HV (2009) Coalition games with cooperative transmission: a cure for the curse of boundary nodes in selfish packet-forwarding wireless networks. IEEE Trans Commun 57(1):203–213.
18. Hastie T, Tibshirani T, Friedman, J (2001) The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York.

19. Huang J, Han Z, Chiang M, Poor HV (2008) Auction-based resource allocation for cooperative communications,. IEEE J Selected Areas Commun 26(7):1226–1237.
20. Massaro V, Buzzi S, Poor HV (2009) Energy-efficient resource allocation in multipath CDMA channels with bandlimited waveforms. IEEE Trans Signal Process 57(4):1494–1510.
21. Meshkati F, Chiang M, Poor HV, Schwartz SC (2006) A game-theoretic approach to energy-efficient power control in multi-carrier CDMA systems. IEEE J Selected Areas Commun 24(6):1115–1129.
22. Meshkati F, Goldsmith A, Poor HV, Schwartz SC (2007) A game theoretic approach to energy-efficient modulation in CDMA networks with delay QoS constraints. IEEE J Selected Areas Commun 25(6):1069–1078.
23. Meshkati F, Guo D, Poor HV, Schwartz SC (2008) A unified approach to energy-efficient power control in large CDMA systems. IEEE Trans Wireless Commun 7(4):1208–1216.
24. Meshkati F, Poor HV, Schwartz SC (2007) Energy-efficient resource allocation in wireless networks. IEEE Signal Proc Mag 24(3):58–68.
25. Meshkati F, Poor HV, Schwartz SC (2009) Energy efficiency-delay tradeoffs in multiple-access networks. IEEE Trans Inform Theory 55(7): 3220–3228.
26. Meshkati F, Poor HV, Schwartz SC, Balan R (2009) Energy-efficient resource allocation in wireless networks with quality-of-service constraints. IEEE Trans Commun 57(11): 3406–3414.
27. Meshkati F, Poor HV, Schwartz SC, Mandayam N (2005) An energy-efficient approach to power control and receiver design in wireless data networks. IEEE Trans Commun 53(11):1885–1894.
28. Mudumbai R, Madhow M, Brown DR, III, Poor HV (2009) Distributed transmit beamforming: challenges and recent progress. IEEE Commun Mag 47(2):102–110.
29. Ochiai H, Mitran P, Poor HV, Tarokh V (2005) Collaborative beamforming for distributed wireless ad hoc sensor networks. IEEE Trans Signal Process 53(11):4110–4124.
30. Ochiai H, Mitran P, Poor HV, Tarokh V (2008) Random array theory and collaborative beamforming. In: Handbook on Advancements in Smart Antenna Technologies for Wireless Networks. Idea Group, Hershey, PA.
31. Poor HV (1994) An Introduction to Signal Detection and Estimation, 2nd edn. Springer, New York.
32. Poor HV (2004) Iterative multiuser detection. IEEE Signal Process Mag 21(1):81–88.
33. Predd JB, Kulkarni SR, Poor HV (2006) Consistency in models for distributed learning under communication constraints. IEEE Trans Inform Theory 52(1):52–63.
34. Predd JB, Kulkarni SR, Poor HV (2006) Distributed learning in wireless sensor networks. IEEE Signal Process Mag 23(4):56–69.
35. Predd JB, Kulkarni SR, Poor HV (2009) A collaborative training algorithm for distributed learning. IEEE Trans Inform Theory 55(4):1856–1871.
36. Predd JB, Osherson D, Kulkarni SR, Poor HV (2008) Aggregating forecasts of chance from incoherent and abstaining experts. Decis Anal 5(4):177–189.
37. Pun MO, Brown DR, III, Poor HV (2009) Opportunistic collaborative beamforming with one-bit feedback. IEEE Trans Wireless Commun 8(5):2629–2641.
38. Schölkopf B, Smola A (2002) Learning with Kernels. MIT Press, Cambridge, MA.
39. Verdú S (1998) Multiuser Detection. Cambridge University Press, Cambridge, UK.
40. Verdú S (2002) Spectral efficiency in the wideband regime. IEEE Trans Inform Theory 48(6):1319–1343.
41. Wang X, Poor HV (2004) Wireless Communication Systems: Advanced Techniques for Signal Reception. Prentice-Hall, Upper Saddle River, NJ.

Distributed and Recursive Parameter Estimation

Srinivasan Sundhar Ram, Venugopal V. Veeravalli, and Angelina Nedić

Abstract Parametric estimation is a canonical problem in sensor networks. The intrinsic nature of sensor networks requires regression algorithms based on sensor data to be distributed and recursive. Such algorithms are studied in this chapter for the problem of (conditional) least squares regression when the data collected across sensors is homogeneous, i.e., each sensor observes samples of the dependent and independent variable in the regression problem. The chapter is divided into three parts. In the first part, distributed and recursive estimation algorithms are developed for the nonlinear regression problem. In the second part, a distributed and recursive algorithm is designed to estimate the unknown parameter in a parametrized state-space random process. In the third part, the problem of identifying the source of a diffusion field is discussed as a representative application for the algorithms developed in the first two parts.

1 Introduction

Sensor networks consist of spatially deployed sensors that sense their local environment across time to learn some feature of interest about the underlying field. In many applications, using a priori information, it is possible to build approximate parametrized model sets for the feature of interest. In such cases, the problem of learning the feature simplifies to the problem of estimating these unknown parameters that determine the model that best fits the observed data.

For example, consider sensors deployed to determine the source of a spatiotemporal temperature field. A model set for the heat source could be the set of all point sources with constant intensity. The model set is parametrized by the source location and the constant intensity, and the sensors measurements have to be used to estimate these values. The diffusion equation that governs the propagation of heat can then be

S.S. Ram (✉)
Electrical and Computer Engineering, University of Illinois at Urbana–Champaign, Urbana, IL, USA
e-mail: ssriniv5@illinois.edu

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_2,

used to map the model for the heat source to a model for the sensor measurements. Note that the actual source may have an arbitrary shape and have an exponentially decreasing intensity. In effect, the goal is to only determine the best approximation for the source among all point sources with constant intensity.

There are multiple advantages to estimating the unknown parameters jointly using data collected by a network of sensors, rather than individual sensors. First, measurements made by a single sensor may not be sufficient to uniquely resolve the parameter. For example, triangulation of sources require distance measurements format least three sensors that are not on a straight line. Second, there are fundamental limits on the sampling rate of a sensor. By using multiple sensors, it is possible to obtain a higher effective sampling rate. Finally, even when multiple sensors add redundancy, they serve to mitigate the effect of noise in observations.

There are two basic steps in estimation. The first step is to use physical laws to map the models for the features to models for the sensor measurements. In the heat source example, the diffusion equation can be used to map the model for the heat source to a model for the sensor measurements. In this chapter, it is assumed that a parametrized model for the sensor measurements is available and the focus is only on the second step, which is to use statistical techniques to determine the form of the estimator.

The intrinsic nature of sensor networks impose certain challenges. When there are a large number of sensors spread across a large operational area, it is not energy efficient to jointly estimate the parameter by collecting all the measurements from the sensors at a fusion center. Instead, algorithms have to be distributed and the estimate must be calculated by the network through local communication between sensors and their neighbors, without the aid of a central fusion center. Further, each sensor has a limited memory and measurements have to be purged periodically. Thus, the estimation procedures should also be recursive, and only a constant summary statistic must be stored and updated when a new measurement is made. Thus estimation algorithms that are to be used in sensor networks have to be distributed and recursive. In this chapter, recursive and distributed procedures for (conditional) least squares estimation are discussed. The chapter is divided into three parts. In the first part, distributed and recursive estimation algorithms are developed for the nonlinear regression problem. In the second part, a distributed and recursive algorithm is designed to estimate the unknown parameter in a parametrized statespace random process. In the third part, the problem of identifying the source of a diffusion field is discussed as a representative application for the algorithms developed in the first two parts.

2 Preliminaries

The following notational convention is used. For a matrix A, $[A]_{i,j}$ denotes its (i, j)-th entry. All vectors are column vectors and the set consisting of the first k elements of a sequence $\{r_\ell\}_{\ell\in\mathbb{N}}$ is denoted by r^k. A total of m, $m \in \mathbb{N}$, sensors are

spatially deployed and are indexed using the set $V = \{1, \ldots, m\}$. Throughout this chapter the variable $i \in V$ and the variable $k \in \mathbb{N}$. The sensors sequentially and synchronously sense the underlying field once every time unit. The k-th observation made by the i-th sensor is denoted by $z_{i,k}$. We will find it convenient to view $\{z_{i,k}\}$ as a sample path of a random process $\{Z_{i,k}\}$.

We denote by x, $x \in \Re^n$, the unknown vector that is to be estimated. To aid in the estimation, a priori information is used to develop a collection of models, called a *model set*, that is parametrized by x. Typically, the random process $\{Z_k\}$, rather than the actual observed sample path, is modeled. Thus a model set can be denoted by $\{\hat{Z}_k(x); x \in X\}_{k\in\mathbb{N}}$, where for each x $\hat{Z}_k(x)$ is a model, i.e., "guess", for Z_k, and different models are obtained as x takes different values in the parameter set X.

In some applications, it is possible that the a priori information available is not sufficient to determine complete model sets for the process $\{Z_k\}$. One such specific case is regression. In this application, z_k is divided into two parts: a dependent component r_k, and an independent component u_k. The model set *only specifies* how $\{R_k\}$ depends on $\{U_k\}$. We will refer to such model sets as *regression model sets*[1] and they can be represented by $\{\hat{R}_k(x, U_i^k); x \in X\}_{k\in\mathbb{N}}$. We will study regression model sets. Note that there is no loss in generality and the traditional estimation setting is simply the extreme case when $R_k = Z_k$ and $U_k = \emptyset$.

A model set captures the information in a "convenient" form [15]. In the heat source example discussed earlier, we could have taken the model set to be the set of all rectangular sources with random intensity. A priori information may tell us that this model set contains a model which is a better approximation to the actual source than any model in the point source model set. However, one may still work with the point source model set since it may not be easy to determine the specific model in the rectangular source model set that is the best. In the context of sensor networks, "convenient" is to be interpreted as having the structure to allow distributed and recursive estimation. Thus we might possibly need to discard some a priori information and work with less-accurate model sets, compared to a centralized and non-recursive setting. This is the cost that is to be paid to obtain a distributed implementation. Specifically, we will discard any information available about the joint statistics[2] of the measurements. Thus we will consider model sets that *only* specify how $\{R_{i,k}\}$ depends on $\{U_{i,k}\}$. They can be represented as $\{\hat{R}_{i,k}(x, U_i^k);\ x \in X, i \in V\}_{k\in\mathbb{N}}$. We will refer to such model sets as *separable regression model sets*.

The estimate of the parameter is the value that determines the model in the model set that "best fits" the observed data. In this chapter, we will define "best" as the *conditional least-squares criterion*, described next. Define

$$p_{i,k+1}(x, u_i^{k+1}, r_i^k) = \mathsf{E}\big[\hat{R}_{i,k+1}(x, U_i^k) \mid U_i^{k+1} = u_i^{k+1}, \{\hat{R}_{i,\ell}(x, U_i^\ell) = r_{i,\ell}\}\big].$$

[1] When $U_k = R_{k-1}$, the model set is *auto-regressive*.

[2] Not knowing anything about the joint statistics of the sensor measurements, should not be confused with knowing that they are independent.

Essentially, $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ is the best estimate for $R_{i,k+1}$ based on the past and present measurements of the independent variable, and all past values of the dependent variable, when it is known that the true description of $\{R_{i,k+1}\}_{k\in\mathbb{N}}$ is $\{\hat{R}_{i,k}(x, U_i^k)\}_{k\in\mathbb{N}}$. The conditional least squares estimate (CLSE) of [12] is given by:

$$x_N^* = \underset{x\in X}{\operatorname{argmin}} \sum_{i=1}^{m} \frac{1}{N} \sum_{k=1}^{N} \left(r_{i,k+1} - p_{i,k+1}(x, u_i^{k+1}, r_i^k)\right)^2. \tag{1}$$

We refer the reader to [12] for a discussion of the properties of the CLSE.[3] The CLSE estimator has also been studied as the minimum prediction error estimate (MPEE) in [15].

When the sensors make a large number of measurements, a convenient abstraction is to consider this number to be infinite. Denote, by x^* the limit of x_N^*, i.e.,

$$x^* = \underset{x\in X}{\operatorname{argmin}} \sum_{i=1}^{m} \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \left(r_{i,k+1} - p_{i,k+1}(x, u_i^{k+1}, r_i^k)\right)^2. \tag{2}$$

The goal is to generate a sequence $\{x_n\}$ in a recursive and distributed manner that converges to x^*. Traditionally, the convergence is proved only when there exists a $\bar{x} \in X$, for which $\{\hat{Z}_k(\bar{x})\}$ accurately describes $\{Z_k\}$. This is restrictive and may not be reasonable, especially considering the that we would like to work with "convenient" model sets. Thus, we will study the properties under more general conditions. In the following two sections, we discuss two different classes of model sets:

- *Simple non-linear regression*. This corresponds to the case when $\hat{R}_{i,k}(x, U_i^k)$ is of the form $g_i(x, U_{i,k}) + E_{i,k+1}$, where $\{E_{i,k+1}\}$ is an i.i.d. sequence.
- *Stationary Gaussian linear state space model sets*. This corresponds to the case when $\hat{R}_{i,k}(x, U_i^k)$ is a parametrized stationary Gaussian linear state space process with input process $\{U_{i,k}\}$.

3 Simple Non-linear Regression

In the simple nonlinear regression problem, the model sets have the following form:

$$\hat{R}_{i,k}(x, U_i^k) = g_i(x, U_{i,k}) + E_{i,k}, \tag{3}$$

[3] When the function in (1) has multiple minima, then each minimum is a CLSE estimate. For example, this would be the case if less than three sensors were used in the acoustic source localization. In most cases, by increasing the number of sensors, and thus the spatial diversity of the measurements, it is possible to ensure that the estimation problem is well defined and there is a unique least squares estimate. Such an assumption would be analogous the observability condition of [10].

where $\{E_{i,k}\}$ is zero mean noise. The statistics of $\{E_{i,k}\}$ are not of any concern in determining the CLSE. For the model set in (3),

$$p_{i,k+1}(x, u_i^{k+1}, r_i^k) = g_i(x, u_{i,k+1}),$$

and note that the CLSE in (1) reduces to the well known least squares estimate (LSE),

$$x^* = \underset{x \in X}{\operatorname{argmin}} \lim_{N \to \infty} \frac{1}{N} \sum_{k=1}^{N} \sum_{i=1}^{m} \left(r_{i,k} - g_i(x, u_{i,k+1})\right)^2.$$

3.1 Algorithms

We next review three different distributed and recursive estimation algorithms that have been proposed in literature.

3.1.1 Cyclic Incremental Recursive Algorithm

Each sensor designates another sensor as an upstream neighbor and one as a downstream neighbor so that they form a cycle. See Fig. 1 for an example. Without loss of generality, we will index the upstream neighbor of sensor i as $i + 1$, with the understanding that the upstream neighbor of sensor m is a fictitious sensor, denote as sensor 0. Between any two consecutive sampling times, one iteration of the algorithm is performed. In iteration k, sensor i receives the current iterate $z_{i-1,k}$ from sensor $i - 1$, updates the iterate using its latest measurement and passes it to its upstream neighbor, sensor i. The update rule is

$$\begin{aligned} z_{0,k} &= z_{m,k-1} = x_{k-1}, \\ z_{i,k} &= P_X \left[z_{i-1,k} - 2\alpha_k \left(g_i(z_{i-1,k}, u_{i,k}) - r_{i,k} \right) \nabla g_i(z_{i-1,k}, u_{i,k}) \right], \end{aligned} \tag{4}$$

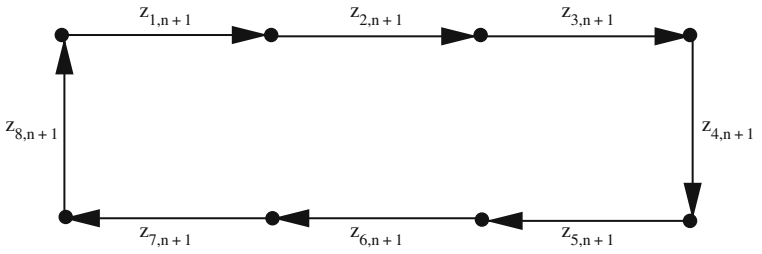


Fig. 1 A network of 8 sensors with cyclical incremental processing. The estimate is cycled through the network. The quantity $z_{i,k}$ is the intermediate value after sensor i updates at time k

where the initial iterate x_0 is chosen at random. Here, α_k is the stepsize, P_X denotes projection onto the set X and $\nabla g_i(x)$ denotes the gradient of the vector function g_i with respect to the vector parameter x. Thus, if x_j denotes the j-th component of the vector x,

$$\left[\nabla g_i(x, u_{i,k})\right]_{\ell,j} = \frac{\partial g_i^{(\ell)}(x, u_{i,k})}{\partial x_j}.$$

The cyclical incremental algorithm can be implemented only when each sensor identifies a suitable upstream and downstream neighbor. This could be achieved in a setup phase in a distributed manner using the algorithm proposed in [14]. Note the existence of a cycle is a stronger assumption than connectivity. We refer the reader to [23] for a discussion of other implementation issues.

3.1.2 Markov Incremental Recursive Algorithm

In this algorithm, the order in which the sensors update the iterate is not fixed and could be random. Let $N(i)$ denote the set of neighbors of sensor i and $|N(i)|$ denote the number of neighbors. Suppose at time $k-1$, sensor $s(k-1)$ updates and generates the estimate x_{k-1}. Then, agent $s(k-1)$ may either pass this estimate to a neighbor j, $j \in N(s(k-1))$, with probability

$$[P]_{s(k-1),s(k)} = \frac{1}{m},$$

or choose to keep the iterate with the remaining probability, in which case $s(k) = s(k-1)$. See Fig. 2 for an illustration. The update rule for this method is given by

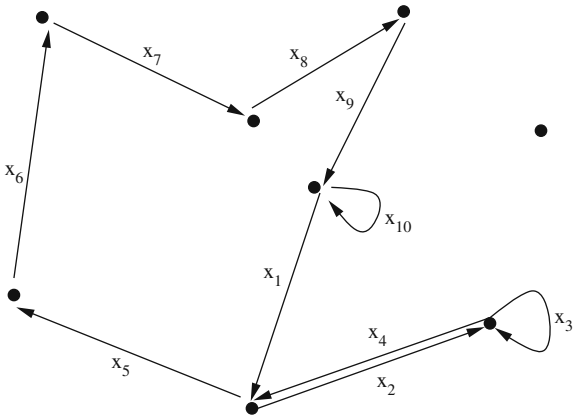


Fig. 2 A network of 8 sensors with incremental processing. The estimate is randomly passed through the network

$$x_k = P_X\left[x_{k-1} - 2\alpha_k\left(g_{s(k)}(x_{k-1}, u_{s(k),k}) - r_{s(k),k}\right)\nabla g_{s(k)}(x_{k-1}, u_{s(k),k})\right], \quad (5)$$

where $x_0 \in X$ is some random initial vector. Observe that the value of $s(k)$ depends only on the value of $s(k-1)$. Thus, the sequence of agents $\{s(k)\}$ can be viewed as a Markov chain with states $V = \{1, \ldots, m\}$ and transition matrix P. If the network topology changes with time, for example, in mobile sensor networks, then the transition matrix will also be time-varying [26]. Also note that the sensors need not sense in every slot, and a sensor needs to make a measurement only when it obtains the iterate. To save energy, sensors without an iterate may enter a sleep mode.

The Markov incremental algorithm has some advantages over the cyclic incremental algorithm. First, the algorithm does not require the setup phase that is required in the cyclic incremental algorithm. Second, the algorithm requires only connectivity of the network, which is weaker than the condition imposed on the network in the cyclic incremental algorithm. However, one can expect the Markov incremental algorithm to take more time to converge than the cyclic incremental algorithm. In one sensing interval only one update of the iterate can be performed as a sensor may choose to retain the iterate for the next iteration and will have to wait until the new measurement to update the iterate. In contrast, in the cyclic incremental algorithm in each sampling interval m updates of the iterate are performed.

In the above discussion, the performance of the algorithms as a function of time was discussed. A better comparison would be to keep the communication costs a constant, and compare one iteration of the cyclic incremental algorithm with m iterations of the Markov incremental algorithm. Even in this case one can expect the cyclic incremental algorithm to converge faster. Since the sequence in which the agents update the agents in the Markov incremental algorithm is random, the iterate may be caught in some "corner" of the network and the other agents may receive the iterate only after a large number of iterations.

3.1.3 Diffusive Nonlinear Recursive Algorithm

In this setting, each agent maintains and updates an iterate sequence. This is fundamentally different from the incremental algorithms in which a single iterate sequence is incrementally updated by the agents. We will use $w_{i,k-1}$ to denote the iterate with agent i at the end of time slot $k-1$. One iteration of the algorithm is performed in each sampling interval. Each agent receives the current iterate of its present neighbors. See Fig. 3 for an illustration. Each agent then calculates the following weighted sum $v_{i,k-1}$ given by

$$v_{i,k-1} = w_{i,k-1} + \frac{1}{m}\sum_{j\in N(i)}(w_{j,k-1} - w_{i,k-1}).$$

Sensor i then obtains its new iterate $w_{i,k+1}$ from $v_{i,k}$ according to

$$w_{i,k} = P_X\left[v_{i,k-1} - 2\alpha_k\left(g_i(v_{i,k-1}, u_{i,k}) - r_{i,k}\right)\left(\nabla g_i(v_{i,k}, u_{i,k})\right)\right]. \quad (6)$$

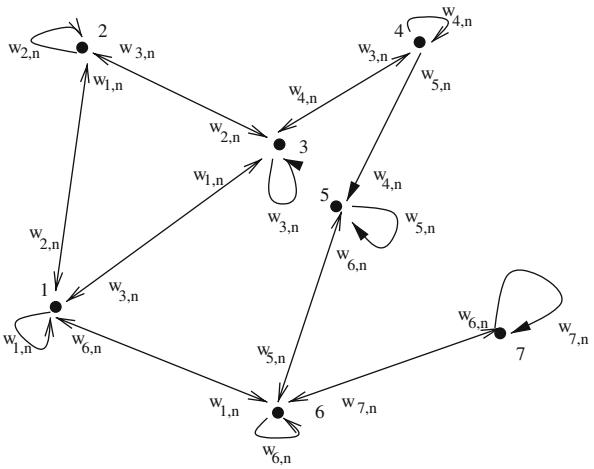


Fig. 3 A network of 8 sensors with parallel processing. Each agent shares its current iterate with its neighbors

The initial points $\{w_{i,0}\}$ are chosen randomly.

Similar to the Markov incremental algorithm, this algorithm requires the network to be only connected and does not require a setup phase. Similar to the cyclic incremental algorithm, each sensor uses its latest measurement to update in every iteration. However, note that each sensor updates a different iterate sequence. Therefore, unless the network is "densely connected" and each sensor has a large number of neighbors, one can expect the performance of the algorithm with time to be worse than that of the cyclic incremental algorithm. It is not immediately clear, however, if the performance of the distributed parallel algorithm as a function of communication costs would be better than the Markov incremental algorithm. The diffusive algorithms are closely related to consensus algorithms [9, 32], and they can be viewed as general consensus algorithms with stochastic errors.

3.2 Convergence of the Algorithms

The algorithms described above in Sect. 3.1 have been defined for arbitrary sequences $\{r_{i,k}\}$ and $\{u_{i,k}\}$. However, to study their convergence we would need to impose some restrictions on the actual measurement sequences.

Assumption 1 *The data $\{z_k\}$ is the sample path of an i.i.d. random process $\{Z_k\}$.*

In view of this assumption, it is possible to interpret the algorithms discussed above as stochastic approximation algorithms [3, 8, 22] and this will form the basis for the convergence results we state later. Due to Assumption 1,

$$x^* = \underset{X}{\operatorname{argmin}} \sum_{i=1}^{m} \mathsf{E}\Big[\big(R_{i,k} - g_i(x, U_{i,k})\big)^2\Big]. \tag{7}$$

Define $f_i(x) = \mathsf{E}\big[\big(R_{i,k} - g_i(x, U_{i,k})\big)\big]^2$ and $f(x) = \sum_{i=1}^{m} f_i(x)$. Observe that x^* is the minimum of $f(x)$. Suppose that the statistics of $R_{i,k}$ and $U_{i,k}$ are explicitly known to sensor i. Then, the problem simplifies to minimizing the sum of deterministic functions, where each function is known to a sensor. This distributed optimization problem has been studied in [18, 19, 25, 26] and different iterative distributed algorithms have been proposed. For example, consider the incremental gradient algorithm. In each iteration, the iterate is cycled through the network. Sensor i updates the iterate in the k-th iteration according to

$$\begin{aligned} z_{i,k} =& P_X\left[z_{i-1,k} - \alpha_k \nabla f_i(z_{i-1,k})\right] \\ =& P_X\left[z_{i-1,k} - 2\alpha_k \mathsf{E}\big[\big(g_i(z_{i-1,k}, U_{i,k}) - R_{i,k}\big)\big(\nabla g_i(z_{i-1,k}, U_{i,k})\big)\big]\right], \end{aligned}$$

and communicates the updated iterate to sensor $i + 1$. Note that algorithm requires sensor i to only know its local function f_i.

In the estimation problem, the statistics of $U_{i,k}$ and $R_{i,k}$ are not known. If they were known, the least squares estimate could have been obtained without making any measurements. Thus, sensor i cannot evaluate the gradient term $\nabla f_i(z_{i-1,k})$ that is required in the k-th iteration. Instead, if the k-th iteration is performed in the $(k + 1)$-th sampling interval, i.e., after the k-th observation has been made, sensor i can approximate $\nabla f_i(z_{i-1,k})$ with an LMS style instantaneous empirical approximation, i.e,

$$\mathsf{E}\big[\big(g_i(z_{i-1,k}, U_{i,k}) - R_{i,k}\big)\big(\nabla g_i(z_{i-1,k}, U_{i,k})\big)\big] \approx \big(g_i(z_{i-1,k}, u_{i,k}) - r_{i,k}\big)\big(\nabla g_i(z_{i-1,k}, u_{i,k})\big).$$

Observe that the incremental algorithm with this approximation is the cyclic incremental recursive estimation algorithm in (4). Thus, in effect the cyclic incremental recursive estimation algorithm is a stochastic optimization algorithm.

In a similar manner, the Markov incremental recursive algorithm and the diffusive recursive algorithm are obtained from their deterministic counterparts discussed in [25, 26]. We next formally state the conditions that are required for the convergence of the algorithm. We refer the reader to [25, 26] for the proofs.

Theorem 1 *Consider the model set in (3). Let the set X be closed and convex, and let Assumption 1 hold. Further, let the functions $f_i(x)$ be convex and have bounded (sub)gradients on the set X. If $\{\alpha_k\}$ is chosen such that $\sum_k \alpha_k^2 < \infty$ and $\sum_k \alpha_k = \infty$. Then the following convergence results hold:*

1. *If the network topology allow a cycle that connects all the sensors, then iterates generated in (4), converge to a minimum of $f(x)$ with probability 1 and in mean square.*
2. *If the network is connected, then the iterates generated in (5) converge to a minimum of $f(x)$ in mean square.*
3. *If the network is connected, then the iterates generated in (6) converge to the same minimum of $f(x)$, for all $i \in V$, with probability 1 and in mean square.*

The convergence results in [25, 26] are more general and deal with general choices of probabilities in (5) and general weights in (6). While we have no proof, we believe that the convergence of the algorithms can be extended to the general case when $\{Z_k\}$ is required to satisfy only the conditions of exponential stability and stationarity (explained below). This has been verified for the specific case where $g_i(x, \cdot)$ is linear in x.

Theorem 1 requires the functions $f_i(x)$ to be convex. A simple case when this holds is when $g_i(x, \cdot)$ is linear in x. However, in general the condition is quite implicit and there seems to be no direct way to verify it. We next state another convergence result, which is applicable for a different class of functions. We only state the result and refer the reader to [24] for the essential ideas of the proofs.

Theorem 2 *Consider the model set (3) and let the network be connected. Additionally, if the incremental algorithm is used, let the network topology have a cycle. Let Assumption 1 hold, and let the set X be $\Re^n$, or compact and convex. Further, let the functions $f_i(x)$ be differentiable on the set X and let the gradient be Lipschitz continuous. If the set $X = \Re^n$, also assume that the gradients are bounded. If $\{\alpha_k\}$ is chosen such that $\sum_k \alpha_k^2 < \infty$ and $\sum_k \alpha_k = \infty$, then every cluster point of the iterate sequences generated in (4), (5) and (6) will be a stationary point.*

First note that the theorem guarantees only convergence to a stationary point. In addition, the conditions imposed on the set X are also stronger than the conditions in Theorem 1. However, note that the conditions imposed on the functions f_i are weak, and easier to verify. For example, a sufficient condition for convergence when X is compact is for the function $g_i(x, \cdot)$ to be twice continuously differentiable.

3.3 Effect of Quantization

Typically, the iterates are first quantized before they are communicated in wireless networks. We next study the effect of quantization on the estimation algorithms. The quantization lattice is the set

$$Q = \{(q_1\Delta, \ldots, q_n\Delta); q_j \in \mathbb{Z}\}.$$

For a vector $x \in \Re^n$, we use $Q[x]$ to denote the quantized value of x. We consider the following two types of quantizers:

- *Basic quantizer*. The quantized value of a vector $x \in X$, is the Euclidean projection of x. Thus, $Q[x] = P_Q[x]$. Observe that $\|Q[x] - x\|$ is deterministically bounded by $\frac{\sqrt{n}\Delta}{2}$ [23].
- *Dither quantizer*. The quantized value of a vector $x \in X$, is the Euclidean projection of x added with a dither signal. Thus, $Q[x] = P_Q[x + D]$, where D is a dither signal whose component are uniformly and independently chosen in $[\frac{-\Delta}{2}, \frac{\Delta}{2}]$. In this case, $\|Q[x] - x\|$, is random, statistically independent of x and uniformly distributed in $\left[\frac{-\sqrt{n}\Delta}{2}, \frac{\sqrt{n}\Delta}{2}\right]$[2, 10].

When there is quantization, it is better to use a constant stepsize than a diminishing stepsize. We motivate this through the following discussion. It seems reasonable to require the algorithms, with quantization, to converge to $Q[x^*,]$ i.e., the lattice point that is the closest to x^*. However, this is not the case even when there are no stochastic errors. To see this consider the standard gradient descent algorithm to minimize $f(x)$ over the set $X = \Re$ with a basic quantizer without any dither. The iterates are generated according,

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k).$$

Suppose $x_0 \neq Q[x^*]$, and $\alpha_k < \frac{\Delta}{2C}$, where C is the bound on the subgradient of f_i, then it can be immediately seen that $x_k = x_0$, for all k. More generally, one can conclude that stepsizes should always be large enough to push the iterate from a non-optimal lattice point, but small enough to ensure that the iterate gets caught in the optimal lattice point.

We will only discuss the quantized cyclic incremental algorithm here. Similar results can be easily obtained for quantized versions of the other two algorithms. Formally, the quantized cyclical incremental recursive estimation algorithm is given by:

$$\begin{aligned} z_{0,k} &= z_{m,k-1} = x_{k-1}, \\ z_{i,k} &= Q\left[P_X\left[z_{i-1,k} - 2\alpha_k \left(g_i(z_{i-1,k}, u_{i,k}) - r_{i,k}\right)\left(\nabla g_i(z_{i-1,k}, u_{i,k})\right)\right]\right]. \end{aligned} \quad (8)$$

It is possible to obtain bounds on that the asymptotic performance of the algorithm by extending the analysis for constant stepsize in [26] to handle quantization by using ideas similar to those in [23]. Define ν_i to be some constant such that

$$\nu_i \geq \sup_{x \in X} \mathsf{Var}\left[\left(R_i - g_i\left(x.U_i^T\right) -\right)\left(\nabla g_i\left(x, U_i\right)\right)\right].$$

Then ν_i is an upper bound on the variance of the empirical gradient estimate.

Theorem 3 *Let the Assumptions of Theorem 1 hold. Additionally, assume that X is bounded and a constant stepsize α is used. Then, with basic quantization the iterates* $\{x_k\}$ *in (8) satisfy*

$$\liminf_{k\to 0} \mathsf{E}[f(x_k)] \leq f^* + \frac{\sqrt{n}\Delta \max_{x,y\in X} \|x - y\|}{2\alpha} + \frac{\alpha}{2}\left(\sum_{i=1}^{m}\left(C_i + \nu_i + \frac{\sqrt{n}\Delta}{2\alpha}\right)\right)^2,$$

and with dither quantization the iterates satisfy

$$\liminf_{k\to 0} \mathsf{E}[f(x_k)] \leq f^* + \frac{\sqrt{n}\Delta \max_{x,y\in X} \|x - y\|}{2\alpha} + \frac{\alpha}{2}\left(\sum_{i=1}^{m}\left(C_i + \nu_i + \frac{\sqrt{n}\Delta}{\alpha\sqrt{6}}\right)\right)^2.$$

Furthermore, the bounds hold with probability 1 for $\inf_{k\geq 0} f(x_k)$.

Observe that with dither quantization it is possible to obtain a better bound. This indicates that dither quantization may result in smaller errors than basic quantization.

3.4 Special Case: Linear Regression

We next consider the special case when $g_i(x, U_{i,k}) = x^T U_{i,k}$, and $X = \Re^n$. In a centralized setting, it is possible to recursively evaluate the exact least squares, i.e, evaluate $\hat{x}_N$, from $\hat{x}_{N-1}$ and new measurements [15]. From a stochastic optimization perspective, the recursive least squares (RLS) algorithm approximates the standard Newton's gradient algorithm, where the gradient term is approximated using an instantaneous empirical approximation and a sample average for the Hessian term [15]. Attempts have been made to exploit this to develop distributed and recursive algorithms that perform better than the LMS-type algorithms discussed above. We discuss these next.

Incremental RLS algorithms can be obtained by scaling the gradient term in the incremental LMS with a matrix $L_{i,k}$. Formally,

$$z_{i,k} = z_{i-1,k} - \alpha_k L_{i,k} u_{i,k} (u_{i,k}^T z_{i-1,k} - r_{i,k}).$$

The matrix $L_{i,k}$ is to be viewed as an approximation to the Hessian in the Newton descent algorithm, which is determined recursively. If we allow the sensors to share a little more information to evaluate $L_{i,k}$, then it is possible to implement the least squares algorithm exactly, in a recursive and incremental manner [16]. Specifically, the matrix $L_{i,k}$ must have the form

$$\begin{aligned} L_{i,k} &= \frac{P_{i-1,k}}{1 + u_{i,k}^T P_{i-1,k} u_{i,k}} \\ P_{i,k} &= P_{i-1,k} - \alpha_k \frac{P_{i-1,k} u_{i,k} u_{i,k}^T P_{i-1,k}}{1 + u_{i,k}^T P_{i-1,k} u_{i,k}}. \end{aligned} \tag{9}$$

Note that the algorithm requires sensor $i-1$ to communicate to sensor i the statistic $P_{i-1,k}$, in addition to the iterate sequence. A more communication efficient incremental estimation algorithm can be obtained by replacing the $P_{i,k}$ in (9) with an approximation, $Q_{i,k}$, that is evaluated locally in the following manner

$$Q_{i,k} = Q_{i,k-1} - \frac{Q_{i,k-1} u_{i,k} u_{i,k}^T Q_{i,k-1}}{1 + u_{i,k}^T Q_{i,k-1} u_{i,k}}.$$

Note that the resulting algorithm requires the sensors to only cycle the iterates. The matrix $Q_{i,k}$ is recursively evaluated at sensor i using only local information. When α_k is fixed at 1, the estimates do not asymptotically converge to the least-squares estimate, but only to its neighborhood [30]. While it has not been verified, when

$\{\alpha_k\}$ diminishes at a suitable rate, one can expect the algorithm to converge to the least squares estimate.

Similarly, diffusive RLS algorithms can be obtained by scaling the gradient term in (6) with a matrix $L_{i,k}$ [5, 7, 33]. As stated previously, the matrix $L_{i,k}$ is the "equivalent" of the Hessian in the Newton direction, and can be evaluated in a distributed and recursive manner in different ways [7, 33]. A diffusive LMS algorithm that is quite different from the diffusive algorithms discussed so far is proposed in [31]. This algorithm uses a hierarchical network structure, in which a subset of sensors are designated as "bridge sensors" and a sensor is required to communicate only with neighbors, that are bridges. However, the algorithm converges only to a region around the optimal estimate.

3.5 Special Case: Accurate Model Sets

Next consider the case when the model set is accurate, i.e., there exists a $\bar{x}$ such that

$$R_{i,k} = g_i(\bar{x}, U_{i,k}) + E_{i,k}.$$

In this case it can be immediately seen that x^*, which is the asymptotic limit of the least squares estimators, will equal $\bar{x}$. For this case, the problem of estimating the parameter can be viewed as solving a system of equations, rather than an optimization problem. Specifically, observe that x^* is a solution to the following problem:

$$\text{Determine } x \text{ such that} \qquad \sum_{i=1}^{m} \mathsf{E}\big[R_{i,k} - g_i(x, U_{i,k}) - h_i(x)\big] = 0$$

The statistics of $R_{i,k}$ and $U_{i,k}$ are not known but we observe the samples $\{r_{i,k}\}$ and $\{u_{i,k}\}$. In a centralized setting, Robbins-Monro algorithm [28] can be used to solve the problem, by generating iterates according to:

$$x_k = x_{k-1} - \alpha_k \sum_{i=1}^{m} (r_{i,k} - g_i(x_{i,k}, u_{i,k})).$$

Distributed versions of these algorithm can developed. In the diffusive algorithms, proposed in [10], the iterates are generated according to

$$w_{i,k+1} = v_{i,k} - \alpha_k (v_{i,k} - g_i(v_{i,k}, u_{i,k})).$$

Similar cyclic and Markov incremental fixed point algorithms can also be developed. While this approach requires the stronger assumption that the model set be accurate, it is has the advantage that convergence can be obtained by imposing conditions directly on g_i and h_i [10].

4 Gaussian Linear State Space Model Sets

We next focus on the following class of model sets

$$\begin{aligned}\Theta_{i,k+1}(x) &= F_i(x)\Theta_{i,k}(x) + U_{i,k+1}^T G_i(x) + V_{i,k+1}(x)\\ \hat{R}_{i,k+1}(x, U^k) &= H_i(x)\Theta_{i,k+1}(x) + W_{i,k+1}(x),\end{aligned} \tag{10}$$

where $\{W_{i,k+1}(x)\}$ and $\{V_{i,k+1}(x)\}$ are i.i.d. random processes. The state vector $\Theta_{i,k+1}(x)$ is a vector of dimension q and the distribution of $\Theta_{i,0}(x)$ is such that the process $\{\hat{R}_{i,k}(x)\}_{k\in\mathbb{N}}$ is stationary. We will also assume that the random processes $\{W_{i,k}(x)\}_{k\in\mathbb{N}}$ and $\{V_{i,k}(x)\}_{k\in\mathbb{N}}$ are zero mean i.i.d. random sequences.[4]

Note that the (10) is the most general linear system description [15]. Linear state space models arise naturally, or as approximations to non-linear systems. For example, the auto-regressive moving average (ARMA) model set is an important special case, where

$$\hat{R}_{i,k+1}(x, R_i^k) = R_{i,k-1}g_i(x) + E_{i,k+1}(x).$$

Here $E_{i,k+1}(x) = E_{i,k}(x) + N_{i,k}(x)$, with $\{N_{i,k}(x)\}$ being an i.i.d. sequence. We refer the reader to [15] for further discussion.

For the model set in (10), the optimal predictor $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ will be the Kalman predictor. Let $K_i(x)$ be the Kalman gain for the state-space system in (10), which is determined from $D_i(x)$, $H_i(x)$, $\mathsf{Cov}\big(W_{i,k}(x)\big)$, and $\mathsf{Cov}\big(V_{i,k}(x)\big)$ as the solution to the Riccati equation [15]. Define $F_i(x) = D_i(x) - K_i(x)H_i(x)$. The Kalman predictor, $p_{i,k+1}(x, u_i^{k+1}r_i^k)$, is

$$\begin{aligned}\phi_{i,k+1}(x, u_i^{k+1}, r_i^k) =& F_i(x)\phi_{i,k}(x, u_i^k, r_i^{k-1}) + K_i(x)r_{i,k} + u_{i,k+1}^T G_i(x),\\ p_{i,k+1}(x, u_i^{k+1}, r_i^k) =& H_i(x)\phi_{i,k+1}(x, u_i^{k+1}, r_i^k).\end{aligned} \tag{11}$$

Note that to evaluate the gradient of $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ at x, u_i^{k+1} and r_i^k are required. In contrast, in the simple non-linear regression model, $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ was a function of only $u_{i,k+1}$ and we could make the algorithm recursive by using an LMS approximation. Thus, we need to make two approximations (a) an LMS-like approximation for the gradient, and (b) an approximation to make the LMS approximations recursive. These ideas are based on the recursive prediction error (RPE) algorithm of [15] and hence we call the incremental algorithm proposed below the incremental RPE (IRPE) algorithm [27]. Formally, the iterates are generated according to the following relations for $i \in V$ and $\ell = 1, \ldots, d$,

[4] We make the zero mean assumption to keep the presentation simple. The algorithm can be extended to the case when they are not zero mean.

$$
\begin{aligned}
x_{k-1} &= z_{m,k-1} = z_{0,k}, \qquad (12)\\
\begin{bmatrix} \psi_{i,k} \\ \chi_{i,k}^{(\ell)} \end{bmatrix} &= \begin{bmatrix} F_i(z_{i,k-1}) & 0 \\ \nabla^{(\ell)} F_i(z_{i,k-1}) & F_i(z_{i,k-1}) \end{bmatrix} \begin{bmatrix} \psi_{i,k-1} \\ \chi_{i,k-1}^{(\ell)} \end{bmatrix} + \begin{bmatrix} K_i(z_{i,k-1}) \\ \nabla^{(\ell)} K_i(z_{i,k-1}) \end{bmatrix} r_{i,k-1} \\
&\quad + \begin{bmatrix} G_i(z_{i,k-1}) \\ \nabla^{(\ell)} G_i(z_{i,k-1}) \end{bmatrix} u_{i,k} \\
\begin{bmatrix} \zeta_{i,k} \\ \xi_{i,k}^{(\ell)} \end{bmatrix} &= \begin{bmatrix} H_i(z_{i-1,k}) & 0 \\ \nabla H_i^{(l)}(z_{i-1,k}) & H_i(z_{i-1,k}) \end{bmatrix} \begin{bmatrix} \psi_{i,k} \\ \chi_{i,k}^{(\ell)} \end{bmatrix}, \\
\varepsilon_{i,k} &= r_{i,k} - \zeta_{i,k}, \\
z_{i,k}^{(\ell)} &= z_{i-1,k}^{(\ell)} - \alpha_k \left(\xi_{i,k}^{(\ell)}\right)^T \varepsilon_{i,k}, \\
z_{i,k} &= \begin{bmatrix} z_{i,k}^{(1)} & \dots & z_{i,k}^{(d)} \end{bmatrix}^T.
\end{aligned}
$$

The initial values for the recursion are fixed at $x_0 = x_s$, $\psi_{i,0} = \psi_{i,s}$ and $\chi_{i,0}^{(\ell)} = \chi_{i,s}^{(\ell)}$. Observe that the algorithm has a distributed and recursive implementation. In iteration k, sensor i first uses its iterates from the previous iteration, i.e., $z_{i,k-1}$, and $u_{i,k}$ to evaluate $\chi_{i,k}^{(1)}, \dots \chi_{i,k}^{(d)}$ and $\psi_{i,k}$. Sensor i then receives $z_{i-1,k}$ from sensor $i-1$ and updates it using $z_{i,k-1}$ to generate $z_{i,k}$. This is then passed to the next sensor in the cycle. Thus, sensor i only needs to store $z_{i,k}$, $\chi_{i,k}^{(1)}, \dots \chi_{i,k}^{(d)}$ and $\psi_{i,k}$ for the next iteration. The algorithm is therefore recursive and distributed. Furthermore, note that sensor i only needs to know its own system matrices $H_i(x)$, $F_i(x)$ and $G_i(x)$.

4.1 Convergence of the Algorithm

The data is assumed to satisfy the following assumption

Assumption 2 *For each $i \in V$, the sequence $\{z_k\}_{k\in\mathbb{N}}$ is the sample path of stationary and exponentially stable random process $\{Z_k\}_{k\in\mathbb{N}}$.*

Exponential stability requires z_k and z_s to be weakly related when $t \gg s$. Both these conditions are quite weak and are satisfied in practice. The precise mathematical definitions and other details are available on page 170 of [15]. We next state the convergence result. The proof is available in [27].

Theorem 4 *Let Assumption 2 hold and let the network topology allow a cycle between the sensors. For each $x \in X$, let the system in (10) be stable and let the matrix functions $F_i(x)$, $H_i(x)$ and $K_i(x)$ be twice continuously differentiable. Let the step-size α_k be such that $k\alpha_k$ converges to a positive constant. Then, the iterates $\{x_k\}$ generated by the IRPE algorithm in (12) converge to a local minimum of $f(x)$ in (2) with probability 1.*

5 Application: Determining the Source of a Diffusion Field

A diffusion field is a spatio-temporal field that follows the diffusion partial differential equation. The diffusion equation models diverse physical phenomena like the conduction of heat, dispersion of plumes in air, dispersion of odors through a medium and the migration of living organisms. We study the problem of determining the source of a diffusion field (specifically, temperature field) generated in a room.

We briefly review the literature on source identification in the diffusion equation. The point source model is a common model for diffusing sources and has been extensively used [1, 13, 17, 34]. Usually the intensity is assumed to be a constant [1, 13, 17] or instantaneous. Localization of sources with time-varying intensity have been studied in a centralized and non-recursive setting in [6, 20]. These studies consider a deterministic evolution of the leak intensity and use a continuous observation model. Most papers study that case when the medium is infinite or semi-infinite since the diffusion equation has a closed form solution in that case [13, 17]. An exception is [11] where two dimensional shaped medium is considered.

While centralized recursive source localization has received much interest [13, 17, 20, 21] there are very few papers that discuss a distributed solution. A recursive and distributed solution to the problem in a Bayesian setting is discussed in [34]. A related paper is [29] that deals with the problem of estimating the diffusion coefficient in a distributed and recursive manner. We are not aware of any prior work that solves the source localization problem using a distributed and recursive approach in a non-Bayesian setting. We refer the reader to [34] for a more detailed discussion of the literature.

We will consider two different parametrized model sets for the source (the feature of interest) and use the algorithms developed so far to determine the parameters.

5.1 Point Source and Constant Intensity Model Sets

We first consider the case when based on a priori information the model set for the source can be chosen to be the set of all point sources with constant intensity. The model set is parametrized by the source location $x = (x_1, x_2)$ and intensity I. Thus the problem of learning the source simplifies to estimating x and I. Additionally, it is also known that the warehouse is large, the initial temperature is a constant throughout the warehouse, and the thermal conductivity is large and uniform throughout the medium, and known.

To map the model set for the source to a model set for the sensor measurements we will use the diffusion equation. Let $C(s, t; x, I)$ denote the temperature at a point s at time t when the source is at x and intensity is I. For the source model set and medium, $C(s, t; x, I)$ can be approximated using the steady-state value given by [17]:

$$C(s; x, I) = \frac{I}{2\nu\pi \|s - x\|},$$

where ν is the diffusion coefficient. If s_i is the location of the i-th sensor, then the model set for its k-th measurement is

$$\hat{R}_{i,k}(x, I) = C(s_i; x, I) + N_{i,k},$$

where $N_{i,k}$ is a zero mean i.i.d. measurement noise. Thus the estimation problem is a simple nonlinear regression problem with no measurable inputs and can be solved using the algorithms developed in Sect. 3.

5.1.1 Numerical Results

We illustrate the performance of the algorithms with simulation results. In the simulation experiments the diffusion coefficient $\nu = 1$. The actual location of the source is $x^* = (37, 48)$ and the actual intensity value is 10. A network of 27 sensors is randomly deployed. The initial iterate value is fixed at (50, 50) for the source location and 5 for the intensity. The results are plotted in Figs. 4 and 5. Observe that about 200 iterations are sufficient to obtain a good estimate of the source location and 1000 iterations to obtain a good estimate of the intensity using the cyclic incremental, Markov incremental and diffusion algorithms. In addition, we observed that the convergence speed of the algorithm is affected by the initial point. If the initial points are taken very far from the actual value then there is convergence to other stationary points in the problem.

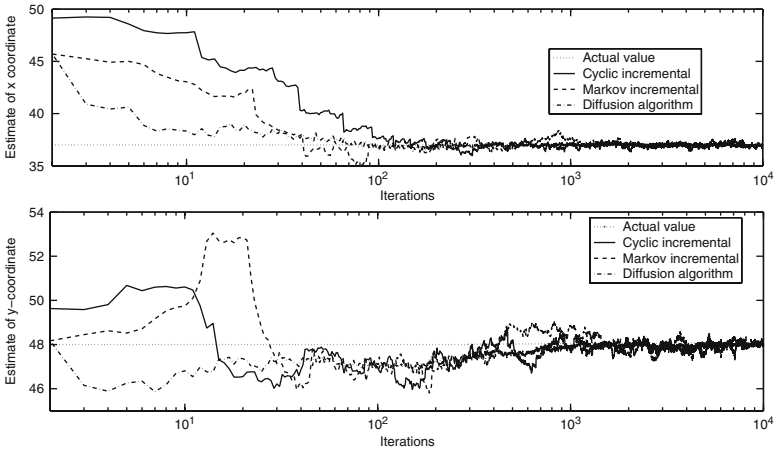


Fig. 4 Estimates of the x and y coordinates generated by the cyclic incremental, Markov incremental and diffusion gradient algorithms

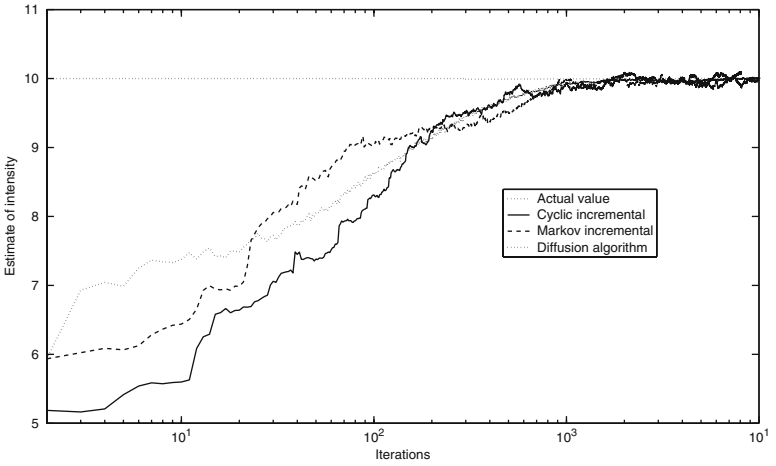


Fig. 5 Estimate of the intensity generated by the cyclic incremental, Markov incremental and diffusion gradient algorithms

5.2 Point Source and Time-Varying Intensity Model Sets

Suppose that based on a priori information, the model set that is chosen consists of all point sources that switch on at time $t = 0$ with intensity values $I(t) = I_k$, for $t \in [k-1, k)$, where $I_{k+1} = \rho I_k + S_k$. Here, ρ is a known scalar and $\{S_k\}$ is a sequence of i.i.d. Gaussian random variables with mean $(1-\rho)\mu$ and variance $(1-\rho^2)\sigma_s^2$, and $I(0)$ is Gaussian with mean μ and variance σ_s^2. Thus the model set for the source, which is the feature of interest, is parametrized by the location $x = (x_1, x_2)$. Note that $\{I_k\}$ does not parametrize the model set and is an incidental random process whose statistics are specified. Thus we only need to estimate the source location.

Additionally the following is known about the medium. Let D denote the room and ∂D denote the boundary of the room. The medium is uniform throughout the room. The temperature at the boundaries of the room is always a constant and at time $t = 0$, the temperature is modeled to be a constant everywhere in the room. Without loss of generality, we fix the constant temperature to be 0, i.e., $C(\cdot, 0; \cdot) = 0$.

For the above source model set and medium, the temperature at a point y at time t is

$$\frac{\partial C(y,t;x)}{\partial t} = \nu \nabla^2 C(y,t;x) + I(t)\bar{\delta}(y-x), \tag{13}$$

with the initial and boundary conditions

$$C(s,0;x) = 0 \text{ for all } s \in D,$$
$$C(s,t;x) = 0 \text{ for all } t \geq 0 \text{ and } s \in \partial D.$$

Here, ν is the medium conductivity, ∇^2 is the Laplacian differential operator and $\bar{\delta}$ is the Dirac delta function. Let s_i be the location of the i-th sensor. Then the model set for the sensor measurements is

$$\hat{R}_{i,k}(x) = C(s_i, k; x) + N_{i,k},$$

where $N_{i,k}$ is a zero mean i.i.d. Gaussian measurement noise with known variance σ_n^2. By using Green's technique to solve the differential equations, it is possible to obtain an approximate state-space description for each sensor's observation process. The estimation problem can now be solved using the techniques developed in Sect. 4. We refer the reader to [27] for a detailed analysis.

5.2.1 Numerical Results

We illustrate the performance of the algorithm with simulation results for the problem discussed above. In the simulation experiments, $l_1 = l_2 = 100$ and diffusion coefficient $\nu = 1$. The actual location of the source is $x^* = (37, 48)$. The value of $\mu = 1000$, $\rho = 0.95$ and $\sigma_s^2 = 1$. A network of 27 sensors is deployed. To ensure complete coverage of the sensing area, we first placed 9 sensors on a grid and then randomly deployed 3 sensors in the immediate neighborhood of each of the 9 sensors. The network is shown in Fig. 6. The sampling interval is 10 time units and the measurement noise variance is set to 0.1.

Observe from Fig. 7 that the IRPE algorithm does not converge to the correct parameter value. This is because, in the absence of convexity, the algorithms in general are only guaranteed to converge to a stationary point and necessarily the global minimum of the cost function.

One way to improve the structure of the cost function is to partially model the dependence between sensor measurements. The idea is to group sensors in clusters. The dependence between the sensor measurements within a cluster is explicitly modeled, while across clusters no model is assumed. Each sensor could then pass its measurements to a cluster head and the distributed algorithms discussed could be run between the cluster heads. Note that in this case the $p_{i,k+1}$ term in the conditional least squares term in (1) should be the best estimate for $R_{i,k+1}$ based on the past and present measurements of the independent variable, and all past values of the dependent variable, of *all the sensors in the cluster*. Therefore, to improve the topology of the cost function the network is divided into 9 clusters of size 3. The cluster heads are the sensors marked by circles in Fig. 6. At the beginning of each slot, a cluster head collects the measurements from the sensors in the cluster. A total of 1000 iterations of the IRPE is run between the cluster heads and the performance is compared with 1000 iterations of the RPE. Observe from Fig. 7 that the algorithm now converges to the correct parameter value.

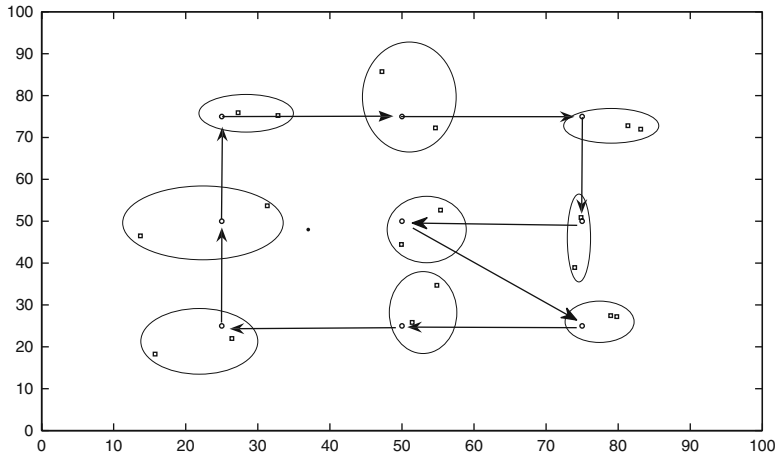


Fig. 6 A network of 27 sensors. The *small circles* denote the cluster heads and the *squares* denote the sensors. The source is represented by a *dot*. The *arrows* indicate the order in which the iterates are passed in the cluster IRPE

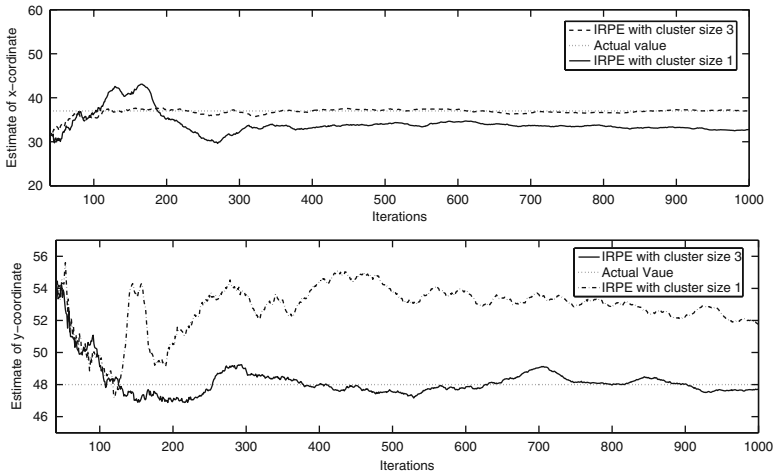


Fig. 7 Estimates of the x and y coordinates generated by the standard IRPE and the cluster IRPE. Observe that the standard IRPE converges to a stationary point while the cluster IRPE converges to the correct source location

6 Discussion

In this chapter we have focused on least squares parameter estimation. All of the estimation algorithms were based on stochastic gradient optimization algorithms. Similar ideas can be used to obtain weighted, and robust least-squares estimators, for the case where the criterion is not necessarily quadratic. If the sensor measurements are independent across sensors, the ideas presented here can be extended to

obtain distributed and recursive maximum likelihood estimators; the independence across sensors provides the additive form for the estimation cost function, which makes it possible to use the distributed stochastic algorithms. Related work, which we did not discuss in this chapter, is that involving recursive Bayesian estimation over networks. In [4], the asymptotic convergence of optimal Bayesian estimates is investigated, under the premise that the network is not very complicated and each sensor can keep track of the total information flow. In [34], cyclical incremental Bayesian estimation algorithms are discussed.

There are a number of avenues for future extensions. Perhaps, the most immediate is the development and analysis of Markov incremental and diffusive algorithms for Gaussian state space model set. Another avenue involves relaxing the assumption that the sensors sense and process in a synchronous manner, since this assumption may not be valid in large networks. At a broader level, it is of interest to develop recursive and distributed estimation algorithms for estimation in other model sets that the two specific model sets that we described in the chapter.

Acknowledgement This work was supported in part by a Vodafone Fellowship, Motorola, and the National Science Foundation, under CAREER grant CMMI 07-42538.

References

1. Alpay, M.E., Shor, M.H.: Model-based solution techniques for the source localization problem. IEEE Transactions on Control Systems Technology **8**(6), (2000)
2. Aysal, T., Coates, M., Rabbat, M.: Distributed average consensus with dithered quantization. IEEE Transactions on Signal Processing **56**(10), 4905–4918 (2008)
3. Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and Distributed Computation: Numerical Methods. Athena Scientific, Belmont, MA (1997)
4. Borkar, V., Varaiya, P.P.: Asymptotic agreement in distributed estimation. IEEE Transactions on Automatic Control **27**, 650–655 (1982)
5. Calafiore, G., Abrate, F.: Distributed linear estimation over sensor networks. International Journal of Control **82**(5), 868–882 (2009)
6. Cannon, J.R., DuChateau, P.: Structural identification of an unknown source term in a heat equation. Inverse Problems **14**, 535–551 (1998)
7. Cattivelli, F.: Diffusion recursive least-squares for distributed estimation over adaptive networks. IEEE Transactions on Signal Processing **56**(5), 1865–1877 (2008)
8. Ermoliev, Y.: Stochastic quasi-gradient methods and their application to system optimization. Stochastics **9**(1), 1–36 (1983)
9. Jadbabaie, A., Lin, J., Morse, S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Transactions on Automatic Control **48**(6), 998–1001 (2003)
10. Kar, S., Moura, J., Ramanan, K.: Distributed parameter estimation in sensor networks: nonlinear observation models and imperfect communication. Submitted, available at http://arxiv.org/abs/0809.0009 (2008)
11. Khachfe, A.R., Jarny, Y.: Estimation of heat sources within two dimensional shaped bodies. In: Proceedings of 3rd International Conference on Inverse Problems in Engineering (1999)
12. Klimko, L., Nelson, P.: On conditional least squares estimation for stochastic processes. The Annals of Statistics **6**(3), 629–642 (1978)
13. Levinbook, Y., Wong, T.F.: Maximum likelihood diffusive source localization based on binary observations. In: Conference Record of the Thirty-Eighth Asilomar Conference on Signal, Systems and Computers (2004)

14. Levy, E., Louchard, G., Petit, J.: A distributed algorithm to find Hamiltonian cycles in G(n,p) random graphs. Lecture Notes in Computer Science **3405**, 63–74 (2005)
15. Ljung, L., Söderström, T.: Theory and Practice of Recursive Identification. The MIT Press, Cambridge, MA (1983)
16. Lopes, C.G., Sayeed, A.H.: Incremental adaptive strategies over distributed networks. IEEE Transactions on Signal Processing **55**(8), 4064–4077 (2007)
17. Matthes, J., Gröll, L., Keller, H.B.: Source localization for spatially distributed electronic noses for advection and diffusion. IEEE Transactions on Signal Processing **53**(5), 1711–1719 (2005)
18. Nedić, A., Bertsekas, D.P.: Incremental subgradient method for nondifferentiable optimization. SIAM Journal of Optimization **12**, 109–138 (2001)
19. Nedić, A., Ozdaglar, A.: Distributed sub-gradient methods for multi-agent optimization. Transactions on Automatic Control **54**(1), 48–61 (2009)
20. Niliot, C.L., Lefévre, F.: Multiple transient point heat sources identification in heat diffusion: application to numerical two- and three-dimensional problems. Numerical Heat Transfer **39**, 277–301 (2001)
21. Piterbarg, L.I., Rozovskii, B.L.: On asymptotic problems of parameter estimation in stochastic PDE's: the case of discrete time sampling. Mathematical Methods of Statistics **6**, 200–223 (1997)
22. Polyak, B.T.: Introduction to Optimization. Optimization Software Inc., New York (1987)
23. Rabbat, M.G., Nowak, R.D.: Quantized incremental algorithms for distributed optimization. IEEE Journal on Select Areas in Communications **23**(4), 798–808 (2005)
24. Ram, S.S., Nedić, A., Veeravalli, V.V.: Asynchronous gossip algorithms for stochastic optimization. Submitted to IEEE CDC (2000)
25. Ram, S.S., Nedić, A., Veeravalli, V.V.: Distributed stochastic subgradient algorithm for convex optimization. Submitted to SIAM Journal on Optimization. Available at http://arxiv.org/abs/0811.2595 (2008)
26. Ram, S.S., Nedić, A., Veeravalli, V.V.: Incremental stochastic sub-gradient algorithms for convex optimization. SIAM Journal on Optimization **20**(2), 691–717 (2009)
27. Ram, S.S., Veeravalli, V.V., Nedić, A.: Distributed and recursive parameter estimation in parametrized linear state-space models. To appear in IEEE Transactions on Automatic Control
28. Robbins, H., Monro, S.: A stochastic approximation method. Annals of Mathematical Statistics **22**(3), 400–407 (1951)
29. Rossi, L., Krishnamachari, B., Kuo, C.C.J.: Distributed parameter estimation for monitoring diffusion phenomena using physical models. In: First IEEE International Conference on Sensor and Ad Hoc Communications and Networks (2004)
30. Sayed, A., Lopes, C.: Distributed recursive least-squares strategies over adaptive networks. In: 40th Asilomar Conference on Signals, Systems and Computers, pp. 233–237 (2006)
31. Schizas, I., Mateos, G., Giannakis, G.: Stability analysis of the consensus-based distributed LMS algorithm. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3289–3292 (2008)
32. Tsitsiklis, J.N.: Problems in decentralized decision making and computation. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA (1984)
33. Xiao, L., Boyd, S., Lall, S.: A space-time diffusion scheme for peer-to-peer least-square estimation. In: Fifth International Conference on Information Processing in Sensor Networks, pp. 168–176 (2006)
34. Zhao, T., Nehorai, A.: Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks. IEEE Transactions on Signal Processing **55**, 4669–4682 (2007)

Self-Organization of Sensor Networks with Heterogeneous Connectivity

Arun Prasath, Abhinay Venuturumilli, Aravind Ranganathan, and Ali A. Minai

Abstract Most research on wireless sensor networks has focused on homogeneous networks where all nodes have identical transmission ranges. However, heterogeneous networks, where nodes have different transmission ranges, are potentially much more efficient. In this chapter, we study how heterogeneous networks can be configured by distributed self-organization algorithms where each node selects its own transmission range based on local information. We define a specific performance function, and show empirically that self-organization based on local information produces networks that are close to optimal, and that including more information provides only marginal benefit. We also investigate whether the quality of networks configured by self-organization results from their generic connectivity distribution (as is argued for scale-free networks) or from their specific pattern of heterogeneous connectivity, finding the latter to be the case. The study confirms that heterogeneous networks outperform homogeneous ones, though with randomly deployed nodes, networks that seek homogeneous out-degree have an advantage over networks that simply use the same transmission range for all nodes. Finally, our simulation results show that highly optimized network configurations are as robust as non-optimized ones with respect to random node failure, but are much more susceptible to targeted attacks that preferentially remove nodes with the highest connectivity, confirming the trade-off between optimality and robustness postulated for optimized complex systems.

1 Introduction

Self-organization is seen increasingly as an attractive alternative to design for engineering large-scale complex systems such as sensor networks, robot swarms, multi-agent systems, self-reconfiguring robots and smart structures [14]. As systems grow in size, self-organization provides an inherently scalable, flexible and robust way to obtain effective functionality without the need for global communication or control. However, most theoretical work on self-organization has focused on physical

A. Prasath (✉)
ECE Department, University of Cincinnati, Cincinnati, OH, USA
e-mail: news4arun@gmail.com

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_3,

systems or their simplified models (e.g., cellular automata, percolation models, sandpiles, etc.), where the process is driven by simple rules that have little to do with domain-dependent performance considerations. In contrast, systems for engineering applications must accomplish goal-directed tasks, and their self-organization rules must be based on domain-specific considerations such as bandwidth, capacity, cost, energy resources, etc. Ultimately, the success (or failure) of such self-organization procedures must be judged by whether the resulting system has close to optimal *performance* – an issue seldom considered for abstract models such as cellular automata or sandpiles.

This is seen most clearly in the area of complex networks, where much of the work has focused on systems obtained through purely structural self-organization rules such as preferential attachment [6, 7, 2, 3] or rewiring [56, 3], and the resulting systems have been characterized primarily in terms of their global structural properties such as degree distribution or connectedness. Though functional attributes such as robustness have been studied extensively [5, 35], they are evaluated *post facto*, and not explicitly integrated into the self-organization algorithm itself. Even in studies that have systematically considered the optimization of functional properties [45, 51, 38, 50], the approach has been to *first* generate ensembles of networks using algorithms parameterized by generic quantities such as connectedness, attachment preference, degree distribution, etc., and *then* to characterize the results to find the best set of parameters (see [10] for an exception). However, as pointed out cogently by Doyle and coworkers [14, 20, 15, 16, 30, 19], even the best networks produced in such cases are typically far from optimal with regard to specific application domains and problem instances. The central challenge for complex systems engineering [14] is to find application-specific self-organization algorithms that have the desirable attributes of their abstract counterparts but are based on concrete system properties [34, 32]. In particular, the goal is to find self-organization algorithms capable of producing systems with *optimally structured heterogeneity*, which is how custom-designed systems achieve their performance [16].

In this chapter, we consider a relatively simple but important class of systems – wireless sensor networks – and present a set of self-organization rules that try to optimize a specific, application-relevant performance criterion. We show empirically that the resulting networks are indeed close to optimal, that their performance derives from the *specific* structuring of their heterogeneity rather than from simple generic attributes, and that they represent *atypical* samples in the overall configuration space. These results are of broad interest in the context of complex networks (and, indeed, complex systems) because they partially bridge the divide between purely open-loop self-organization [6, 7, 2, 3, 56] and explicit design. This work demonstrates that, in a reasonably complex and practical network, it is possible to obtain most of the benefits of optimized design through self-organization based only on local information. While we focus only on geometric networks (i.e., networks where connectivity is based on geometric neighborhoods in physical space), the ideas may extend to other distributed systems such as swarms [11], where agents make decisions based on local information while seeking to achieve a globally desirable configuration.

A major issue of interest in the complex networks literature is the robustness of such systems to random node failure and targeted attack. Networks with power-law degree distribution are much more robust to random failure than random networks, but more susceptible to targeted attack [6, 4, 5, 58]. While it arises from inherent structure in scale-free networks, Carlson and Doyle have argued that this "robust-yet-fragile" attribute also characterizes all highly optimized systems, which achieve their enhanced performance by trading off robustness to likely problems against fragility to unlikely ones [20, 15, 16, 30]. We test this empirically on networks generated through explicit optimization (using genetic algorithms), networks produced through heuristic self-organization methods, and those obtained by applying simple uniform rules (e.g., identical out-degree for all nodes). Our results show that, in the case of geometric networks, all configurations lose fitness with both random failure and targeted attack. However, the non-optimized configurations are affected in the same way in both cases while the optimized configurations show very different responses. Specifically, the latter lose much more fitness under targeted attack than under random failure. These results are in line with the "robust-yet-fragile" view of optimized systems.

2 Background and Motivation

Geometric networks are networks where nodes are distributed in a metric space, and each node connects to all others within a certain range, called the *connection radius* of the node. Such networks arise naturally in the context of wireless networks (including sensor networks), but, to some degree, neuronal networks in the brain can also be considered geometric [52]. Three attributes are of particular interest for such networks:

1. *Connectedness*. It is usually essential to have global connectivity, i.e., have the network form a single connected component. For example, in wireless sensor networks, information is typically obtained locally and processed collaboratively, which requires connectedness.
2. *Network diameter*. This is the average length of the shortest paths between all node pairs – typically measured in the number of edges traversed (termed hops). Since each hop in a network usually entails some cost (e.g., transmission energy, wait time, etc.), a small diameter is considered good because it implies more efficient communication over longer distances.
   It is usually more practical to use the *average inverse shortest path length (AISPL)*, which averages the inverse distances, and can readily represent disconnected node pairs with a value of 0 [10].
3. *Mean in-degree*. The number of incoming edges to a node usually represent the "loading" faced by that node, and determines the capacity needed by the node to handle its load. Thus, smaller mean in-degree is considered a desirable attribute. For example, in wireless networks, the number of nodes that can be heard at the

location of node i, and the mean in-degree of the network is a simple measure of congestion interference faced by that node.

In this chapter, we address the problem of obtaining connected geometric networks where the diameter and mean in-degree are minimized.

Most of the literature on geometric (typically wireless) networks has focused on *homogeneous networks*. These can be divided broadly into two classes: (1) *Degree-homogeneous (DH) networks*, where all nodes have the same out-degree, and (2) *Radius-homogeneous (RH) networks*, where all nodes have the same connection radius . It should be noted that each type of homogeneity (radius or degree) usually implies heterogeneity in the other parameter (degree or radius, respectively), but we reserve the term *heterogeneous networks* for those systems where neither radius nor degree are explicitly homogenized.

There is considerable research on degree-homogeneous networks – especially on determining the minimum number of neighbors necessary for a network to be connected [28, 49, 36, 57]. In particular, Xue and Kumar [57] show that for a network with N nodes, $C \log N$ neighbors for each node ensure connectivity, where $C > 1$. However, recent work indicates that such rules are very sensitive to inhomogeneities in node distribution [23]. In any case, choosing a fixed number of neighbors is a device for maintaining connectivity rather than minimizing network diameter or mean in-degree, which is our primary concern.

In radius-homogeneous networks, there is a critical radius above which the network is globally connected with high probability [18, 44]. This is called the *percolation radius* [48], ρ_{perc}. The simplest way to obtain a connected network with minimal mean in-degree is for all nodes to use a connection radius equal to (or just above) ρ_{perc}. However, this leads to a large diameter. Conversely, diameter can be minimized if all nodes use a radius much larger than ρ_{perc}, but this increases mean in-degree. Given the spatial distribution of nodes, one can find an "optimal" radius that represents the best compromise of diameter and mean in-degree, producing the optimal RH network. However, it has been shown that significantly better performance can be obtained by allowing heterogeneity, i.e., letting each node choose its own radius to maximize overall performance [9, 8, 21]. Unfortunately, assigning these radii is a very difficult combinatorial optimization problem in large systems, which has prompted several proposals for self-organized approaches [40, 12, 31, 13, 42, 41, 53, 54].

In the present chapter, we focus on the simplest possible class of heterogeneous networks, where only two radius choices are available – one smaller than ρ_{perc} and the other larger. The problem is to assign the appropriate value to each node such that a specific function of diameter and mean in-degree is optimized. The key issues are:

- To show whether – and to what extent – this heterogeneity allows optimization beyond that offered by the homogeneous case.
- To propose efficient, scalable self-organization algorithms for configuring near-optimal heterogeneous networks, and validating their performance.

Networks with two classes of nodes differentiated by the spatial extent of their connectivity are interesting from both abstract and concrete perspectives. Since nodes in geometric networks must connect with all other nodes within their transmission radius, these networks cannot have selective "short-cut" links in the sense of small-world connectivity [56], and even obtaining power-law connectivity [4] becomes quite difficult and contrived. However, many interesting properties of small-world and power-law network models actually arise from the existence of *hub nodes* – highly connected nodes that shrink the network's diameter and enhance its robustness [6, 4, 5, 58]. In the networks we study, the large radius nodes correspond to such hubs, and in this sense, these networks are the geometric analog of the classic complex network models. This chapter addresses the issue of whether they provide the same sort of advantages in terms of robustness and efficiency.

From a concrete viewpoint, two-level heterogeneous networks can be used to model many natural and artificial systems. For example, neuronal networks in the cerebral cortex consist of pyramidal cells with spatially wide-ranging connectivity and interneurons with more restricted range [22], and "hub-like" architectures in the cortex have been proposed as the crucial substrate of cognitive processes [47, 46]. In this chapter, we use wireless networks as the motivating case. Thus, the connection radius for a node is related to its transmission power, with larger radii requiring greater energy. Since nodes in wireless networks are typically energy-limited, there is considerable incentive to minimize diameter without increasing radii, i.e., transmission power, more than necessary [39, 1, 26, 17]. However, it should be noted that we do *not* explicitly try to minimize mean radius or total radius, and use the radii only as tunable parameters to minimize diameter and mean in-degree. We explicitly avoid specifying details such as handshake protocols, buffer sizes, etc., to maintain the generic character and scope of this study, which is intended to focus on the broad issue of self-organized network structures.

It should be noted that wireless networks with short-range and long-range nodes can arise naturally in many situations. For example, in chapter "Enhancing Underwater Acoustic Sensor Networks Using Surface Radios: Issues, Challenges and Solutions" of this volume, Cui et al. describe a network with short-range underwater nodes augmented by surface radio nodes with larger transmission radius.

3 System Description

We consider a system with N nodes distributed randomly in a unit square. Each node, i, has a connection radius r_i, and projects outgoing connections to every node j within this radius. This defines the network $G(V, E)$, where V is the set of all nodes and E the set of all edges representing connections between nodes. Because nodes can have different radii, the network is heterogeneous, and the edges are directed. We assume that each node has a unique (randomly generated) identifier, but this is used only for local identification and there is no globally available lists of node IDs. We also assume that nodes are aware of their position in a global

coordinate system, which may be based on GPS or self-organized within the network [37, 24, 27]. The position of node i is denoted by $l_i = (x_i, y_i)$, and the set $L \equiv \{l_i\}, i = 1, \ldots, N$, is termed a *layout*. For simplicity, we assume that each node can take radius values from a finite set, $R = \{\rho_k\}, k = 1, \ldots, M$. Thus, for a layout with N nodes, there are M^N possible assignments of radii. Each such assignment, $Q(L) \equiv \{r_i\}, i = 1, \ldots, N$, is called a *configuration*, and corresponds to a specific graph, $G(V_Q, E_Q)$, where V_Q is the set of nodes in the underlying layout, L, and E_Q is the set of edges induced on this layout by Q (Here (and henceforth), we have omitted writing the argument L for notational clarity.) The in-degree of node i in configuration Q is denoted by $f_i^I(Q)$, and its out-degree by $f_i^O(Q)$.

The AISPL for a configuration, Q, is defined as:

$$H(Q) = \frac{1}{N(N-1)} \sum_{i,j \in V(Q)} 1/d_{ij}$$

where d_{ij} is the distance in hops from node i to j, and $1/d_{ij} = 0$ if there is no path from i to j. The mean congestion for a configuration Q is defined as:

$$C(Q) = \frac{1}{N} \sum_{i \in V(Q)} f_i^I(Q)$$

Given a layout, L, the goal is to find a configuration Q^* that maximizes the *fitness function*, $\phi = H(Q)/C(Q)$, thus trying to minimize congestion while keeping the network diameter as small as possible.

3.1 Whisperers and Shouters

As discussed above, we focus in this chapter on the case of $M = 2$ (see [42, 41] for the $M > 2$ case.) Accordingly, we define two classes of nodes: (1) *Whisperers* are nodes with a small transmission radius, ρ_w, while (2) *shouters* have a larger radius, ρ_s. To obtain a uniform parametrization, the two radii are expressed in terms of the nominal percolation radius, ρ_{perc} as $\rho_w = \alpha\rho_{\text{perc}}$ and $\rho_s = \beta\rho_{\text{perc}}$, where $\alpha < 1 < \beta$. The nominal ρ_{perc} is calculated as in [44]. Each node, i, maintains two adjacency lists for the nodes within its transmission range. Nodes within whisperer range of i are listed in its *whisper-adjacency list*, A_i^w, while the nodes in shouter range but not in whisperer range, called the *ring nodes* of i, are listed in the *ring-adjacency list*, A_i^r. These adjacency lists are the primary source of information as each node autonomously decides its transmission radius using local information in order to get close to an optimum ϕ value.

4 Self-Organization Algorithms

The scalability and flexibility of self-organization methods derives from three attributes:

1. *Decentralization of action*. Global organization emerges as a result of interaction between autonomous elements rather than by global prescription.
2. *Simplicity of decisions*. The decision space in which each element operates is very simple, and only has a few degrees of freedom.
3. *Locality of information*. The information used by each element in its decision-making is local (or limited in some other way).

The first two of these roughly determine what are termed *design degrees-of-freedom* (DDOF) [43]. Any successful self-organization algorithm must satisfy these conditions, and be able to generate near-optimally structured, heterogeneous configurations with high performance.

We present a series of self-organization algorithms based on a specific heuristic principle. While the algorithms are similar in terms of the decentralization of action and simplicity of decision criteria, each algorithm in the sequence takes increasingly more neighborhood information into account.

Each node in the system faces a binary choice of whether to be a whisperer or a shouter. The heuristic principle on which our algorithms are based is as follows: *If a node determines that it can reach all its ring-adjacent neighbors indirectly with connection radius ρ_w, it becomes a whisperer. Else, it must become a shouter*. When each node makes its choice using this criterion, the connectivity of the network is ensured, as many nodes as possible become whisperers, and shouters are deployed only when they are truly useful (i.e., would create connectivity beyond what is possible with whisperers alone). The nodes apply this criterion asynchronously and iteratively until the network is relaxed. The key difference among the algorithms is in the information used to determine whether the criterion is satisfied.

The heuristic described above tries to increase fitness in two ways: (1) It reduces the denominator of the fitness function (congestion) explicitly; and (2) It keeps the numerator of the fitness function (AISPL) relatively high by preventing disconnection. Though the first operation (reducing congestion) tends to *decrease* the AISPL and thus works against the second operation, this effect is largely swamped by the benefit of reduced congestion – especially in larger networks. More complex heuristics can mitigate this problem somewhat, but we use the current heuristic in the interest of simplicity.

The algorithms are as follows:

4.1 Basic Self-Organization (BSO) Algorithm

This is the basic algorithm with the least amount of information available to nodes. The process begins by each node transmitting a message giving its identifier and its

position (a so-called "hello" *message*) at shouter transmission power. As a result, all nodes acquire a list of nodes within shouter range of themselves, and their locations. From this, each node, i, can obtain its whisper-adjacency list, A_i^w, and its ring-adjacency list, A_i^r. Using these lists, each i determines if all nodes in A_i^r are within ρ_w of some node in A_i^w. If all nodes satisfy this criterion, $r_i = \rho_w$, else $r_i = \rho_s$.

This algorithm represents the most pessimistic and naive version of the heuristic principle stated above: pessimistic because it assumes that all neighbors choose whisperer radii, and naive because it checks only for 1-hop links from nodes in A_i^w to those in A_i^r. Because of this, the algorithm makes maximally conservative choices, resulting in guaranteed connectivity but more shouters than needed to achieve it. The latter leads to high AISPL but also relatively high congestion.

4.2 Self-Organization Algorithm A

This algorithm relaxes the "naive" assumption in the BSO algorithm slightly. Each node obtains additional information on the current radius of the whisper-adjacent nodes. If a node, j, whisper-adjacent to i is a shouter, node i checks if the nodes in A_i^r are within ρ_s of j rather than within ρ_w. Node i also checks whether a ring-adjacent node not satisfying this condition is within ρ_w of a ring-adjacent node that does (see Fig. 1). Node $r_i = \rho_s$ only if some node in A_i^r fails these tests. The additional cost of this step is minimal since the additional information used is the radius choices of i's whisper-adjacent nodes. The result is that every node that was a whisperer under BSO remains a whisperer and some that were shouters now become whisperers. Thus, Algorithm A tends to reduce congestion, but also reduces AISPL.

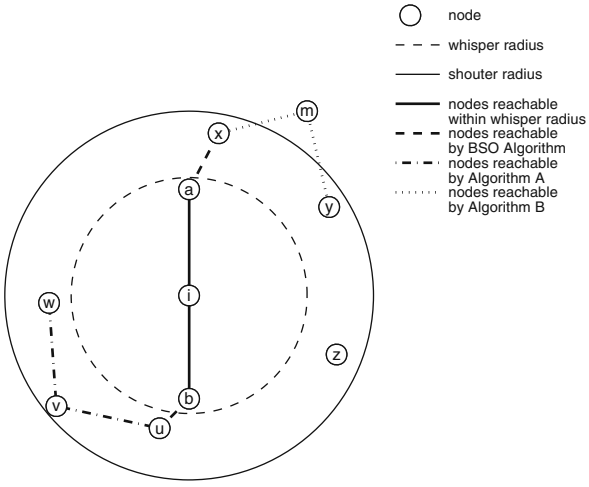


Fig. 1 A schematic representation of node i running the self organizing algorithm. Nodes a and b are whisper-adjacent to i, while u, v, w, x, y and z are ring-adjacent

4.3 Self-Organization Algorithm B

As shown in Fig. 1, it is possible that a node ring-adjacent to i is reachable, but requires a path through a node that is not a neighbor of i (about which i has no information in the previous algorithms). Algorithm B addresses this issue, and also relaxes the "pessimistic" assumption of the BSO algorithm. Here, each node, i, also obtains two additional items of information: (1) The current radius choices of all its ring-adjacent nodes; and (2) The whisper adjacency lists of all its ring-adjacent nodes. This information is obtained through additional communication following the initial "hello" message. Using the first item of information, i checks for connectivity from nodes in A_i^w using the *actual* current radii of neighbors rather than assuming whisperer radii for all ring-adjacent nodes. It also checks if each ring-adjacent node that is not reachable from A_i^w has an adjacent node that can be reached from a reachable ring-adjacent node. If so, the previously unreachable node is marked reachable because it has a path from A_i^w through a non-neighbor *link node* (see Fig. 1). If any ring-adjacent nodes for i are still not reachable, i sets $r_i = \rho_s$, otherwise $r_i = \rho_w$. As with Algorithm A, this further reduces the total number of shouters, decreasing congestion further but also decreasing AISPL.

4.4 Self-Organization Algorithm C

The final algorithm we consider augments Algorithm B by using the *actual* radii of the link nodes described above. This requires that i have information about radii of nodes that are not within its ρ_s radius. This is still possible to obtain locally by having the ring-adjacent neighbors of i forward the radius information about their neighbors (which they have in Algorithm B) to i. Algorithm C has the effect of further reducing the number of shouters over Algorithm B.

Obviously, this progression of algorithms can be continued by having each node obtain information about a wider area around it, but that also diminishes scalability which is the main reason for using self-organization. Thus, we focus on simulations done with Algorithms A, B and C. As pointed out above, each successive algorithm reduces the number of shouters in the network, and an interesting issue is to see whether the gain due to reduced congestion is more than the loss due to reduced AISPL.

To address the main issue of the chapter (i.e., optimality), we also explicitly obtain optimized configurations for each layout using a genetic algorithm [33]. The algorithm is run repeatedly with high-mutation episodes, etc., to ensure reasonably that the configurations obtained are, indeed, very close to optimal in terms of fitness. This is a crucial part of our study. In most cases previous proposals for self-organization of wireless networks [59, 55, 29, 25], there was no attempt to check how the results obtained compared with the best achievable. However, we are also constrained by the computational difficulty of obtaining optimal configurations, and this has been done only for relatively small networks (200 nodes). Given that the self-organization algorithms produced solutions close to optimal in this case, we

expect that the same will hold true for larger systems. This is verified indirectly by comparing the fitness of randomly generated configurations with those obtained through self-organization.

Results from our simulations are presented and discussed in the next section.

5 Simulation, Results and Discussion

The simulations described below were done to address the following specific questions:

1. Are networks optimized through heuristic self-organization comparable in quality to those obtained by direct optimization?
2. Are the heterogeneous networks obtained through heuristic and direct optimization significantly different from the best radius-homogeneous and degree-homogeneous networks?
3. Are the optimized networks significantly different from statistically similar randomly generated networks?
4. How robust are the various optimized and homogeneous networks to node failure and targeted attack?

5.1 Simulations

Simulations were run for networks with 200 and 1000 nodes. The whisperer radius was chosen as 0.8 times percolation radius ($\alpha = 0.8$) and shouter radius as 1.25 times percolation radius ($\beta = 1.25$). Percolation radius for 200 node network was set to 0.11, giving $\rho_w = 0.088$ and $\rho_s = 0.1375$. Percolation radius for 1000 node network was set at 0.0463, giving $\rho_w = 0.03704$ and $\rho_s = 0.05785$.

The genetic algorithm (GA) (see Appendix), which was run only for the 200 node case for computational reasons, used a population size of 20, and was run for 4000 generations. It used the fitness ϕ, which was also used to evaluate all other networks. Each solution in the GA population was encoded as a vector of length N (i.e., number of nodes), with the ith element indicating whether the node i was a whisperer or a shouter. New solutions were generated using two-point crossover of parents selected through an elitist method. Mutations were applied by switching whisperers to shouters and vice-versa with a probability of 0.05. To exclude the possibility for premature convergence, the GA was run 3 times for each layout. Each run was "punctuated" by mutation shocks (high-mutation iterations) whenever the algorithm converged to a putative optimum, with at least 3 such punctuations in each run. Only if the algorithm converged repeatedly to very similar fitness values within and across runs was the best fitness achieved by the GA treated as optimal for a layout.

Both degree- and radius-homogeneous networks were compared with optimized networks. For each layout, a range of degree and radius choices were evaluated for the DH and RH cases, respectively, and the best one was used for the comparison.

To compare optimized networks with randomly generated ones, it is necessary to choose the latter in a way that makes the comparison justified. We did this by scrambling the radius assignments for networks obtained through the GA for the 200-node case and through Algorithm C for the 1000-node case. Thus, the ratio of shouters and whisperers in all *scrambled networks* is identical to that in the best available configurations for those layouts. The only difference is in the assignment. The relative difference in quality between the two is, therefore, entirely the result of specific assignment rather than the generic attribute of radius distribution – reflecting the argument made by Doyle and others [16, 30] that optimal systems achieve their performance through *specific configuration*, and are atypical rather than generic within their class.

In addition to the scrambled networks generated as above, we also generated random networks with the fraction of shouters distributed around the value for the optimized cases rather than set equal to it. These networks are termed *random networks*.

5.2 Results and Discussion

5.2.1 Performance Comparison Between Algorithms

Figure 2 shows the best fitness achieved by the GA and the three heuristic optimization algorithms (A, B and C) over 10 different 200-node layouts. Clearly, the GA outperforms the heuristic algorithms by a small amount in all cases, but the latter come close to achieving the optimal fitness. Also, there is a small but consistent improvement in performance from Algorithm A through Algorithm B to Algorithm C.

While it is important that the heuristic algorithms produce configurations with close to optimal fitness, the utility of these algorithms must ultimately be judged in comparison with simpler (and faster) non-optimizing algorithms. Figures 3 and 4 show results comparing Algorithm C – the best of the heuristic algorithms – with the two homogeneous configuration methods – RH and DH – and with scrambled networks to provide a baseline for performance. Clearly, Algorithm C outperforms all the non-optimizing configurations consistently for both 200-node and 1000-node layouts, though it is interesting to note that degree-homogeneous (DH) configurations are also quite good, and much better than radius-homogeneous (RH) configurations which are not much better than the scrambled case. This is presumably because degree-homogeneity is a stronger guarantor of connectivity than radius-homogeneity.

5.2.2 Comparison with Non-optimized Networks

As discussed earlier, it has been argued that optimized configurations in application systems must be atypical, i.e., they must fall well outside the space of generic configurations. To test this for our optimized configurations, we randomly scrambled

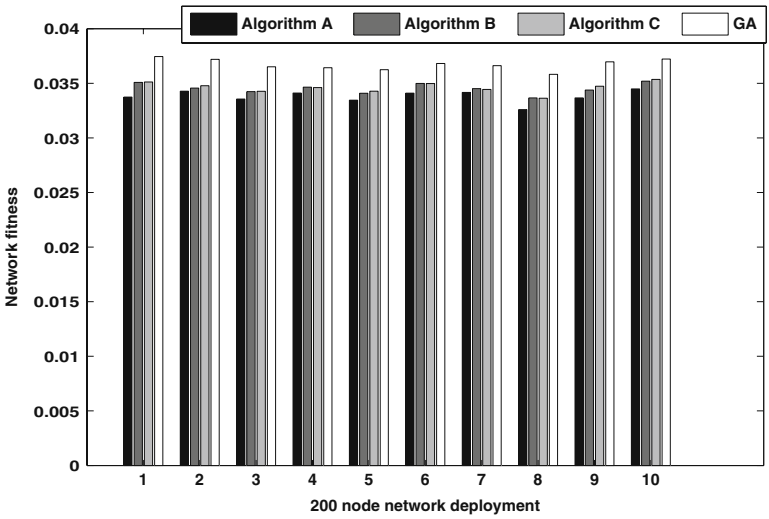


Fig. 2 Comparison of fitness for networks produced with Algorithms A,B and C and the GA for 10 different 200-node layouts. The *graphs* show that self-organized networks produce networks close to those obtained through the GA. Note that performance improves further as the self-organized algorithms use more information

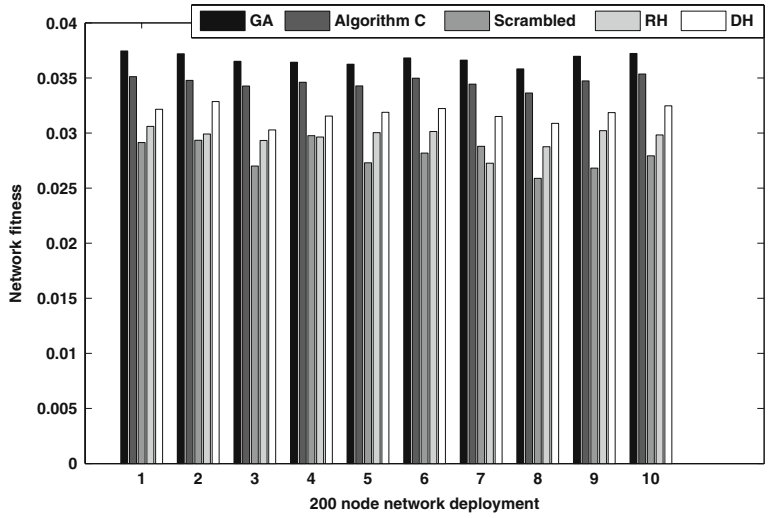


Fig. 3 Comparison of GA, Algorithm C, scrambled and homogeneous networks in the 200-node case, showing that self-organized networks are better than scrambled or homogeneous ones. The layouts are the same as in Fig. 2

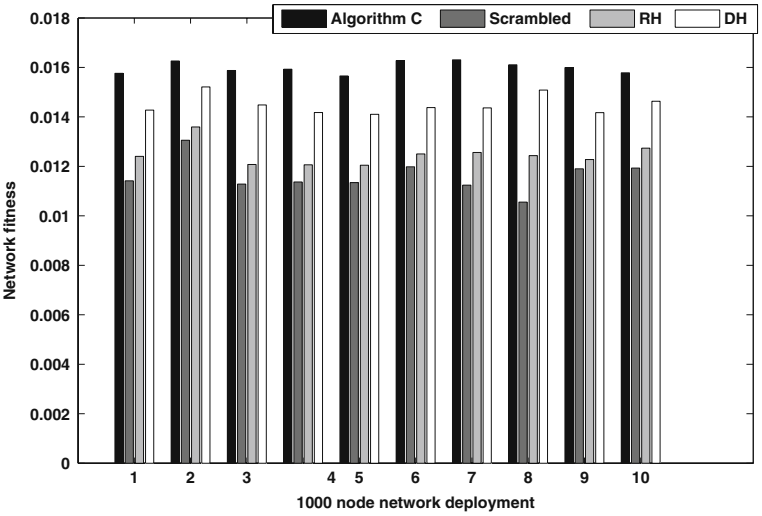


Fig. 4 Comparison of Algorithm C, scrambled and homogeneous networks in 1000-node case, showing the advantage of self-organized networks over scrambled or homogeneous ones. The x-axis indexes 10 independent random layouts of 1000 nodes each

the radius assignments for the best optimized network in each case to generate a population of networks that had the same radius distribution as the optimal. Figure 5 shows the fitness distribution for this class of 1000-node networks as well as the fitnesses obtained for the DH and RH cases and the optimized cases. Clearly, the optimized algorithms all produce fitness well outside the generic range, indicating that the optimization has produced statistically significant benefits. Of the two homogeneous cases, RH networks tend to fall in the upper tail of the scrambled network distribution, indicating that they are just very good random samples. However, the DH networks are well outside the generic fitness range, though still significantly worse than optimized networks.

Since the scrambled networks all have the same whisperer-shouter ratio, it is worth investigating whether varying this parameter my produce better configurations. To check this, we generated random networks for each layout with shouter-fractions ranging from 15% below that in the optimal case to 15% above it. The results for these random networks are shown in Fig. 6, and are virtually identical to those for scrambled networks. The main conclusion from both these comparisons is that high-quality (near-optimal) networks cannot be produced by sampling and sifting randomly generated configurations – even those with carefully chosen shouter-fractions – and an explicit optimization process must be used. This emphasizes the necessity for efficient, scalable self-organizing optimization algorithms such as Algorithm C.

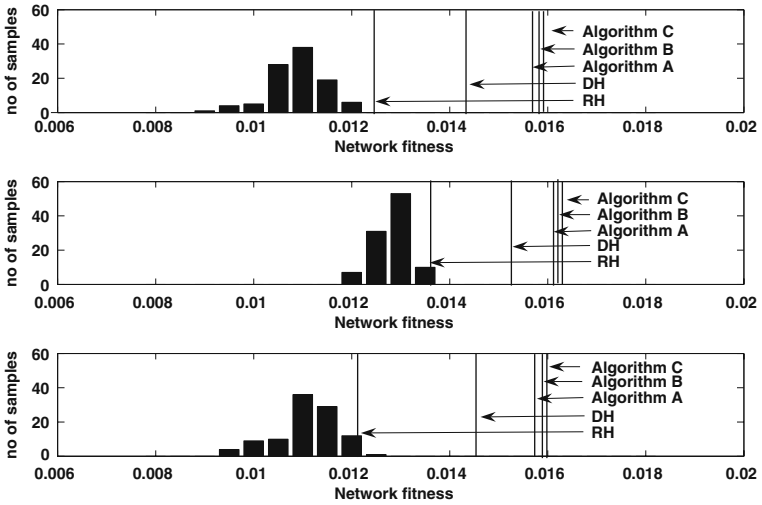


Fig. 5 Fitness distribution for 100 randomly scrambled networks for a particular layout, obtained from the best available configuration for the layout. The *vertical lines* indicate the best fitnesses obtained through various optimization procedures

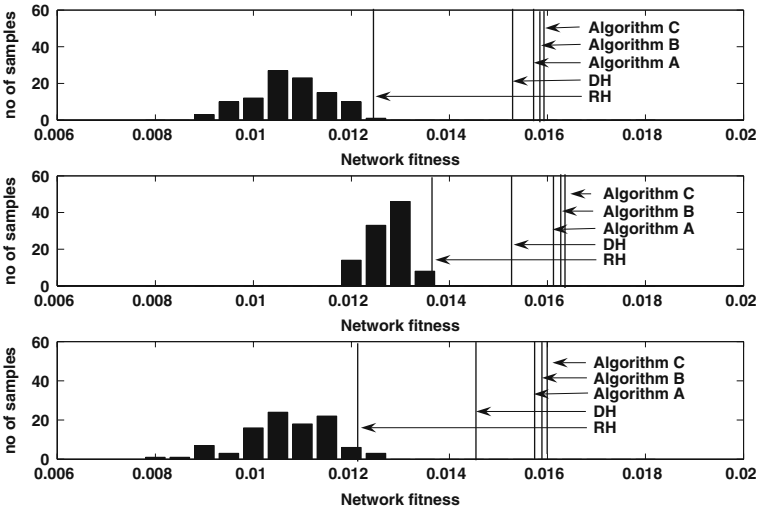


Fig. 6 Fitness distribution for 100 randomly generated networks for a particular layout, with shouter fractions distributed in a 30% window around that of the best available configuration for the layout. The *vertical lines* indicate the best fitnesses obtained through various optimization procedures

5.2.3 Robustness Evaluation

To consider the issue of robustness, each network configuration was subjected to systematic deterioration using random failure and targeted attack models. In the random failure case, randomly chosen nodes were removed from the network along with all connections to and from these nodes, while targeted attack involved removing nodes in decreasing order of out-degree. Figures 7 and 8 show the *percentage loss* of fitness as an increasing number of nodes is removed. Clearly, all configurations except those produced by Algorithm A have very similar response to random failure (Fig. 7), but the responses are very different for targeted attack. As seen in Fig. 8, the optimized configurations lose fitness much more rapidly in this case. Indeed, the robustness appears to depend inversely on the degree of optimization, with the GA-derived networks faring the worst followed by Algorithms C, B and A, with the two homogeneous cases showing the greatest robustness. This is seen very clearly in Fig. 9, which plots the fitness of networks with 25% node loss against the fitness of the undamaged networks. Clearly, highly fit optimized networks show a strong divergence in their response to random failure and targeted attack while the non-customized homogeneous networks are almost identically affected in the two cases. Note that, by definition, the random and targeted failure cases are identical for DH networks.

Interestingly, Algorithm A configurations, which are the most robust in the random failure case, behave like non-optimized configurations for low levels of targeted attack but become more similar to optimized configurations for higher attack levels. Thus, overall, Algorithm A appears to be the best choice for balancing the benefits of optimization with robustness.

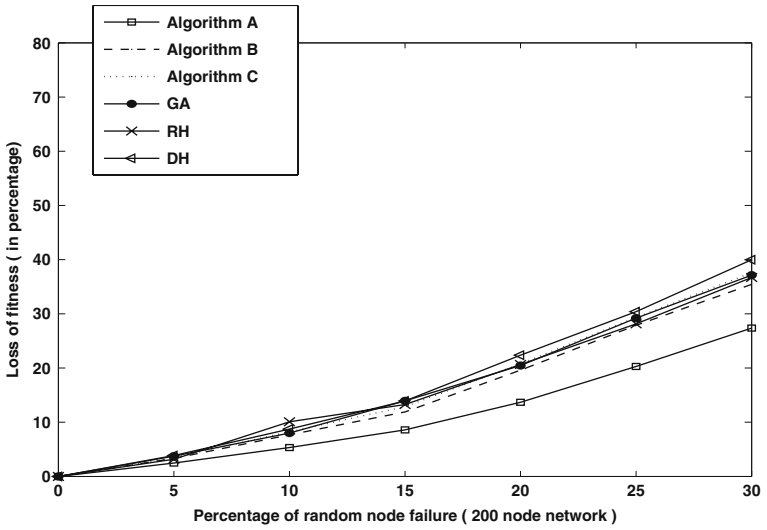


Fig. 7 Robustness of networks based on fitness loss with random node failure

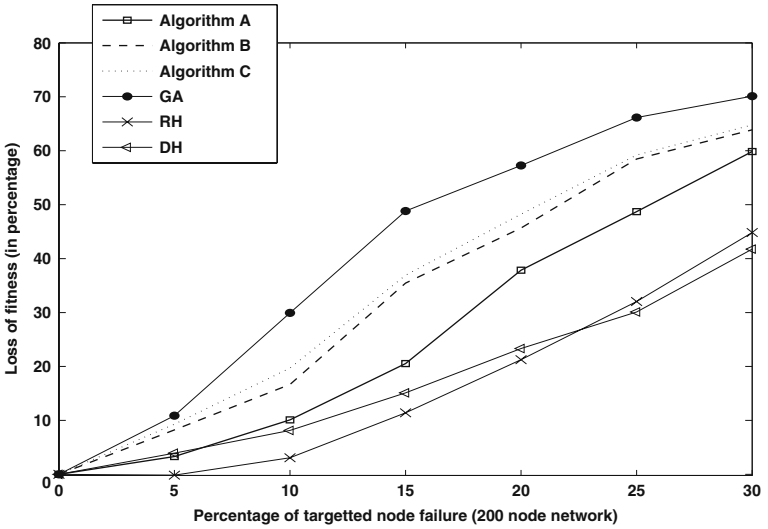


Fig. 8 Robustness of networks based on fitness loss with targeted node failure

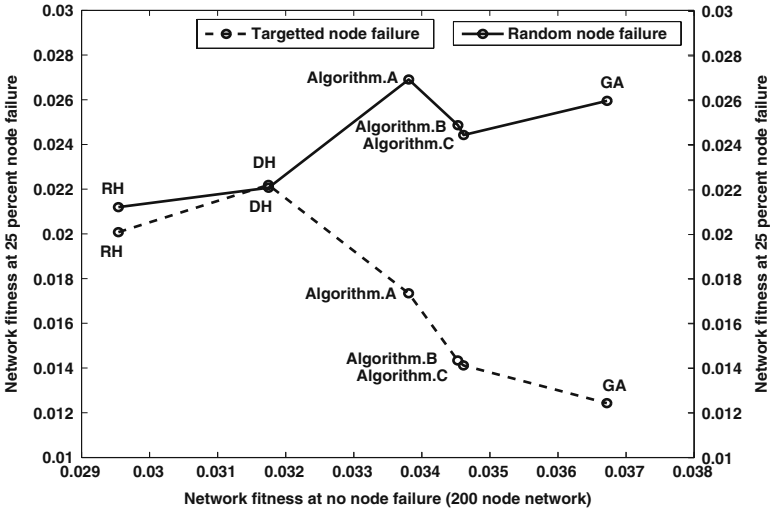


Fig. 9 Plot of fitness at 25% node failure against fitness in the no-failure case for different networks. The *plots* clearly show that the heterogeneous networks respond very differently to random and targeted failures while non-optimized homogeneous networks have virtually identical responses

6 Conclusion

In this study, we have investigated heterogeneous geometric wireless networks with two types of nodes: whisperers, projecting over a small radius, and shouters projecting over a larger radius. Several interesting conclusions about these heterogeneous networks have emerged from the study:

1. Given a node layout, very specific assignments of node type are needed to achieve optimal performance, which is significantly better than that of the best homogeneous networks.
2. The performance of optimized networks is not just a consequence of the fraction of hub nodes (shouters) in the population, but depends sensitively on the precise choice of these nodes. Thus, optimal configurations are *atypical* rather than generic in the space of configurations with the same radius distribution.
3. It is possible to find simple, efficient and scalable self-organization algorithms to produce near-optimal network configurations.
4. Highly optimized network configurations are as robust as non-optimized ones with respect to random node failure, but are much more susceptible to targeted attacks that preferentially remove nodes with the highest connectivity. All heterogeneous networks thus show a strong "robust-yet-fragile" effect [20, 15, 16, 30].
5. Of all the optimization procedures investigated, Algorithm A, which used the least amount of neighborhood information, provided the best balance between performance and robustness. The results suggest that most of the benefits of heterogeneity can be obtained with minimal loss of robustness by using simple self-organization algorithms rather than more complex ones.

The study has also produced an interesting conclusion about homogeneous networks, showing that, with randomly deployed nodes, networks that seek homogeneous out-degree are much better than networks that simply use the same connection radius for all nodes – though both are worse than heterogeneous configurations. Interestingly, there is evidence that neuronal networks in the cerebral cortex configure themselves by trying to equalize the synaptic input for each neuron rather than equalizing the size of the axonal arbors (i.e., projection radius) [52].

Appendix

In order to assess the quality of the results obtained through self-organization, we generated optimal (or near-optimal) configurations for the 200-node layouts studied in the simulations. A genetic algorithm was used to produce these optimized configurations. The algorithm comprised the following steps.

Given a layout L with N nodes, set $generation = 1$:

1. Generate an initial population of n_{pop} configurations by assigning whisperer and shouter radii randomly to each node in n_{pop} copies of the layout. Thus, each configuration is represented by a binary string, $\zeta^k = \{\zeta_i^k\}$ of n_{pop} bits, where

$\zeta_i^k = 1(0)$ means that node i in configuration k is a shouter(whisperer). These strings are the *chromosomes* representing the respective configurations.
2. Evaluate the fitness of each configuration in the current population using the fitness function defined above.
3. Select $0.8n_{pop}/2$ fittest configurations and $0.2n_{pop}/2$ randomly chosen configurations as *parents* for the next generation.
4. Randomly pair the parents into $n_{pop}/2$ couples, and generate two *offspring* from each couple using two-point crossover, i.e., both chromosomes in the couple are divided into three parts by breaking them at the same two randomly chosen points, and recombined into two new chromosomes of length N.
5. Select the $n_{pop}/4$ fittest configurations and $n_{pop}/4$ randomly chosen configurations as *survivors* from the current generation, i.e., not including the new offspring.
6. Remove the remaining members of the current generation and replace them with the offspring to obtain a new population of size n_{pop}.
7. In all but the two fittest survivors:
   - If mutation is to be normal, randomly switch 0.05% of the bits from 1 to 0 or vice versa.
   - If the current generation is selected for a *mutation shock*, randomly switch 0.25% of the bits from 1 to 0 or vice versa. The mutation shock is intended to break the system out of a local optimum and prevent premature convergence.
8. If termination condition (maximum number of generations) is not met, increment *generation* by 1 and repeat from step 2.

References

1. D. Braha, A.A. Minai, and Y. Bar-Yam, Eds., *Complex Engineered Systems: Science Meets Technology*, Springer/NECSI, New York, 2006.
2. A.-L. Barabási and R. Albert, "Emergence of scaling in random networks", *Science*, vol. 286, pp. 509–511, 1999.
3. A.-L. Barabási, R. Albert, and H. Jeong, "Mean-field theory for scale-free random networks", *Physica A*, vol. 272, pp. 173–187, 1999.
4. R. Albert and A.-L. Barabási, "Topology of evolving networks: Local events and universality", *Physical Review Letters*, vol. 85, pp. 5234–5237, 2000.
5. R. Albert and A.L. Barabási, "Statistical mechanics of complex networks", *Reviews of Modern Physics*, vol. 74, pp. 47–97, 2002.
6. D.J. Watts and S.H. Strogatz, "Collective dynamics of 'small-world' networks", *Nature*, vol. 393, pp. 440–442, 1998.
7. R. Albert, H. Jeong, and A.-L. Barabási, "Error and attack tolerance of complex networks", *Nature*, vol. 406, pp. 378–382, 2000.
8. M.E.J. Newman, S.H. Strogatz, and D.J. Watts, "Random graphs with arbitrary degree distribution and their applications", *Physical Review E*, vol. 64, pp. 026118, 2001.
9. B. Shargel, H. Sayama, I.J. Epstein, and Y. Bar-Yam, "Optimization of robustness and connectivity in complex networks", *Physical Review Letters*, vol. 90, pp. 168701, 2003.
10. A.X.C.N. Valente, A. Sarkar, and H.A. Stone, "Two-peak and three-peak optimal complex networks", *Physical Review Letters*, vol. 92, pp. 118702, 2004.

11. G. Paul, T. Tanizawa, S. Havlin, and H. E. Stanley, "Optimization of robustness of complex networks", *European Physical Journal B*, vol. 38, pp. 187–191, 2004.
12. T. Tanizawa, G. Paul, R. Cohen, S. Havlin, and H. E. Stanley, "Optimization of network robustness to waves of targeted and random attacks", *Physical Review E*, vol. 1, no. 4, pp. 047101, Apr. 2005.
13. A. Beygelzimer, G. Grinstein, R. Linsker, and I. Rish, "Improving network robustness by edge modification", *Physica A*, vol. 357, pp. 593–612, 2005.
14. J. Doyle and J.M. Carlson, "Power laws, highly optimized tolerance, and generalized source coding", *Physical Review Letters*, vol. 84, pp. 5656–5659, 2000.
15. J.M. Carlson and J. Doyle, "Highly optimized tolerance: Robustness and design in complex systems", *Physical Review Letters*, vol. 84, pp. 2529–2532, 2000.
16. J.M. Carlson and J. Doyle, "Complexity and robustness", *Proceedings of the National Academy of Sciences USA*, vol. 99 Suppl. 1, pp. 2539–2545, 2002.
17. L. Li, D. Alderson, R. Tanaka, J.C. Doyle, and W. Willinger, "Towards a theory of scale-free graphs: Definition, properties, and implications (extended version)", Tech. Rep. CIT-CDS-04 –006, Engineering and Applied Science, California Institute of Technology, 2005.
18. J.C. Doyle, D.L. Alderson, L. Li, M. Roughan, S. Shalunov, R. Tanaka, and W. Willinger, "The 'robust yet fragile' nature of the internet", *Proceedings of the National Academy of Sciences USA*, vol. 102, pp. 14497–14502, 2005.
19. R. Nagpal, "Engineering amorphous systems, using global-to-local compilation", in *Complex Engineered Systems: Science Meets Technology*, D. Braha, A.A. Minai, and Y. Bar-Yam, Eds., pp. 291–306. Springer/NECSI, New York, 2006.
20. A.A. Minai, D. Braha, and Y. Bar-Yam, "Complex engineered systems: A new paradigm", in *Complex Engineered Systems: Science Meets Technology*, D. Braha, A.A. Minai, and Y. Bar-Yam, Eds., pp. 1–21. Springer/NECSI, New York, 2006.
21. E. Bonabeau, M. Dorigo, and G. Theraulaz, Eds., *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, Oxford, 1999.
22. R. Albert, H. Jeong, and A.-L. Barabási, "Diameter of the world wide web", *Nature*, vol. 401, pp. 130–131, 1999.
23. S.-H. Yook, H. Jeong, and A.-L. Barabási, "Modeling the internet's large-scale topology", *Proceedings of the National Academy of Sciences USA*, vol. 99, pp. 13382–13386, 2002.
24. A. Van Ooyen and J Van Pelt, "Activity-dependent outgrowth of neurons and overshoot phenomena in developing neural networks", *Journal of Theoretical Biology*, vol. 167, pp. 27–43, 1994.
25. L. Kleinrock and J. Sylvester, "Optimum transmission radii for packet radio networks or why six is a magic number", in *NTC '78; National Telecommunications Conference, Birmingham, Ala., December 3–6, 1978, Conference Record. Volume 1. (A79–40501 17–32) Piscataway, NJ, Institute of Electrical and Electronics Engineers, Inc., 1978, p. 4.3.1–4.3.5.*, 1978, pp. 431–435.
26. L. Takagi and H. Kleinrock, "Optimal transmission ranges for randomly distributed packet radio terminals", *IEEE Transactions on Communications*, vol. 32, no. 3, pp. 246–257, 1984.
27. J. Ni and S.A.G. Chandler, "Connectivity properties of a random radio network", *IEE Proceedings – Communications*, vol. 141, pp. 289–296, 1994.
28. F. Xue and P.R. Kumar, "The number of neighbors needed for connectivity of wireless networks", *Wireless Networks*, vol. 10, no. 2, pp. 169–181, 2004.
29. O. Ferrari and G. Tonguz, "Minimum number of neighbors for fully connected uniform ad hoc wireless networks", in *Proceedings of IEEE International Conference on Communications*, June, pp. 4331–4335, 2004.
30. Y.-C. Cheng and T.G. Robertazzi, "Critical connectivity phenomena in multihop radio models", *IEEE Transactions on Communications*, vol. 37, pp. 770–777, 1989.
31. P. Santi, "The critical transmitting range for connectivity in mobile ad hoc networks", *IEEE Transactions on Mobile Computing*, vol. 4, no. 3, pp. 310–317, 2005.
32. D. Stauffer and A. Aharony, *Introduction to Percolation Theory*, Taylor & Francis, London, UK, 1994.

33. C. Bettstetter, “On the minimum node degree and connectivity of a wireless multihop network”, in *MobiHoc ’02: Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pp. 80–91, 2002.
34. C. Bettstetter, “On the connectivity of wireless multihop networks with homogeneous and inhomogeneous range assignment”, in *Proceedings of the IEEE Vehicular Technology Conference*, 2002.
35. E. Duarte-Melo and M. Liu, “Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks”, in *Proceedings of IEEE GLOBECOM 2002*, November 2002.
36. R. Ramanathan and R. Rosales-Hain, “Topology control of multihop wireless networks using transmit power adjustment”, in *Proceedings of IEEE INFOCOM 2000*, pp. 404–413, 2000.
37. S. Borbash and E. Jennings, “Distributed topology control algorithm for multihop wireless networks”, in *Proceedings of the 2002 World Congress on Computational Intelligence*, 2002.
38. N. Li and J.C. Hou, “Localized topology control algorithms for heterogeneous wireless networks”, *IEEE/ACM Transactions on Networking*, vol. 13, pp. 1313–1324, 2005.
39. G Srivastava, P. Boustead, and J. Chicharo, “Connected fixed node degree based topologies in ad hoc networks”, in *Proceedings of the 12th IEEE International Conference on Networks (ICON 2004)*, pp. 1330–1340, 2006.
40. P. Ranganathan, A. Ranganathan, A. Minai, and K. Berman, “A self-organizing heuristic for building optimal heterogeneous ad hoc sensor networks”, in *Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC ’06)*, pp. 774–779, 2006.
41. P. Ranganathan, A. Ranganathan, K. Berman, and A. Minai, “Discovering adaptive heuristics for ad-hoc sensor networks by mining evolved optimal configurations”, in *Proceedings of the 2006 IEEE International Conference on Evolutionary Computation (CEC’06)*, pp. 3064–3070, 2006.
42. A. Venuturumilli and A.A. Minai, “Obtaining robust wireless sensor networks through selforganization of heterogeneous connectivity”, in *Proceedings of the 6th International Conference on Complex Systems*, 2006.
43. Y. Wang, X. Wang, D.P. Agrawal, and A.A. Minai, “Impact of heterogeneity on coverage and broadcast reachability in wireless sensor networks”, in *Proceedings of the 15th International Conference on Computer Communications and Networks*, pp. 63–67, 2006.
44. D.J. Felleman and D.C. Van Essen, “Distributed hierarchical processing in the primate cerebral cortex”, *Cerebral Cortex*, vol. 1, pp. 1–47, 1991.
45. O. Sporns, G. Tononi, and G.M. Edelman, “Theoretical neuroanatomy: Relating anatomical and functional connectivity in graphs and cortical connection matrices”, *Cerebral Cortex*, vol. 10, pp. 127–141, 2000.
46. O. Sporns and G. Tononi, “Classes of network connectivity and dynamics”, *Complexity*, vol. 7, pp. 28–38, 2002.
47. V. Raghunathan, C. Schurgers, S. Park, and M.B. Srivastava, “Energy-aware wireless microsensor networks”, *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 40–50, 2002.
48. I.F. Akyildiz, W. Su, Y. Sankaasubramaniam, and E. Cayirci, “A survey on sensor networks”, *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.
49. W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks”, in *Proceedings of the Hawaii International Conference on System Science, Maui*, 2000.
50. B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, “SPAN: An energy-efficient coordination algorithm for topology maintenance in ad-hoc wireless networks”, in *Mobile Computing and Networking*, pp. 85–96, 2001.
51. N. Patwari, J.N. Ash, S. Kyperountas, A.O. Hero, R.L. Moses, and N.S. Correal, “Locating the nodes: Cooperative localization in wireless sensor networks”, *IEEE Signal Processing Magazine*, vol. 22, pp. 54–69, 2005.
52. S. Gezici, Z. Tian, G.B. Giannakis, H. Kobayashi, A.F. Molisch, H.V. Poor, and Z. Sahinoglu, “Localization via ultra-wideband radios”, *IEEE Signal Processing Magazine*, vol. 22, pp. 70–84, 2005.

53. S. Kim, A.P. Brown, T. Pals, R.A. Iltis, and H. Lee, "Geolocation in ad hoc networks using DS-CDMA and generalized successive interference cancellation", *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 984–998, 2005.
54. D. Reynolds, J.M. Carlson, and J. Doyle, "Design degrees of freedom and mechanisms of complexity", *Physical Review E*, vol. 66, pp. 016108, 2005.
55. M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, MA, 1998.
56. H. Zhang and A. Arora, "*gs*3: Scalable self-configuration and self-healing in wireless networks", *Computer Networks*, vol. 43, pp. 459–480, 2003.
57. X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration in wireless sensor networks", in *Proceedings of SenSys'03*, Los Angeles, CA, November 2003.
58. B. Krishnamachari, S. Wicker, R. Bejar, and C. Fernandez, "On the complexity of distributed self-configuration in wireless networks", *Telecommunication Systems*, vol. 22, pp. 33–59, 2003.
59. H. Gupta, S.R. Das, and Q. Gu, "Connected sensor cover: Self-organization of sensor networks for efficient query execution", in *Proceedings of MobiHoc'03*, Annapolis, MD, June 2003.

Cooperative Strategies in Dense Sensor Networks

Anna Scaglione, Y.-W. Peter Hong, and Birsen Sirkeci Mergen

Abstract In this chapter we discuss cooperative source and channel coding strategies and their performance in very dense sensor networks. In particular we investigate how the efficient acquisition of correlated data mandates that the node transmit cooperatively, instead of contending to report their local information. This is important as correlated information it is one of the remaining resource to tap into, to overcome the capacity limitations of large scale wireless networks. We indicate a simple approach to attain this objective by converting the data retrieval process into a querying strategy, where each query is used to convert the correlated data into cooperative codes, which are broadcasted over the wireless channel. We show that over a Markovian field the method would provide a reliable mean to broadcast the correlated information to the whole network with a delay that is on average in the order of the aggregate entropy of the data field.

Collaboration aids sensor networks operations. In this chapter we argue that the complexity of a data collection problem and the energy consumption needed can be made invariant with respect to the network size for a given underlying sensor field. The way communication requirements can be decreased is by adapting to the underlying data structure, and inference objective in a proactive way. Wireless networks, and sensor networks in particular, are *swelling* with correlated information. Basic information theoretic results indicate clearly that data correlation among the transmitter expands the capacity bounds a multiple access channels. We offer evidence of this fact through an example. The procedure we identify is not optimal, but it highlights how this intrinsic property of many wireless networking problems can be turned into a useful resource, especially in networks with large deployments of low complexity elements, for example simple energy scavenging RFIDs.

Hence, the decision on the network deployment does not have necessarily to favor a configuration of very accurate sensors deployed at the minimum needed density over a network crowded with cheap devices but it can find the optimum point in

A. Scaglione (✉)
Electrical and Computer Engineering, UC Davis, Davis, CA, USA
e-mail: ascaglione@ucdavis.edu

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_4,

terms of energy between these two extremes. The denser network is actually more versatile and therefore is the solution that we contend is the best.

1 The Role of Correlated Information in Sensor Systems

Consider a network of N sensors, denoted by $S \in \{1, 2, \ldots, N\}$, and a random vector $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ that represents the sensors' data, i.e., X_i is the data at sensor i. We consider $\mathbf{X}$ to be a discrete vector of digitized measurements collected at the nodes.

Both sensing $\mathbf{X}$ as well as the process of gathering these data from remote locations contribute, albeit in different ways, in creating correlated information. Dense networks are swelling with correlated information and, therefore, considering ways of using it is a worthy task.

In fact, the measurements $\mathbf{X}$ are often intrinsically correlated (at least locally), because the nodes sense the same underlying phenomenon whose number of degrees of freedom is in most cases limited, so that $P(\mathbf{x}) \neq \prod_{i=1}^{N} P(x_i)$.

This type of correlation does not translate in obvious ways of encoding cooperatively, except for some special cases; nevertheless, it always expands the capacity attainable over a multiple-access channel [5]. Let Z be the output of the MAC channel, with as input the vector $\mathbf{Y} = [Y_1, Y_2, \ldots, Y_N]$. In a standard MAC channel with independent inputs, given any subset of the indexes $S \in \{1, \ldots, N\}$, and denoting by $\bar{S}$ its complement, and by $\mathbf{X}_S$ the vector of random variables with indexes in S, Ahlswede [1] and Liao [10] determined the following limits:

$$\sum_{i \in S} H(X_i) < I(Z; \mathbf{Y}_S | \mathbf{Y}_{\bar{S}}). \tag{1}$$

Cover, El Gamal and Salehi proved that, when the sources are dependent, choosing channel codes that depend on the source variables can widen the capacity region. To specify the region a caveat is that some random vectors $\mathbf{X}$ are such that there exist functions extracting a *common part* $\mathbf{W}_S = f_S(\mathbf{X}_S) = g_S(\mathbf{X}_{\bar{S}})$ in the sense introduced by [6]. One can easily recognize that this would be the case if $\mathbf{X}$ were data received over a broadcast channel transmitting a message $\mathbf{W}$. For correlated data $\mathbf{X}$ capacity region changes in two ways compared to (1): by lowering the left side of the inequality and by increasing the boundary on the right side. Reliable transmission is possible if there exist probability mass functions: $P(\mathbf{U}_S,)P(\mathbf{Y}_S | \mathbf{U}_S, \mathbf{X}_S)P(\mathbf{Y}_{\bar{S}} | \mathbf{U}_S, \mathbf{X}_{\bar{S}})$ such that:

$$H(\mathbf{X}_S | \mathbf{X}_{\bar{S}}) < I(Z; \mathbf{Y}_S | \mathbf{Y}_{\bar{S}}, \mathbf{X}_{\bar{S}}, \mathbf{U}_S) \tag{2}$$

$$H(\mathbf{X} | \mathbf{W}_S) < I(Z; \mathbf{Y} | \mathbf{W}_S, \mathbf{U}_S) \tag{3}$$

$$H(\mathbf{X}) < I(Z; \mathbf{Y}), \tag{4}$$

where $\mathbf{U}_S$ is a suitable auxiliary random variable.

The authors in [5] noted that the reduction of the left side could be attained in a decentralized setting using Slepian and Wolf Data Compression [15], but the expansion of the boundaries is a different effect, indicating that correlation can be tied not only to greater compression ability but also to a wider capacity region that can be attributed to *cooperative coding*. The last of the inequalities, in fact, is the *cooperative capacity* and is the only constraint that stands if the messages coincide exactly and, hence, they are identical to their common part.

The cooperative capacity limit is also very relevant to the problem at hand. This type of correlated information is generating during the data gathering process itself, due to the broadcast nature of the wireless medium, because nodes that are inactive and overhear other sensors' transmissions are candidates to cooperatively relay the information that they overhear. Their ability to decode some elements of $\mathbf{Y}$ is a sufficient condition for granting them the ability to cooperate, with a number of methods that are classified as *decode and forward methods*. When the information is decoded and it coincides to the useful $\mathbf{X}$, the relay nodes are bounded by the cooperative bound $H(\mathbf{X}) < I(Z;\mathbf{Y})$. As a matter of fact, in perfect agreement with (2), other alternative exist when these random variables do not have access to common information.

In some cases, not coding is preferable [7] and, not surprisingly the *amplify and forward* method is one example of cooperation that does not rely on the ability of extracting common information.

The result in [5] points out the sub-optimality of the separation between source and channel coding when multiplexing correlated data. The question is if it would take much more complex radio designs than the ones that we currently use to exploit the untapped resource of *correlated information*. An indicator that complex encoding is not what is necessarily needed is given by the work of [7].

1.1 Feedback and Correlation

Feedback is also omnipresent in wireless networks. Using a modification of the Kailath-Schalkwijk scheme Ozarow [11] characterized the capacity of a two-user additive white Gaussian noise with noise variance N_0 and MAC, with users powers bounded by P_1 and P_2. He proved that that the capacity is the following convex hull:

$$\bigcup_{0\le\rho\le 1}\left((R_1,R_2): 0\le R_1\le \frac{1}{2}\log\left(1+\frac{P_1(1-\rho^2)}{N_0}\right),\right.$$
$$0\le R_2\le \frac{1}{2}\log\left(1+\frac{P_2(1-\rho^2)}{N_0}\right);$$
$$\left.0\le R_1+R_2\le \log\left(1+1+\frac{P_1+P_2+2\sqrt{P_1P_2}\rho}{N_0}\right)\right).$$

This region is, again, larger than the limit provided by Ahlswede [1] and Liao [10]. In the scheme Ozarow proposed, one can see that feedback expands the

capacity of the MAC channel via a cooperative effect, that is made possible by the fact that the two sources share now correlated information.

2 Sensor Data Model

In addition to not wanting a complex architecture, one would want to save the sensor batteries to forward information about their own measurements, dedicating a small, possibly vanishing, fraction of energy per node to forward information around. Also, if the sensors are used as relays, the adoption of complex reception strategy to manage interference would increase the cost per sensor and, hence, cooperative strategies need to be simple also at the receiver side. Non cooperative multi-hop networks can trade-off density with relay power, but congestion would inevitably arise as the network scales up [8] not to mention the cost of computing routes towards specific sinks.

What we discuss next is a very simple model to rip the benefits of correlation-induced cooperative coding in dense networks. To have cooperation of correlated sources what it often lost is the modularity of traditional source and channel coding, because the encoding technique depends on the structure of the data. In order to quantify possible benefits of correlated data and determine an appropriate encoding strategy, here we specify the model for the sensor data that we are going to assume throughout the chapter.

We considers sensors $\mathcal{N} = \{1, \ldots, N\}$ that are in a fixed deployment at locations, denoted by v_i, in the set $\mathcal{V} = \{v_1, \ldots, v_N\}$; we assume that they are time synchronized and that are aware of their location v_i (the location identifies the sensor as well so v_i and the index of the sensor i will be used interchangeably). For the sake of simplicity we consider a one-dimensional deployment with equidistant sensors over a range $[0, D]$, spaced by $d = D/N$, so that $v_i = (i-1)d$.

We model $\mathbf{X}$ as a first-order shift-invariant Markov sequence with $X_i \in \{0, 1\}$, for all i. That is, the probability of $\mathbf{X} = \mathbf{x}$, for $\mathbf{x} = [x_1, \ldots, x_N]$, can be expressed as

$$P(\mathbf{X} = \mathbf{x}) = P(X_1 = x_1) \prod_{i=2}^{N} P(X_i = x_i | X_{i-1} = x_{i-1}).$$

We denote by $\alpha = P(X_i = 1 | X_{i-1} = 0)$ and $\beta = P(X_i = 0 | X_{i-1} = 1)$ the transition probabilities (see in Fig. 1(a)). Since the Markov chain is shift invariant, or spatially homogeneous, it is also *reversible* i.e., $\Pr(X_{i-1} = a | X_i = b) = \Pr(X_i = a | X_{i-1} = b)$ for all $a, b \in \{0, 1\}$. The stationary distribution of the Markov chain has state probabilities:

$$p := P(X_i = 1) = \frac{\alpha}{\alpha + \beta} \tag{5}$$

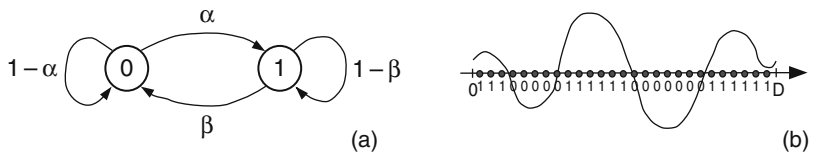


Fig. 1 The binary Markov data model for the sensors observations **X**. (**a**) The two state Markov chain. (**b**) The line network

and

$$q := P(X_i = 0) = 1 - p = \frac{\beta}{\alpha + \beta}. \tag{6}$$

The correlation coefficient between adjacent nodes is

$$\rho := \frac{\mathrm{Cov}(X_i, X_{i+1})}{\sigma_{X_i}\sigma_{X_{i+1}}} = 1 - (\alpha + \beta) = 1 - \bar{\rho} \tag{7}$$

where $\sigma^2_{X_i} = \mathbf{E}[X_i - \mathbf{E}(X_i)]^2 = p(1-p)$ is the variance of X_i and $\mathrm{Cov}(X_i, X_{i+1}) = p(1-p)[1-(\alpha+\beta)]$ is the covariance between X_i and X_{i+1}. The pair (p, ρ) and the pair of transition probabilities (α, β) are one to one.

Let $H_b(p)$ be the entropy of a Bernoulli random variable with success probability p. The aggregate rate of the entire network is:

$$H(\mathbf{X}) = H_b(p) + (N-1)H(X_i|X_{i-1}) \tag{8}$$

The first order Markov model may be a first order approximation capturing the statistics of the data observed by a line of sensors that quantize with 1-bit a continuous random field that is sampled at each location, e.g., Fig. 1(b). When the number of sensors increases in the fixed interval $[0, D]$ in most cases of practical interest the transition probabilities will eventually decrease and, therefore, asymptotically $H(X_i|X_{i-1}) \to 0$ as the density scales up.

In particular, using the symbol Ω to indicate Landau notation, the following lemma holds [12]:

Lemma 1 *(1) For fixed* (p, ρ) $H(X_i|X_{i-1})$ *is constant, and we have*

$$H(\mathbf{X}) = \Omega(N); \tag{9}$$

(2) for $\bar{\rho} := 1 - \rho \le c'/N$, *for some* $c' > 0$, *and a fixed value of p, we have*

$$H(\mathbf{X}) = \Omega(\log(N)). \tag{10}$$

Hence, our objective in the following is to show how one can utilize a channel with finite aggregate throughput of one bit per symbol to convey **X** using an amount

of communication resources that scales with N as exactly the same rate of $H(\mathbf{X})$, as predicted it is possible from (2).

3 A Cooperative Broadcast Mechanism for Network Feedback

We assume that the sensors' radios are equipped with an ON-OFF keying ($Y_i = 0/1$) transmitter and an energy detector as receiver, which quantizes the output with one bit. The time is divided in frames of duration T_f that are slotted in P intervals. The initial slot of each frame is dedicated to transmission of data from the sensors' source-encoders and it is followed by $P-1$ slots for cooperative relay. Let $\mathcal{G}[n] \subset \mathcal{N}$ be a group of sensors that is scheduled to transmit their state value during frame n. The source encoder of this group of nodes outputs a node state $b \in \{1, 0\}$, equivalent to the Boolean true or false.

We let the ON symbol, which unlike the OFF symbol costs transmission power, be associated with false $a = 0$, i.e., $Y_i[n] = \bar{a}$ for $i \in \mathcal{G}[n]$. Next we shall omit the frame index form the discussion since it is not useful to clarify the relay mechanism.

The set of nodes active is therefore partitioned in two sets $\mathcal{G}^{(0)} \cup \mathcal{G}^{(1)} = \mathcal{G}$ and nodes that have state $a = 0$ will emit a pulse at unison, while $\mathcal{G}^{(1)}$ will be silent, with the rest of the nodes in $\mathcal{N}/\mathcal{G}$ that are not scheduled to transmit. Denoting by $h_{ij}(t)$ the channel between transmitter i and receiver j, nodes that are not transmitting will test two hypotheses on the received signal ($n_i(t)$ is additive white Gaussian noise):

$$\begin{aligned} H_0 &: r_i(t) = n_i(t) \\ H_1 &: r_i(t) = \sum_{j \in \mathcal{G}^{(0)}} h_{ij}(t) * p(t) + n_i(t) \quad \text{for any } \mathcal{G}^{(0)} \neq \{\emptyset\}. \end{aligned} \tag{11}$$

We assume that the receiver will measure the energy over a time slot designated for the transmission and compare it with a threshold that is set to have a prescribed false alarm probability. Let τ be the energy threshold used to discriminate between the two hypotheses H_0/H_1, and assume that a false alarm occurs at any node independently and with negligible probability.

The communication architecture has the objective to make every node aware that $\mathcal{G}^{(0)} \neq \{\emptyset\}$. Since in a large deployment only few nodes will be able to reliably detect that $\mathcal{G}^{(0)} \neq \{\emptyset\}$, cooperation is needed to render this mechanism robust.

We call the nodes in $\mathcal{G}^{(0)}$ the first level $\mathcal{L}_0$. If $\mathcal{G}^{(0)} \neq \{\emptyset\}$ with some non zero probability groups of nodes will receive sufficient energy in their signal to cross the threshold τ and choose hypothesis H_1. We call this group level $\mathcal{L}_1$. The simple cooperative policy we set is that nodes in $\mathcal{L}_1$ send a pulse in the following slot. The procedure continues, with every group of nodes whose test is positive for H_1 transmit a pulse in the following slot only once. Every group forms a new level of cooperative transmitters $\mathcal{L}_2, \mathcal{L}_3 \ldots, \mathcal{L}_{P-1}$ and so on, for the predetermined number of slots $P-1$ designed to reach the entire network via cooperation.

This communication mechanism is analogous to what has been called *Opportunistic Large Array* (OLA) model [13]. In [14] it was proven that through the OLA, a message that originates from a particular node can flood the entire network with a bounded delay, with a constant relay power density.

In particular, assuming a deterministic path loss model $|h_{ij}(t)|^2 \propto (1 + d_{ij})^2$ one can prove the following [9]:

Lemma 2 *Let $P_s > \tau$ be the power of a source in $\mathcal{G}^{(0)} \equiv \mathcal{L}_0$. Let the power of each relay P_r be such that the aggregate relay power over the network $P_r N$ is finite, such that the power density $P_r/(ND) = \mu$ is constant, no matter how large N is. For a fixed D, and path-gain $|h_{ij}(t)|^2 \propto (1 + d_{ij})^2$ in the limit for $N \to \infty$ it is sufficient to have node density $\mu > (3 + 2\sqrt{2})\tau$ to have probability of detection growing to one over the entire network within a finite number of levels.*

While with deterministic path loss the delay of reception is deterministic, in [14] it was shown that in random fading a similar scenario applies, and propagation can even be faster, although the penalty is that nodes can be in outage. The outage probability can be made arbitrarily small by increasing the power density, the source power or by increasing the number of levels of cooperative nodes $P - 1$. The next section sets the stage for introducing our channel codes, by modeling the OLA channel we described as a noiseless binary channel that computes the logic OR of all the Y_i for $i \in \mathcal{G}[n]$.

3.1 The OR Broadcast Channel

Given the simple cooperative mechanism presented, at the completion of each frame each node will have determined reliably if H_1 is true. The output of the channel in that frame from the designated group $\mathcal{G}[n]$ towards every sensor location v_i can be modeled as a noiseless binary output $Z_i[n]$, which computes the logic OR of the sources that are active in the first level, i.e., $j \in \mathcal{G}^{(0)}[n], \to Y_j[n] = 1$:

$$Z_i[n] = \bigcup_{j \in \mathcal{G}^{(0)}[n]} Y_j[n].$$

Note that the broadcast mechanism is naturally full-duplex, because nodes that transmit automatically know that $\mathcal{G}^{(0)}[n] \neq \{\emptyset\}$, so $\forall j \in \mathcal{G}^{(0)}[n] \to Z_j[n] = 1$ and everybody else in the network can listen to the OLA signal.

The capacity of independent sources over this OR broadcast channel would be vanishingly small for a large network in which each sensor observes a binary sample X_i with generally a marginal entropy $H(X_i)$ that can be close to 1, no matter how correlated are the data and how much smaller is $H(\mathbf{X})$ compared to $\sum_{i=1}^{N} H(X_i) = NH(X_1)$. Because the aggregate channel throughput $\forall j \; I(Z_j; \mathbf{Y})$ is one bit per frame, with independent encoders the time to communicate one sample per node will have to increase linearly with the number of nodes N in a number of frames $\geq NH(X_1)$.

We will discuss next a cooperative source encoder strategy introduced in [12] that can broadcast the aggregate data $\mathbf{X}$ in a number of frames that scales like $NH(X_i|X_{i-1})$.

4 Channel Coding via Query-and-Response Strategies

The structure of our cooperation-inducing encoder is shown in Fig. 2. The key idea is that each network symbol $Z[n]$ is the answer to a query $Q[n]$ formulated based on the previous symbols $Z[n-1]$, $Z[n-2]$, ..., $Z[1]$; these symbols are available at all nodes, as we explained in the previous section. The objective of the encoder design is to minimize the communication resources by minimizing the number of queries needed to obtain a lossless reconstruction of the source codeword $\mathbf{X}$. The type of queries we investigate are searched over so-called *group testing* strategies [2, 4]: each query $Q[n]$ tests the same question over a specific subgroup of nodes $\mathcal{G}$. Because there are 2^N groups that can be chosen, we restrict our attention to groups of continuous nodes, reducing the problem of optimizing the query to a linear search.

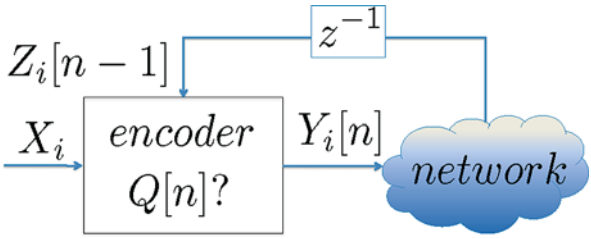


Fig. 2 The query and response cooperative encoder

Denoting by L the total number of queries needed to reconstruct the data $\mathbf{X}$ losslessly. The queries $Q[1]$, $Q[2]$, ..., $Q[L]$ evolve based on the corresponding responses $\mathbf{Z}[1:L] = [Z[1], Z[2], \ldots, Z[L]]$, until $\mathbf{Z}[1:L]$ and $\mathbf{X}$ are one to one. The first query $Q[1]$ is based on the available knowledge on the prior distribution $P(\mathbf{X})$. As more data come in, the n-th query $Q[n]$ is formulated based on the conditional distribution $P(\mathbf{X}|\mathbf{Z}[1:m-1])$. L is the total number of queries needed to reconstruct $\mathbf{X}$ iff $P(\mathbf{X} = \mathbf{x}^*|\mathbf{Z}[1:L]) = 1$ for the $\mathbf{x}^*$ that corresponds the realized value of $\mathbf{X}$.

Clearly, the following bounds hold:

$$H(\mathbf{X}) \leq \mathbf{E}[L] \leq N. \tag{12}$$

We consider two suboptimal methods inspired by group testing algorithms [2, 4]: (1) *the optimized recursive algorithm*, where the optimal set of sensors to be queried in each time slot is selected by solving a set of recursive equations and (2) *the tree*

splitting algorithm, where the group of sensors are chosen through a series of binary partitions of the network. The first provides very tight bounds for finite N but can only be computed numerically, the second one provides close forms bounds that are looser, but amenable to interpretation.

5 Optimized Recursive Group Testing Algorithm

Let $L_{\text{rec}}(\mathbf{X}_{i+1}^{i+m})$ be the minimum number of queries needed to resolve the data $\mathbf{X}_{i+1}^{i+m} = [X_{i+1}, X_{i+2}, \ldots, X_{i+m}]$ using the optimized recursive strategy and let:

$$G_a(m) := \mathbf{E}[L_{\text{rec}}(\mathbf{X}_{i+1}^{i+m}) | X_i = a], \tag{13}$$

indicate the expected number of queries needed to gather data $X_{i+1}, \ldots, X_{i+m}$ given $X_i = a$. If we know from the initial query that $X_1 = a$, where $a \in \{0, 1\}$, then the expected number of queries needed to resolve $\mathbf{X}_2^N$ is $\mathbf{E}[L_{\text{rec}}(\mathbf{X}_2^N) | X_1 = a] \equiv G_a(N-1)$. Since $X_1 = 1$ with probability p and $X_1 = 0$ with probability $1-p$, the expected number of queries under the optimized recursive scheme is

$$\begin{aligned}\mathbf{E}[L_{\text{rec}}(\mathbf{X})] &= 1 + p \cdot \mathbf{E}[L_{\text{rec}}(\mathbf{X}_2^N) | X_1 = 1] + (1-p) \cdot \mathbf{E}[L_{\text{rec}}(\mathbf{X}_2^N) | X_1 = 0] \\ &= 1 + p \cdot G_1(N-1) + (1-p) \cdot G_0(N-1). \end{aligned} \tag{14}$$

Notice that $G_a(m)$ is invariant with respect to the index i because of the spatial homogeneity of the Markov Chain, i.e., $\mathbf{E}[L_{\text{rec}}(\mathbf{X}_2^{m+1}) | X_1 = a] = \mathbf{E}[L_{\text{rec}}(\mathbf{X}_{i+1}^{i+m}) | X_i = a]$, for all positive integers i.

Clearly, to solve for $\mathbf{E}[L_{\text{rec}}(\mathbf{X})]$ in (14), we must determine $G_a(N-1)$ which can be done recursively [12]. In fact, after acquiring the knowledge that $X_1 = a$ from the initial query, suppose that we go on to choose the next n sensors, with $i = 2$ up to $n+1$, and to ask the question b. If all n sensors contain the bit b (i.e. $X_2 = b, \ldots, X_{n+1} = b$), no sensor will respond to the query and the data $X_2, \ldots, X_{n+1}$ are resolved. The expected number of queries needed to resolve the remaining sensors' data becomes $\mathbf{E}[L_{\text{rec}}(\mathbf{X}_{n+2}^N) | X_{n+1} = b] \equiv G_b(N-n-1)$, which follows from the Markov property. On the other hand, if there exists a sensor that does not contain b (but contains $\bar{b}$ instead, where $\bar{b}$ is the complement of b), the sensor will transmit, indicating that the guess is false, informing that there exists $j \in \{2, \ldots, n+1\}$ such that (s.t.) $X_j = \bar{b}$. Let us denote, in general, by $F_a(m, n, b)$, for $n \leq m$, the expected number of queries needed to gather the data $X_{i+1}, \ldots, X_{i+m}$ given that $X_i = a$ and that there exists $j \in \{i+1, \ldots, i+n\}$ such that $X_j = \bar{b}$:

$$F_a(m, n, b) := \mathbf{E}[L_{\text{rec}}(\mathbf{X}_{i+1}^{i+m}) | X_i = a, \exists j \in \{i+1, \ldots, i+n\} \text{ s.t. } X_j = \bar{b}]. \tag{15}$$

With (15), we can also express more generally $G_a(m)$ as

$$G_a(m) = 1 + \min_{1\le n\le m, b\in\{a,\bar{a}\}} \Big\{ P(\mathcal{E}_b^{(n)}|X_i = a) G_b(m-n) + \Big[1 - P(\mathcal{E}_b^{(n)}|X_i = a)\Big] F_a(m,n,b) \Big\}, \tag{16}$$

where $\mathcal{E}_b^{(n)} := \{X_{i+1} = b, \ldots, X_{i+n} = b\}$ and where the number of sensors n and the question b is chosen to minimize the number of queries needed to resolve the remaining sensors' data. To solve for $F_a(m,n,b)$ one can introducing the function:

$$\begin{aligned} J_a(m,n,b,\ell) :=& \mathbf{E}[L_{\text{rec}}(\mathbf{X}_{i+1}^{i+m})|X_i = a, \exists j \in \{i+1,\ldots,i+n\} \\ & \text{s.t. } X_j = \bar{b}, \exists k \in \{i+1,\ldots,i+\ell\} \text{ s.t. } X_k = b], \end{aligned} \tag{17}$$

and determine that the following set of coupled difference equation hold true:

$$F_a(m,n,b) = 1 + \min_{1\le \ell\le m, c\in\{b,\bar{b}\}} \left\{ \begin{array}{ll} \alpha F_b(m-\ell, n-\ell, b) + [1-\alpha] F_a(m,\ell,b), & c = b,\ 1 \le \ell < n \\ \alpha G_{\bar{b}}(m-\ell) + [1-\alpha] J_a(m,n,b,\ell), & c = \bar{b},\ 1 \le \ell \le m \end{array} \right\} \tag{18}$$

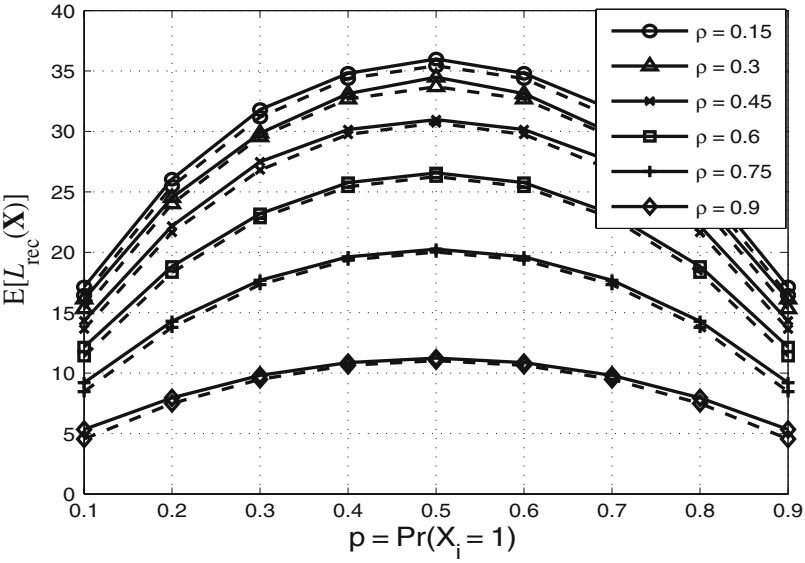


Fig. 3 For $N = 36$ and $\rho = 0.15$–0.9, we show the performance of the optimized recursive algorithm (*solid line*) and the entropy lower bound of (12) (*dashed line*)

$$J_a(m,n,b,\ell) = 1 + \min_{1\le z<n,\, d\in\{b,\bar{b}\}} \left\{ \begin{array}{l} \beta F_b(m-z, n-z, b) + [1-\beta]\, J_a(m,z,b,\ell),\ d=b,\ 1\le z<n, \\ \beta F_{\bar{b}}(m-z, \ell-z, \bar{b}) + [1-\beta]\, J_a(m,z,\bar{b},n),\ d=\bar{b},\ 1\le z<\ell \end{array} \right\} \tag{19}$$

with $\alpha := P(\mathcal{E}_b^{(\ell)} | \mathcal{E}_{1ab}^{(n)})$ and $\beta := P(\mathcal{E}_{\bar{b}}^{(z)} | \mathcal{E}_{2ab}^{(n,\ell)})$. These equations can be solved recursively. The solution is remarkably tight to the Huffman coding bound, as shown in Fig. 3.

We observe that the proposed querying strategy closely approximates the optimal performance (i.e., the entropy lower bound that can be achieved asymptotically with Huffman coding). More importantly, the expected number of channel accesses varies with the entropy of the data as opposed to consuming a fixed number of channel accesses that are proportional to the number of sensors. The advantage of the approach is most visible when p is close to 0 or 1 and when ρ is close to 1, i.e., the cases where sensors' data $\mathbf{X}$ have low aggregate entropy.

6 Binary Tree Splitting Algorithm

To attain an simple asymptotic trend, we further simplify our querying scheme by splitting the set of nodes queried in two groups and querying each group twice over consecutive frames. This method is analogous to the binary tree splitting algorithm used in the context of collision resolution in [4]. More specifically, suppose that $\mathcal{G}[n]$, $n \ge 1$, is the n-th group that is chosen. In this case, we impose the two queries $Q[2n-1] = (\mathcal{G}[n], 0)$ and $Q[2n] = (\mathcal{G}[n], 1)$ on the same group in consecutive time slots. This approach yields a pair of outputs $(Z[2n-1], Z[2n])$ and provides us with the ternary information on $\mathcal{G}[n]$, namely,

0: $(Z[2n-1], Z[2n]) = (0,1)$
1: $(Z[2n-1], Z[2n]) = (1,0)$
e: $(Z[2n-1], Z[2n]) = (1,1)$

where **0** indicates that all sensors in $\mathcal{G}[n]$ contain the bit 0 and, similarly, **1** indicates that all sensors in $\mathcal{G}[n]$ contain the bit 1 and **e**, called *erasure*, indicate that the group values are not uniform. When **e** is received, the group is partitioned into two subgroups of equal size as specified by the binary tree splitting algorithm. In both the first two cases, instead, the query terminates the tree, since the values of the sensor group tested are resolved.

For example, we consider a network of 16 nodes that are partitioned into a binary tree, as shown in Fig. 4. For a network of $N = 2^M$ nodes, there are $M+1$ levels in the binary tree, i.e., levels $0, 1, \ldots, M$. In the i-th level of the tree, the network is partitioned into 2^i groups of size equal to 2^{M-i}. The group $\mathcal{G}_{ij}$ denotes the j-th group in the i-th level of the tree (where $j \in \{0, 1, \ldots, 2^i - 1\}$), which consists of all

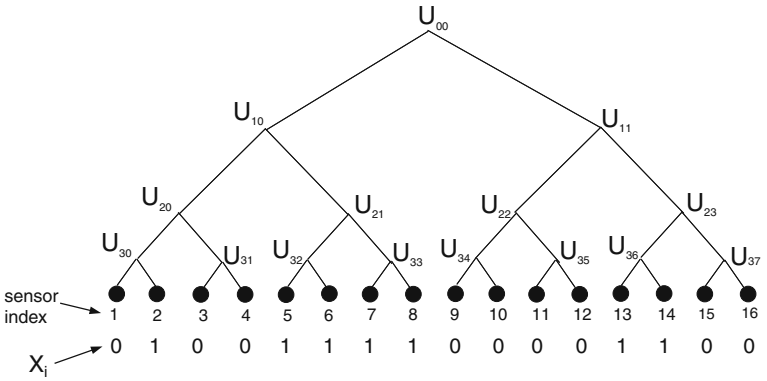


Fig. 4 Example of the realization of a sensor field with the binary sequence 0100111100001100. The sequence of queries are done in the order of the following groups: $\mathcal{G}[1] = \mathcal{G}_{10}$, $\mathcal{G}[2] = \mathcal{G}_{20}$, $\mathcal{G}[3] = \mathcal{G}_{30}$, $\mathcal{G}[4] = \mathcal{G}_{40}$, $\mathcal{G}[5] = \mathcal{G}_{41}$, $\mathcal{G}[6] = \mathcal{G}_{31}$, $\mathcal{G}[7] = \mathcal{G}_{21}$, $\mathcal{G}[8] = \mathcal{G}_{11}$, $\mathcal{G}[9] = \mathcal{G}_{22}$, $\mathcal{G}[10] = \mathcal{G}_{23}$, $\mathcal{G}[11] = \mathcal{G}_{36}$ and $\mathcal{G}[12] = \mathcal{G}_{37}$

sensors within its subtree, i.e., $\mathcal{G}_{ij} = \{j2^{M-i} + 1, j2^{M-i} + 2, \ldots, (j+1)2^{M-i}\}$. In the binary tree splitting algorithm, the sequence of queries starts from the subgroups of $\mathcal{G}_{00}$, i.e., $\mathcal{G}_{10}$ and $\mathcal{G}_{11}$, and continues splitting and querying the smaller subgroups each time the larger group results in **e**. If the query on $\mathcal{G}_{ij}$, for $j < 2^i$, results in either **0** or **1**, the process will go on to query the smallest group that is not yet resolved. If the test results in an erasure, the vertex $\mathcal{G}_{ij}$ branches into two subgroups where the group $\mathcal{G}_{i+1,2j}$ is queried next.

What is important is that the average performance can be expressed analytically, as follows:

Theorem 1 *Consider a network of $N = 2^M$ sensors and the binary observations* $\mathbf{X}$ *modeled by the two-state Markov Chain with the parameters (p, ρ). The expected number of queries needed to retrieve the data* $\mathbf{X}$ *with the binary tree splitting algorithm is given by*

$$\mathbf{E}[L_{\text{tree}}(\mathbf{X})] = 4 + \sum_{i=1}^{M-1} 2^{i+2} \psi(M - i, p, \rho). \tag{20}$$

where

$$\psi(M - i, p, \rho) = 1 - p(1 - q\bar{\rho})^{2^{M-i}-1} - q(1 - p\bar{\rho})^{2^{M-i}-1}$$

and $\bar{\rho} := 1 - \rho$.

Furthermore, using the result above, in [12] we have proven that:

Theorem 2 *Let* $\mathbf{E}[L_{\text{opt}}(\mathbf{X})]$ *be the expected number of queries needed by the optimal Q&R strategy. (1) For fixed (p, ρ),*

$$\mathbf{E}[L_{\text{opt}}(\mathbf{X})] = O(N) = O(H(\mathbf{X})); \tag{21}$$

(2) for fixed p *and* $\bar{\rho} \leq c'/N$ *for some* $c' > 0$*,*

$$\mathbf{E}[L_{\text{opt}}(\mathbf{X})] = O(\log(N)) = O(H(\mathbf{X})). \tag{22}$$

Note that the result is obtained employing the OR broadcast channel, which can at most convey one bit per frame to the entire network.

Overall this attests that our cooperative Q&R is not limited by the same broadcast rate upper-bound $I(Z_i; \mathbf{Y}) = 1$ that follows from our specific sensor access architecture. The interesting fact is that we experience a capacity expansion that follows gracefully the trend of the source demand, as predicted by [5].

7 Conclusions

One of the key messages of [5] was that the concept of capacity becomes elastic in the presence of sources that carry correlated information. The take-away message is that it is possible to stretch the capacity region but that this requires using correlated channel codes formed using the underlying correlation structure of the data.

Gastpar's work noted a particularly dramatic effect of this fact by considering the classic Gaussian CEO problem introduced in [3] over an AWGN MAC channel. The best scaling law in means squared error performance obtained via optimal distributed source coding and channel coding were shown to be far worse than those for uncoded transmission. The question that remains to address is what is the right approach to rip off the benefits of correlation that is known to exist, in a systematic way and possibly in a way that is universal. The idea of learning the structure of the data via queries is what leads to experience a dramatic expansion in capacity in the example we discussed in detail above. This idea is appealing because it naturally suggests that one could learn the sources and increase the rate at which the sources are gathered progressively. But once again it is unlikely that such benefits can come so close to the compression limits in general, as they did for the Markovian model we considered. Furthermore, it is unclear and complex to find optimal channel codes. Seeing through the fog that surrounds this problem is worthwhile because correlated information it is one of the remaining resource to tap into, to overcome the capacity limitations of large scale wireless networks.

References

1. R. Ahlswede.: Multi-way communication channels, in Proceedings of 2nd International Symposium Informatics Theory, Tsahkadsor, Armenian S.S.R., 1971; Hungarian Academy of Sciences, pp. 23–52, 1973.
2. T. Berger, N. Mehravari, D. Towsley, and J. Wolf.: Random multiple-access communication and group testing, IEEE Transactions on Communications, Vol. 32, No. 7, pp. 769–779, Jul. 1984.

3. T. Berger, Z. Zhang, and H. Viswanathan.: The CEO problem, IEEE Transactions on Information Theory, Vol. IT-42, pp. 887–902, May 1996.
4. J. Capetanakis.: Generalized TDMA: the multi-accessing tree protocol, IEEE Transactions on Communications, Vol. 27, No. 10, pp. 1476–1484, Oct. 1979.
5. T. Cover, A. El Gamal and M. Salehi.: Multiple access channels with arbitrarily correlated sources, IEEE Transactions on Information Theory, Vol. IT-26, No. 6, pp. 648–657, Nov. 1980.
6. P. Gács and J. Korner.: Common information is far less than mutual information, Problems of Control and Information Theory, Vol. 2, pp. 149–162, 1973.
7. M. Gastpar and M. Vetterli.: Power, spatio-temporal bandwidth, and distortion in large sensor networks, IEEE Journal on Selected Areas in Communications, Vol. 23, No. 4, pp. 745–754, April 2005.
8. P. Gupta and P.R. Kumar.: The capacity of wireless networks, IEEE Transactions on Information Theory, Vol. IT-46, pp. 388–404, Mar. 2000.
9. Y.-W. Hong, A. Scaglione, R. Manohar, and B. Sirkeci-Mergen.: Dense sensor networks that are also energy efficient: when more is less, in Proceedings of IEEE Military Communications Conference (MILCOM), pp. 1–7, Oct. 2005.
10. H. Liao.: A coding theorem for multiple access communications, presented at the International Symposium Informatics Theory, Asilomar, 1972.
11. L.H. Ozarow.: The capacity of the white Gaussian multiple access channel with feedback. IEEE Transactions on Information Theory, Vol. 36, No. 1, pp. 156–161, 1990.
12. Y.-W. Peter Hong and A. Scaglione.: Group testing for binary Markov sources: data-driven group queries for cooperative sensor networks, IEEE Transactions on Information Theory, Vol. 54, No. 8, Aug. 2008.
13. A. Scaglione and Y.-W Hong.: Opportunistic large arrays: cooperative transmission in wireless multihop ad hoc networks to reach far distances, IEEE Transactions on Signal Processing, Vol. 51, No. 8, pp. 2082–2092, Aug. 2003.
14. B. Sirkeci-Mergen, A. Scaglione, and G. Mergen.: Asymptotic analysis of multistage cooperative broadcast in wireless networks, IEEE Transactions on Information Theory, Vol. 52, No. 6, pp. 2531–2550, Jun. 2006.
15. D. Slepian and J.K. Wolf.: Noiseless coding for correlated information sources, IEEE Transactions on Information Theory, Vol. IT-19, pp. 471–480, Jul. 1973.

Multipath Diversity and Robustness for Sensor Networks

Christina Fragouli, Katerina Argyraki, and Lorenzo Keller

Abstract Balancing energy efficiency and reliability is a common underlying goal for most information collection protocols in sensor networks. Multipath diversity has emerged as one of the promising techniques to achieve such a balance. In this chapter, we provide a unified framework for the multipath techniques in the literature and discuss their basic benefits and drawbacks. We also discuss emerging techniques from the area of network coding.

1 Introduction

The goal of a sensor network is to gather information and communicate it to a collecting entity: sensors typically measure a physical quantity (e.g., temperature) and communicate their measurements (or functions of their measurements, e.g., sums or averages) to a common *sink*. For most applications, sensors communicate to the sink over a wireless channel, while the sink is connected to a wired network. In this chapter, we focus on the problem of designing a *collection protocol*, i.e., a protocol that conveys information from the sensors to the sink.

A typical requirement for such protocols is energy efficiency: sensors are typically powered through small, easily depletable batteries; moreover, they are often installed at inaccessible areas (e.g., forests or mountain slopes), making battery changing practically infeasible. Hence, a collection protocol must be "energy efficient," i.e., consume the minimum amount of energy necessary to meet the performance standards of the corresponding application. The main source of energy consumption in a sensor is the radio, i.e., transmitting data and listening for/receiving incoming data (a per-component breakdown of energy consumption for different sensor platforms can be found in [1]). Hence, one approach to designing energy-efficient collection protocols is to minimize the number of transmissions necessary

C. Fragouli (✉)
School of Computer and Communication Sciences, EPFL, Lausanne, Switzerland
e-mail: christina.fragouli@epfl.ch

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_5,

to convey the required information to the sink. Another, complementary approach is to enable sensors to predict for which periods they are unlikely to receive data and turn their radios off during those periods (a collection protocol that uses this approach is described in [2]).

Energy efficiency is at odds with another typical requirement for collection protocols: reliability in the face of channel and node failures. Compared to wired, wireless channels are relatively unreliable, subject to fading and random fluctuations. This is especially so in sensor networks where the environment can change unpredictably and the sensors themselves can fail (e.g., a landslide can destroy part of a sensor network monitoring forest conditions). To achieve reliable collection in the face of such failures, a collection protocol must generate a certain amount of redundant information – which unavoidably increases both the number of transmissions and the amount of time for which sensors must keep their radios on.

In this chapter, we describe existing as well as emerging collection protocols that balance the competing goals of energy efficiency and reliability.[1] We restrict our attention to the simplest scenario, where the goal is to communicate sensor measurements to the sink,[2] while sensors never turn off their radios. The point of the chapter is to give a flavor of the problems encountered in sensor-network design and outline the different classes of solutions – we far from cover all work done in the area. We start by defining a collection protocol and its associated cost in Sect. 2. Section 3 examines the use of single-path protocols, while Sect. 4 focuses on multipath protocols. Section 5 reviews basic ideas in the area of network coding, while Sect. 6 outlines an approach for deploying network coding in sensor networks. Finally, Sect. 7 concludes this chapter.

2 What is a Collection Protocol?

A fundamental characteristic of wireless communication is that transmissions are broadcast and can be overheard by multiple nodes in the transmitter's vicinity, albeit at different signal levels. For example, in Fig. 1, a transmission by node a_{11} is overheard by nodes a_{10}, a_9 and a_8 at different signal levels; we say that a_{10}, a_9 and a_8 are in a_{11}'s *range* or, equivalently, that they are a_{11}'s "neighbors." As a result, there can be multiple paths between each pair of nodes – Fig. 2 depicts all possible paths between nodes in our example network. A collection protocol essentially determines which of these paths to use and how, in order to convey information from the sensors to the sink.

---

[1] Certain applications can have additional/different requirements, e.g., in a sensor network monitoring for atmosphere poisoning, alarms must to be delivered with minimum latency.

[2] We do not consider the scenario where the goal is to communicate *functions* of sensor measurements, like sums or averages, to the sink.

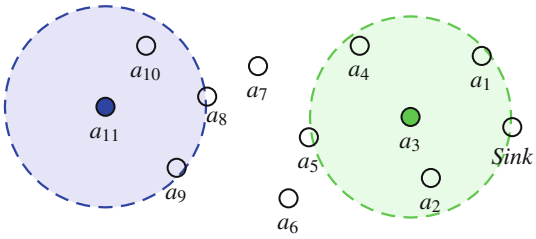


Fig. 1 Wireless transmissions are broadcast. There may be several nodes within the broadcasting range of every sensor node

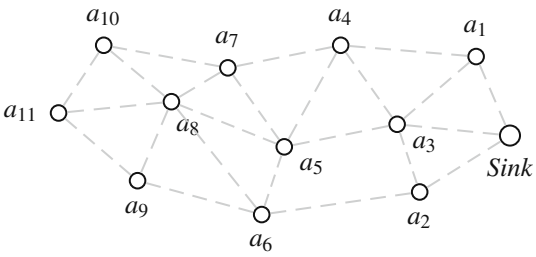


Fig. 2 Representation of the potential connections between neighboring nodes

More specifically, a collection protocol can be decomposed into the following tasks:

1. *Acquisition of channel information*. Each node learns its neighbors and estimates the channel quality between itself and each neighbor.
2. *Topology construction*. Each node builds a routing table that includes a subset of its neighbors; e.g., when the constructed topology is a tree, each node has a single next hop to the sink.
3. *Topology usage*. Each node determines how to route information through the constructed topology – that is, what exactly to send, how often, and to which neighbors.

2.1 Path Cost and Channel Quality

Recall that we are interested in balancing reliability and energy efficiency. As mentioned in Sect. 1, there are two approaches to reducing energy consumption: reduce transmissions and reduce the amount of time for which sensors keep their radios on. Since we are considering a simple scenario where sensors cannot do the latter, in our context, optimizing for energy efficiency means minimizing transmissions. Hence, we define the cost of communication as a function of the number of transmissions required to successfully transfer a certain amount of information between two nodes. More specifically:

Definition 1 We define the cost of communication from a node a to a neighbor b as the average number of transmissions that are necessary to successfully transmit a packet of length L from a to b.

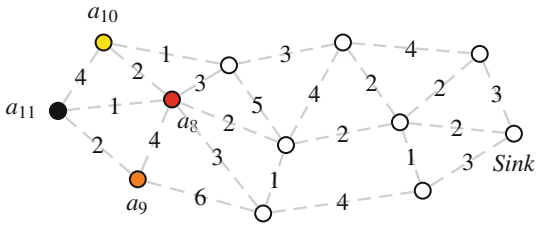


Fig. 3 A graph with vertices corresponding to sensor nodes and edges between any two nodes that are within the transmission range of one another. Each edge has a weight proportional to the number of retransmissions required to reliably transmit information between the nodes it connects

Suppose we know the communication cost between every pair of neighbors in our network at a given point in time. We can depict this information with a *channel-quality graph*, where each vertex represents a sensor and each edge represents a communication channel between two neighbors; each edge has a weight proportional to the corresponding cost. For example, the graph shown in Fig. 3 says that, if node a_{11} broadcasts once, only node a_8 is expected to successfully receive the transmitted packet; if it broadcasts twice, nodes a_8 and a_9 are expected to successfully receive the packet; if it broadcasts 4 times, all neighbors are expected to successfully receive the packet.

Definition 2 Given a channel-quality graph, we define the cost of a path as the sum of the costs of all the edges that make up the path. A *minimum-cost path* from a node a to the sink is a path whose cost is less or equal to the cost of any other path from node a to the sink.

3 Routing on a Tree

A straightforward approach is to connect all sensors to the sink over a tree: (i) Construct a unique, minimum-cost path from each sensor to the sink (such a path is shown in Fig. 4), such that the union of all constructed paths forms a *spanning tree*

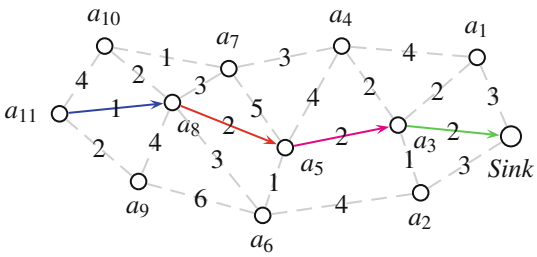


Fig. 4 A minimum-cost path from node a_{11} to the sink. This is also the minimum-cost path for nodes a_8, a_5, a_3 to the sink

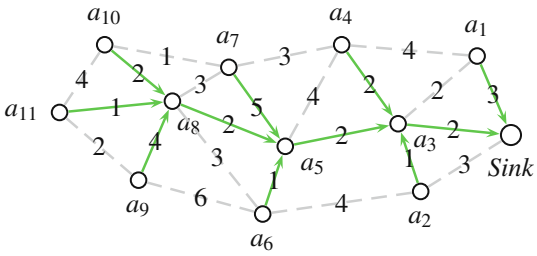


Fig. 5 A tree constructed from the union of the minimum-cost paths from the sensors to the sink

rooted at the sink (such a tree is shown in Fig. 5).[3] (ii) Route each message from sensor a to the sink along the unique path from a to the sink over the tree.

The Collection Tree Protocol (CTP) [3] implements this approach. Task #1 (acquisition of channel information) is accomplished in a setup phase, where each node sends out a sequence of numbered beacons that bear its identity; each node processes the beacons it hears and determines its neighbors; for each neighbor, it counts how many of its beacons it received, estimates the communication cost from that neighbor to itself, and sends this information to the neighbor; beacons are sent with an exponentially increasing interval, which is reset to a small value when certain routing conditions are met. Task #2 (topology construction) is accomplished in a distributed manner over multiple rounds: in each round, each node broadcasts its "distance" to the sink, i.e., the cost of the minimum-cost path from itself to the sink; moreover, each node processes its neighbors' advertisements, identifies the neighbor with the lowest distance to the sink, and chooses it as its "next hop." For example, in Fig. 5, in the first round, nodes a_1, a_2 and a_3 identify the sink as their neighbor; in the second round, nodes a_2, a_4 and a_5 identify node a_3 as their neighbor with the lowest distance to the sink and choose it as their next hop. Finally, task #3 (topology usage) consists of each node transmitting every message it receives to its next hop. For example, in Fig. 5, a message from node a_{11} is routed through nodes a_8, a_5 and a_3.

Routing on a tree faces certain practical challenges:

1. *Changing network conditions*. The tree can break as a result of a node or channel deterioration between two neighbors; moreover, due to channel fluctuations, what used to be an optimal tree may end up using bad-quality paths. To decrease the amount of loss or delay that can result from such events, it is possible to repeat tasks 1 and 2, i.e., monitor channel quality and *adapt* the tree accordingly (CTP follows this approach). However, it is not possible to adapt to all failures: suppose that node a_{11} in Fig. 5 sends a packet to the sink through the constructed

[3] We should clarify that a spanning tree constructed in this manner (i.e., the union of the minimum-cost paths connecting each sensor to the sink) is not necessarily a "minimum-cost spanning tree" as typically defined in graph theory, where the cost of a tree is equal to the sum of the costs of all links that take part in the tree. For example, the tree in Fig. 5, which has cost 25, is not a minimum-cost tree.

tree (nodes a_8, a_5 and a_3); as the packet reaches node a_5, all channels from node a_5 to its neighbors fail; as a result, there is no path connecting a_5 to the sink, and the packet is lost, i.e., a_{11} fails to communicate with the sink, even though there still exists a path from a_{11} to the sink (through nodes a_9, a_6 and a_2). So, it is possible that the tree is unable to adapt fast enough to certain failures, especially in large networks, resulting in temporary disconnection of parts of the network.

2. *Depletion of specific nodes*. Nodes located close to the root of the tree (e.g., node a_3 in Fig. 5) may end up carrying significantly more traffic than the rest, as they participate in multiple paths; as a result, their batteries are depleted significantly faster. Note that these nodes are precisely the ones that need to work in order for the network to remain connected: if node a_3 in Fig. 5 fails, 9 of the remaining 10 nodes are disconnected from the sink.

4 From Tree to Multipath Routing

In the last section, we looked at a collection protocol that constructs a single path from each sensor to the sink. In contrast, multipath collection protocols construct multiple paths from each sensor to the sink, allowing more freedom in route selection. Intuitively, multipath protocols achieve higher reliability than single-path protocols at the cost of higher energy consumption.

4.1 Topology Construction

A node can be connected to the sink over disjoint or overlapping paths; we now discuss these two approaches, as well as some associated computer science problems.

4.1.1 Disjoint Paths

The idea is to construct m disjoint paths from each node to the sink. These paths can be *edge-disjoint* (Fig. 6), providing reliability in the face of channel deterioration; or *vertex-disjoint* (Fig. 7), providing reliability in the face of node failures. Alternatively, a recent proposal suggests that the paths should be "one-hop apart" (Fig. 8), i.e., if a node is used in one of the paths, none of its neighbors should be used in the remaining $m - 1$ paths; the point is to avoid spatially correlated failures [4].

Intuitively, the more constraints we add to path construction, the higher the resulting reliability: Fig. 6 shows two edge-disjoint paths; if node a_8 fails, both paths from a_{11} to the sink are broken. Figure 7 shows two vertex-disjoint paths; if node a_8 fails, a_{11} can still communicate with the sink through the other path. On the other hand, more constraints result in higher-cost paths: given the channel-quality graph of Fig. 3, Fig. 4 shows the minimum-cost path from node a_{11} to the sink. If we want to construct two paths from a_{11} to the sink that are one-hop apart, we

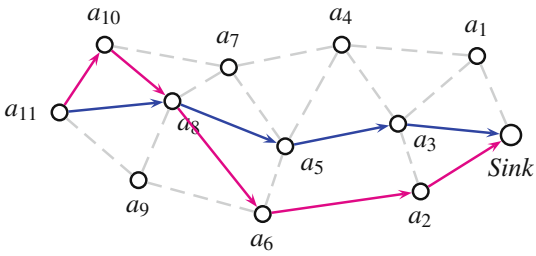


Fig. 6 Two edge-disjoint paths connecting node a_{11} to the sink

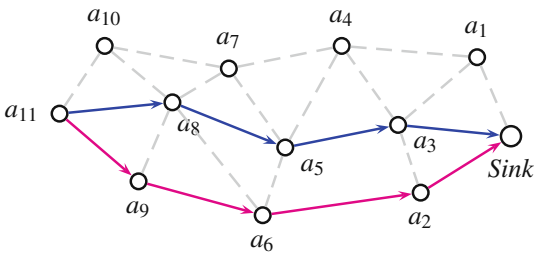


Fig. 7 Two vertex-disjoint paths connecting node a_{11} to the sink

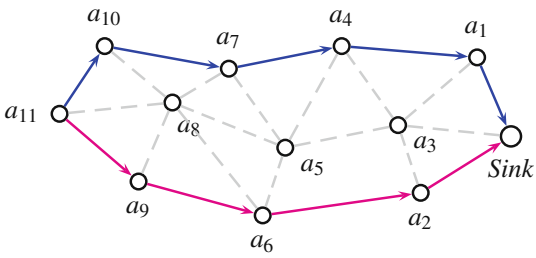


Fig. 8 Two one-hop separated paths connecting node a_{11} to the sink

necessarily have to choose the two paths shown in Fig. 8, none of which is equal to the minimum-cost path. In general, the further away a path is located from the minimum-cost path, the higher its expected cost.

We should note that it is not always feasible to find paths that satisfy such constraints. For example, m edge-disjoint paths exist between two nodes only if the edge min-cut between them is greater or equal to m; similarly, m vertex-disjoint paths only exist if the vertex min-cut between the two nodes is greater than m.

4.1.2 Algorithmic Complexity of Disjoint-Path Construction

Consider a collection protocol that builds two disjoint paths from each sensor to the sink. To minimize the number of transmissions, for each sensor, the protocol must choose the two paths to the sink that have the minimum *overall* cost (i.e., the sum of their costs is less than or equal to the sum of the costs of any other two paths from this sensor to the sink). This problem is known in computer science as the "min-sum 2-path" problem, and it has been shown to be computationally hard to solve (NP-hard). The difficulty comes from the fact that the optimal solution may

involve two paths neither of which is the minimum-cost path, as Example 1 below illustrates.

Example 1 Figure 9 shows two sets of paths that connect node a_{11} to the sink: (i) The first set includes the minimum-cost path (of cost 7) and the second best path (of cost 13). The total cost is 20. (ii) The second set includes two paths, each of cost 9. The total cost is 18, which is the minimum overall cost. This example illustrates that the optimal solution cannot be obtained simply by ranking paths by cost and taking the two best paths.

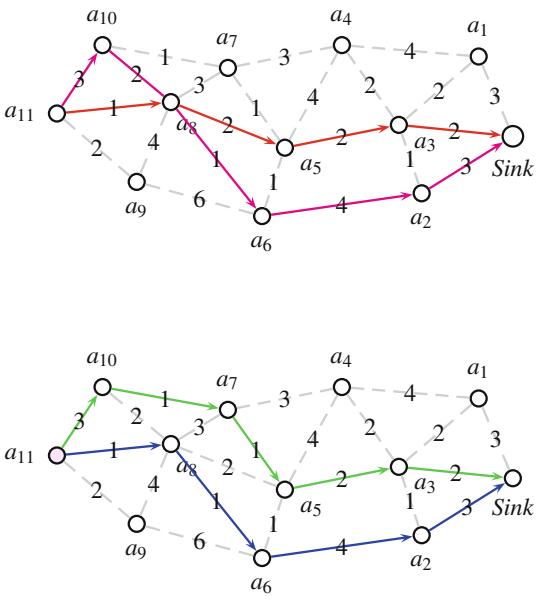


Fig. 9 Two sets of path choices to connect node a_{11} to the sink: the first set includes the minimum-cost path (going through a_1, a_5 and a_3), and the second best path (going through a_{10}, a_8, a_6 and a_2). The second set includes two paths that have the minimum sum cost

So, identifying, for each sensor, two disjoint paths to the sink with the lowest overall cost is hard. An alternative would be to construct *any* two disjoint paths, without looking to minimize the overall cost. We now illustrate that constructing disjoint paths using heuristics can easily lead to configurations where nodes use paths of significantly higher cost than necessary.

Example 2 A computationally reasonable way to select two disjoint paths from each sensor to the sink would be the following:

- Select a minimum-cost path from each sensor to the sink. The union of chosen paths forms a tree T_1.
- Remove all the edges of T_1 from the channel-quality graph.
- Considering only the remaining links, select again a minimum-cost path from every sensor to the sink, which results in a tree T_2.

Figures 10, 11 and 12 illustrate this procedure and show a problematic case, where, in the second tree, node a_3 is connected to the sink through a path of cost 9, whereas,

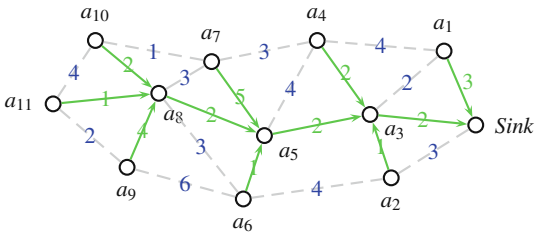


Fig. 10 The first tree T_1 on the cost graph

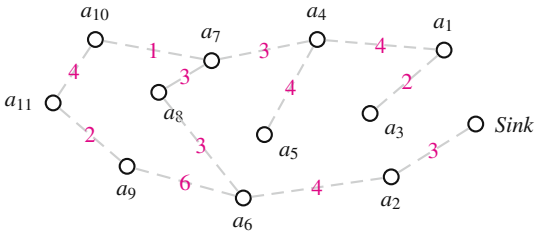


Fig. 11 The remaining cost graph after removing T_1

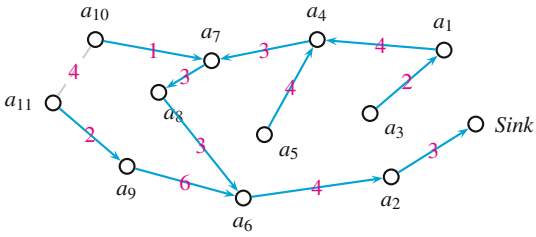


Fig. 12 The second tree T_2. If the paths to be employed are not jointly selected, we may fall into inefficient path constructions

in the original network, there exist edge-disjoint paths of costs 2, 4 and 5, respectively, that connect a_3 to the sink.

4.1.3 Braided Paths

The idea is to first construct a "primary" minimum-cost path from each node to the sink (as described in Sect. 3), then build a "braid" of alternative paths as follows: for each node on the primary path, build a path that does not include that node [5, 6].

Example 3 Consider the braided path choices depicted in Fig. 13. The minimum-cost path from node a_{11} to the sink involves hops through nodes a_8, a_5 or a_3. If node a_8 fails, it can be replaced by the path segment through nodes a_{10} and a_7. Similarly, node a_5 can be replaced through nodes a_6 and a_2, while node a_3 can be replaced through nodes a_4 and a_1.

Intuitively, braided paths are more energy-efficient than disjoint paths, simply because they are more likely to be physically close to the primary (minimum-cost) path. One would think that this higher energy efficiency would come at the cost

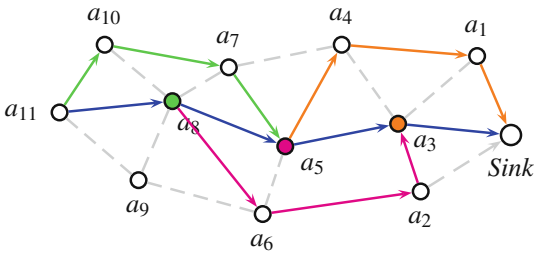


Fig. 13 Braided paths: in the path from node a_{11} to the sink, if any of the nodes a_8, a_5 or a_3 fails, there exists an alternative available path. For example, node a_5 can be replaced by using the path segment through nodes a_6 and a_2

of lower reliability (the reason being that braided paths would be less resilient to multiple-node failures than disjoint paths). Interestingly, early simulation results suggest that this may not be the case: it turns out that, in practical scenarios, braided paths can be as failure-resilient as 2-disjoint and 3-disjoint paths; the intuition is that, in practice, disjoint paths are not physically separated from one another enough to survive multiple-node failures [5].

4.2 Topology Usage

There are several choices on how to use multiple paths from a node to the sink; which one is better depends on the specific application requirements. We group the proposed uses in five broad categories; the first three are relevant for disjoint paths, while the others for braided ones.

4.2.1 Replicate Transmissions

Each source (i.e., each sensor that has a message to convey to the sink) sends its message through all available paths to the sink. This approach is sometimes also described as flooding.

Flooding is typically proposed in conjunction with disjoint paths, where each path carries a replicate of the same message to the sink. Indeed, to flood a braided mesh of paths, we would need the intermediate nodes to suppress incoming packets if a node has more incoming edges than outgoing edges, as is the case for node a_3 in Fig. 13. Similarly, if a node has more outgoing than incoming edges, it would need to create copies of a received packet and implement multicasting.

Flooding favors reliability over energy efficiency: it sends each message over multiple paths, hence, increases the probability of the message reaching the sink in case of node and channel failures; yet, when no such failures occur, flooding uses on the order of m times the number of necessary transmissions to get the message to the sink (assuming m paths are being used). Moreover, when multiple sources send out messages at the same time, flooding can result in congestion, i.e., collisions and queue build-up.

So, flooding is an appropriate choice for applications in which sensors send messages to the sink infrequently and, when they do, reliability is more important than energy efficiency.

4.2.2 Independent Transmissions

In this approach, sources use each path to send a different message to the sink.

This approach favors information rate over reliability: each source can send multiple messages to the sink at the same time; yet, messages sent over paths with node or channel failures may not be delivered. In practice, this approach is not energy-efficient either, in the sense that some messages are sent over more-than-minimum-cost paths. Moreover, similarly to the previous case, when multiple sources send out messages at the same time, it can result in congestion.

Thus, this approach is suitable for applications where sensors send out messages infrequently and, when they do, speed of dissemination is of critical importance. Such an application would be for example, if a sensor wants to send an image of an intruder to the sink; in such a situation, the sensor needs to transmit multiple packets to the sink as fast as possible.

4.2.3 Erasure Coding

Consider a configuration where a source is connected to the sink through m disjoint paths; sending the same message over m paths corresponds to a repetition code of rate $1/m$; sending a different message over each path corresponds to a rate-1 code. Between these two extremes, the source can use an erasure code of rate k/m: the source sends m coded packets; if the sink receives any k out of the m coded packets, it can retrieve the original information.

This approach enables a trade-off between reliability and speed of dissemination. However, similar to the previous case, it is not energy-efficient, as it uses suboptimal paths; moreover, similar to both previous cases, it can cause congestion.

4.2.4 Path-Selective Routing

In this approach, each message is routed along a single path from the corresponding source to the sink. The path is determined dynamically in the following way: each node that receives the message selects the best next-hop to the sink according to some local criterion such as:

- what is currently the minimum-cost next hop towards the sink;
- what is currently the minimum-cost path towards the sink;
- which is the next node that has received less traffic, and
- which is the next node that has most remaining energy.

This approach is well suited for braided-mesh topologies, where each node has multiple choices on how to forward a packet to the sink [5].

4.3 Room for Improvement

Multipath routing gracefully extends routing over trees to routing over larger topologies. There is, however, room for improvement regarding the following issues:

- *Control traffic*. Constructing and maintaining a larger topology requires (significantly) more control traffic. There is evidence that maintaining more than two paths per sensor (or more than two incoming and outgoing links per sensor node) requires control traffic that outweighs the benefits of multipath, as the required transmissions for control information consume a significant portion of the network resources (energy, wireless bandwidth, processing time) [6]. This problem becomes more pronounced when the network topology changes fast – in fact, for mobile sensor networks, constructing and maintaining multiple paths becomes practically infeasible.
- *Algorithmic complexity*. Identifying an optimal multipath topology is a computationally hard problem, even in the case where all topological information (channel quality between all pairs of nodes) is available at all nodes. Using suboptimal topologies can reduce the potential benefits of multipath.
- *Broadcasting*. Existing proposals do not exploit the inherent broadcasting capability of the wireless medium. For instance, consider an approach where each node that overhears a packet forwards it to the sink – even though it was not the packet's intended receiver. Such an "opportunistic" approach has the potential to increase reliability (because each packet is forwarded through multiple paths), but also waste network resources, including battery life (when there are no node/channel failures). In networks with tens or hundreds of nodes, controlling the number of redundant packets in the network is practically infeasible.

Ideally, we would like to have a multipath collection protocol that requires insignificant control traffic (compared to the actual data traffic), is transparent to the underlying topology changes, and exploits broadcasting. New ideas that have recently emerged from the area of network coding hold the potential to help towards this direction; we will next review the basic ideas in network coding and discuss how they fit in the context of sensor networks.

5 What Is Network Coding

Network coding is a new area that promises to revolutionize the way we treat information in a network, and have a deep impact in all network functionalities, such as routing, network storage, and network design [7–11]. The novel paradigm in network operation that network coding brings is that, instead of having individual source packets traversing a network, we instead have combinations of packets, each bringing some type of "evidence" about the source packets. These packet combinations are created throughout the network: we allow intermediate network nodes to

process their incoming information packets, and in particular, combine them to create new packets. A receiver collecting a sufficient number of such combined packets can use them to retrieve the original information sent by the sources. The area of network coding is centered around the application of this basic idea, of dealing with "evidence" instead of individual packets. The following "classical" example in the network coding literature illustrates the potential benefits over a wireless medium.

Example 4 Consider a wireless ad-hoc network, where devices A and C would like to exchange the binary files x_1 and x_2 using device B as a relay. We assume that time is slotted, and that a device can either transmit or receive a file during a time slot (half-duplex communication). Figure 14 depicts on the left the standard approach: nodes A and C send their files to the relay B, who in turn forwards each file to the corresponding destination.

The network coding approach takes advantage of the natural capability of wireless channels for broadcasting to give benefits in terms of resource utilization, as illustrated in Fig. 14.

In particular, node C receives both files x_1 and x_2, and bit-wise `xors` them to create the file $x_1 + x_2$, which it then broadcasts to both receivers using a common transmission. Node A has x_1 and can thus decode x_2. Node C has x_2 and can thus decode x_1.

This approach offers benefits in terms of energy efficiency (node B transmits once instead of twice), delay (the transmission is concluded after three instead of four time slots), wireless bandwidth (the wireless channel is occupied for a smaller amount of time) and interference (if there are other wireless nodes attempting to communicate in the neighborhood). The benefits in the previous example arise from that broadcast transmissions are made maximally useful to all their receivers.

Note that $x_1 + x_2$ is nothing but some type of binning or hashing for the pair (x_1, x_2) that the relay needs to transmit. Binning is not a new idea in wireless communications. The new element is that we can efficiently implement such ideas in practice, using simple algebraic operations.

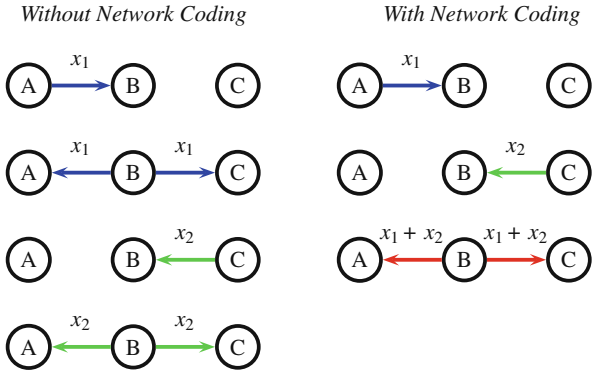


Fig. 14 Nodes A and B exchange information via relay B. The network coding approach uses one broadcast transmission less

5.1 Network Coding in Practice

In Example 3, we implicitly assumed that each node performs fixed encoding operations. The receivers know these operations, and use this knowledge to decode. In particular, nodes A and B know in advance that they will receive the linear combination $x_1 + x_2$. In a practical network, where the network structure, delays and synchronization varies, the selection and knowledge of the linear combination coefficients needs to be distributed in a decentralized manner.

Fortunately, three ideas, that appeared successively in time, give us an elegant and flexible way to perform network coding in a completely decentralized manner. These are:

1. Randomly chose the linear combinations at each network node [12].
2. Append "coding vectors" at the header of each packet to allow the receivers to decode without need of synchronization [13].
3. Use subspace coding to achieve the same goal as in (2) more efficiently [14].

The first idea determines what intermediate nodes in the network do. The second and third offer two alternative approaches for the encoding of the data at the sources and corresponding decoding at the receivers.

5.2 Randomized Network Coding

Assume we have n source packets $\{x_1, \ldots, x_n\}$ that contain symbols over a field $\mathbb{F}_q$ and we want to convey them to multiple destinations over a network using network coding. Throughout the network, intermediate nodes perform linear combining of the source packets. Thus, a destination receives combinations of the form

$$c_1 x_1 + c_2 x_2 + \cdots + c_n x_n,$$

where $c_i \in \mathbb{F}_q$. In the network coding literature, the vector of coefficients

$$c = [c_1,\ c_2,\ \ldots,\ c_n]$$

is called a *coding vector*. Each destination can retrieve the data, if it receives n linearly independent combinations of the source packets, or, n linearly independent coding vectors. For example, let $\{\rho_i\}$ be the combined packets a destination collects, we can write in a matrix form:

$$\begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_n \end{bmatrix} = \underbrace{\begin{bmatrix} c_{11} & c_{21} & \ldots & c_{n1} \\ c_{12} & c_{22} & \ldots & c_{n2} \\ & \cdots & & \\ c_{1n} & c_{2n} & \ldots & c_{nn} \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}. \tag{1}$$

If the linear combinations are independent, i.e., matrix $\mathbf{A}$ is full rank, we can solve the above equations and retrieve the source packets. The task of network code design amounts to deciding what linear combinations to form throughout the network so that each receiver gets a full rank set of equations.

Randomized network coding is based on the simple idea that, for a field size q large enough, there exist so many valid solutions, that even random choices of the coefficients allow us to find a valid solution with high probability. Thus we can simply ask each intermediate node in the network to create and send uniform at random linear combinations of the packets it has received. The associated probability of error can be made arbitrarily small by selecting a suitably large alphabet size [12]. For example, if we could choose the coefficients $\{c_{ij}\}$ of matrix $\mathbf{A}$ in (1) uniformly at random, the matrix $\mathbf{A}$ would be full rank with probability at least $(1 - \frac{1}{q})^n$. In practice, simulation results indicate that even for small field sizes (for example, using $m = 8$ bits per symbol, i.e., $q = 2^8$) the probability of error becomes negligible [15].

Randomized network coding requires no centralized or local information, is scalable and yields to a very simple implementation. Thus, it is very well suited to a number of practical applications, such as sensor networks and more generally dynamically changing networks.

5.2.1 Generations and Coding Vectors

The next question to answer is, even if we randomly select what linear combinations to perform, how do we convey to the destinations what are the linear combinations they have received so that they can decode. Moreover, in a network where information gets generated at a constant rate, we need to decide what packets to combine and how often do we decode. To achieve these, we cannot rely on synchronization, since packets are subject to random delays, may get dropped, and follow different routes.

The approach in [13] first groups the packets into *generations*. Packets are combined only with other packets in the same generation. A generation number is appended to the packet headers to make this possible (one byte is sufficient for this purpose). The size of a generation can be thought of as the number of source packets n in synchronized networks: it determines the size of matrices the receivers need to invert to decode the information. Since inverting an $n \times n$ matrix requires $\mathcal{O}(n^3)$ operations, and also since waiting to collect n packets affects the delay, it is desirable to keep the generation size small. On the other hand, the size of the generation affects how well packets are "mixed", and thus it is desirable to have a fairly large generation size. Indeed, if we use a large number of small-size generations, intermediate nodes may receive packets destined to the same receivers but belonging to different generations. Characterizing this trade-off is an open research problem.

As a second step, the approach in [13] appends within *each* packet header a vector of length n that describes which linear combination of the source packets

$\{x_1, \ldots, x_n\}$ it contains. These vectors are what we called coding vectors. The encoded data is called the *information vector*. For example, the coding vector $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots 0)$, where the 1 is at the ith position, means that the information vector is equal to x_i (i.e., is not encoded). A packet that contains the linear combination $\rho = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$ has the coding vector $(c_1, \ldots, c_n)$ and the information vector ρ.

The coding vectors are updated locally at each node that performs linear combining, to reflect the new linear combination of the source packets that the new packet carries. For example, if a node receives two packets with coding vectors $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots 0)$ and $(c_1, \ldots, c_n)$, with corresponding information vectors x_i and ρ, it can create the new information vector $\alpha x_i + \rho$ for some value $\alpha \in \mathbb{F}_q$. To send this new information vector, it will use the coding vector $(c_1, \ldots c_{i-1}, c_i + \alpha, c_{i+1}, \ldots, c_n)$. Combining can occur recursively and several times inside the network.

Each receiver examines the coding vectors of the packets it receives, to learn what are the linear combinations it has received. In particular, the coding vectors it receives are nothing but the rows of the matrix $\mathbf{A}$ in (1) that determine the linear equations it needs to solve.

Appending coding vectors to packets incurs an additional overhead. For example, for a packet that contains 1400 bytes, where every byte is treated as a symbol over $\mathbb{F}_{2^8}$, if we have $h = 50$ sources, then the overhead is approximately $50/1400 \approx 3.6\%$.

5.2.2 Subspace Coding

The approach based on appending coding vectors in order to be able to decode at the receiver is well suited for large packets where the overhead is small. In wireless sensor networks, and generally, wireless networks, the situation is quite opposite: it is quite often the case that packets consist of a few bits. In such cases, using coding vectors can add a significant overhead.

A new approach recently proposed in [14, 16] promises to be helpful in the case of very short packet lengths. This approach is again designed to work with use of randomized network coding, and is based on using subspaces as "codewords" to convey the information from the sources to the receivers. For simplicity we will here consider a single source transmitting n independent packets to receivers, but the same approach can easily be extended to multiple sources [17]. We note that recent work [18–22] has established that subspace coding only offers benefits as compared to the coding vectors approach for very short block lengths.

Consider a source that would like to convey n independent source packets to receivers over a network that employs randomized network coding. Assume that each packet has length λ over $\mathbb{F}_q$. The n packets can take in total $M = q^{n\lambda}$ values. Thus the source, for each set of packets, has one of these values to convey.

The source can achieve this as follows. First, it selects to operate over an nL dimensional vector space V over $\mathbb{F}_q$, i.e., a vector space, where vectors have length

nL and have elements in $\mathbb{F}_q$. A basis of this space consists of nL linearly independent vectors. For example, the space $\mathbb{F}_2^3$ has the basis

$$\{\mathbf{e}_1 = [1\ 0\ 0], \quad \mathbf{e}_2 = [0\ 1\ 0], \quad \mathbf{e}_3 = [0\ 0\ 1]\}.$$

A subspace π is a subset of the vector space V that is a vector space itself. We can think of subspaces as "planes" that contain the origin. For example, the space $\mathbb{F}_2^3$ contains 7 two-dimensional subspaces. One such subspace is $\pi_1 = <\mathbf{e}_1, \mathbf{e}_2>$. Another is $\pi_2 = <\mathbf{e}_2 + \mathbf{e}_3, \mathbf{e}_1>$. It also contains 7 one-dimensional subspaces, one corresponding to each non-zero vector. Moreover, the subspace (plane) $\pi_1 = <\mathbf{e}_1, \mathbf{e}_2>$ contains the three "line" (one-dimensional) sub-subspaces $\pi_3 = <\mathbf{e}_1>$, $\pi_4 = <\mathbf{e}_2>$, $\pi_5 = <\mathbf{e}_1 + \mathbf{e}_2>$. Therefore, we can define subspaces of lower dimension as sub-spaces of higher dimensional subspaces. In the above example, we can see that $\pi_3 \subset \pi_1 \subset \mathbb{F}_2^3 = V$. We say that two subspaces are *distinct* if they differ in at least one dimension. For example, $\pi_1 = <\mathbf{e}_1, \mathbf{e}_2>$ and $\pi_2 = <\mathbf{e}_2 + \mathbf{e}_3, \mathbf{e}_1>$ are distinct.

The source selects a codebook of M distinct subspaces, and each set of n packets is mapped to a different such subspace. The receivers learn this codebook. To convey the value of the source packets, the source needs to convey what is the particular subspace these packets are mapped to. To do so, it inserts in the network a set of basis vectors (packets) that span the subspace. Assume for example it sends the vectors $\{b_1, \ldots, b_k\}$ that span a subspace π. The critical observation is that, the mixing through randomized network coding intermediate nodes perform, preserves the subspaces. Indeed, linear operations, no matter what these operations are, can only create vectors that are in the span of the basis $\{b_1, \ldots, b_k\}$ and thus within π. As a result, every node that receives k linearly independent vectors will be able to identify which is the subspace π that the source has sent. The source has then transmitted information through the choice of the subspace that it sends. This property makes the use of subspaces for encoding robust to the topology of the network and to arbitrary linear operations performed at the intermediate nodes.

6 Network Coding for Sensor Networks

Network-coding ideas and techniques have already been successfully applied in the context of wireless mesh and/or ad-hoc networks (see [23–26] for some examples). A natural question to ask is, are sensor networks any different? Apart from the fact that both types of networks are wireless, they differ in almost everything else that matters when designing a network protocol: topology, traffic patterns, performance metrics. For example, in wireless mesh/ad-hoc networks, network coding offers throughput benefits when certain multicast or concurrent-unicast traffic patterns occur, such as the well known pattern in Fig. 14. These patterns do not naturally occur in sensor networks, which typically carry many-to-one traffic from the sensors to the sink; for this traffic pattern, network coding does not offer throughput benefits. In many sensor-network applications, throughput is not even relevant

as a metric – as discussed, the two predominant metrics are energy efficiency and reliability.

There is evidence that network coding can help achieve a better energy efficiency/reliability trade-off than traditional single-path or multipath collection protocols. Intuitively, by allowing nodes to mix incoming packets, network coding introduces each piece of information into multiple paths – i.e., network coding offers a new way to implement multipath communication. However, before we examine the potential benefits, we need to solve a crucial practical problem: as the following examples illustrate, in a sensor network where a large number of nodes communicate short messages to the sink, it does not make sense to use neither coding vectors nor subspace coding.

Example 5 Consider a sensor network consisting of 100 sensors, where each sensor periodically communicates a 1-byte message to the sink. Assume that we want to implement network coding using coding vectors and operations over a field of size $q = 2^4$. In this case, we would need 50 bytes of coding-vector data *per packet* to convey just 1 byte of information. Such overhead rules out coding vectors as a practical design option.

Example 6 Subspace coding offers increased efficiency as it removes the need for coding vectors. However, designing subspace code for the case where the sources are not collocated is challenging. Consider for example the case where two sensor nodes use codebooks consist of subspaces of a vector space $\mathbb{F}_q^\ell$, i.e.,

$$\mathcal{C}_i = \{\pi_j^{(i)} : 1 \leq j \leq |\mathcal{M}_i|\}, \quad i = 1, \ldots, n.$$

To transmit information to the sink, source i maps a measured value to one such subspace π and inserts in the network d vectors that span π. In relaying information towards the sink, the sensor linearly combines all packets it has received (including that generated by itself) and transmits the combined packet to the next relays towards to the sink. As a result, the sink will observe vectors from the union of subspaces inserted by all the sources. In particular, if source i inserts the subspace π_i, the sink will observe vectors from the subspace $\pi_1 + \pi_2 + \cdots + \pi_n$. Using the knowledge of the codebooks $\{\mathcal{C}_i\}$, it needs to decode the sensor data.

To be able to correctly decode at the sink, we need to ensure that every combination of sensor data results in a *distinct* union subspace. Assume for simplicity we have two source nodes, S_1 using the codebook $\mathcal{C}_1 = \{\pi_1,\ \pi_2,\ \pi_3\}$, while S_2 the codebook $\mathcal{C}_2 = \{\pi_4,\ \pi_5,\ \pi_6\}$. Table 1 summarizes all outcomes. For this code to be identifiable, we want all (or some) entries in Table 1 to correspond to distinct subspaces. For example, $\pi_1 + \pi_4$ should be a distinct subspace from $\pi_2 + \pi_5$.

This problem is hard to solve even for the case of two sources, and a very small codebook (in our example each node transmits only 3 values). Designing such a code for 100 sources would be an admirable feat, let alone designing a code that can also be efficiently decoded.

Table 1 Coding for two sources

$\mathcal{C}_2/\mathcal{C}_1$	π_1	π_2	π_3
π_4	$\pi_1+\pi_4$	$\pi_2+\pi_4$	$\pi_3+\pi_4$
π_5	$\pi_1+\pi_5$	$\pi_2+\pi_5$	$\pi_3+\pi_5$
π_6	$\pi_1+\pi_6$	$\pi_2+\pi_6$	$\pi_3+\pi_6$

6.1 Code Design

We now describe a network-coding approach for sensor networks. The main intuition in this approach is to attempt to reduce the length of the coding vectors, by restricting the freedom in the coding operations that intermediate network nodes have. Indeed, the classic design of coding vectors, allows potentially *all* source packets to get combined together; our approach is to employ coding vectors that allow at most m packets get combined [18]. This is motivated through three observations:

1. Coding vectors allow us to decode the data even if all the sources' packets get mixed. However, packets get combined only if their paths to the sink overlap. For a network symmetrically deployed around a sink, it is unlikely that all paths will overlap.
2. Not all sensors may be actively measuring during every round. For example, when sensing anomalies, we expect a small fraction of all potential sources to send information.
3. We can artificially restrict the number of sources that get combined, by appending to each packet a few bits to count the number of combined packets it contains.

Our design problem can now be stated as follows. Given n sources, the sink is going to observe packets that contain linear combinations of *at most* m sources. We want to design codes that allow us, by receiving each combined vector, to determine which linear combination of the source packets it contains. Our goal is to utilize vectors of length ℓ that is much smaller than the number of sensor nodes n.

Our construction utilizes properties of erasure correction codes, and proceeds as follows. Select a linear code of length n, minimum distance $2m+1$, and redundancy ℓ, with ℓ as small as possible. Consider the $\ell \times n$ parity check matrix $\mathbb{H}$. Assign to each source as coding vector the 1-dimensional subspace spanned by one column of $\mathbb{H}$. The source, includes this column in front of the data packet it sends, exactly in the same way as in the case of the usual coding vectors.

Intermediate nodes combine up to m source packets to create a coded packet, using randomly selected coefficients. The sink, when receiving each encoded packet can determine by examining the coding vectors:

- which columns of the matrix $\mathbb{H}$ contribute to the resulting linearly combined vector.
- using this knowledge, the exact coefficients that have been used for the combining.

The reason our construction allows to perform these operations, stems from a well known property the columns of matrix $\mathbb{H}$ satisfy. Let $\mathcal{A}$ denote the set of these vectors. Then, if the code has minimum distance $2m+1$, *any set of* $2m$ *vectors in* $\mathcal{A}$ *are linearly independent* [27]. This directly implies the following properties. For vectors in $\mathcal{A}$:

(P1) Any set of $0 < \beta < 2m$ vectors span a distinct β-dimensional subspace.
(P2) The distance between two subspaces π_1 and π_2, where each subspace is spanned by a different set of $0 < \beta < 2m$ vectors, equals $\min\{\beta, 2m-\beta\}$.

From property (P1) we see that the sink receives distinct subspace for *every* distinct set of m sources, and therefore is able to decode the identities and the information. Thus, even though there are $n = |\mathcal{A}|$ possible sources, we do not need to use vectors of length proportional to n, but instead, of length $\ell \geq 2m$.

The following example illustrates this procedure.

Example 7 Consider a code with minimum distance $2m+1 = 5$ and a parity check matrix $\mathbf{H}$ of dimension $\ell \times n$ with columns $h_1, \ldots, h_n$:

$$\mathbf{H} = [h_1 \; h_2 \; \ldots \; h_n] \tag{2}$$

Each source appends a different h_i vector in the header of its information packet, and sends the packet through the network where intermediate nodes combine up to two packets. The sink receives n packets, and extracts from their header n vectors of the form $y_k = \alpha_{k1} h_{k1} + \alpha_{k2} h_{k2}$, for $k = 1, \ldots n$. For each vector y_k, the unknowns are the indices $k1$ and $k2$, i.e., which column vectors are combined, as well as the coefficients α_{k1} and α_{k2}. From property (P1), the sink can uniquely determine the column vectors h_{k1} and h_{k2} in whose span lies y_k. Using this knowledge, it can then uniquely identify α_{k1} and α_{k2}. This implies that the k-th packet received at the sink, contains the linear combination of the packets send by the sources $k1$ and $k2$ with coefficients α_{k1} and α_{k2}.

Thus to decode, the sink can create an $n \times n$ transfer matrix, where row k will contain exactly two nonzero entries: α_{k1} at position $k1$ and α_{k2} at position $k2$, and then solve the system a system of linear equations to retrieve the data.

In order to quantify our savings, we need to examine how ℓ scales with m. This is related to a well-studied problem in coding theory, namely, for a given code length $n \triangleq |\mathcal{A}|$, and a given minimum distance $2m+1$, what are upper and lower bounds on the number of codewords $A(n, m)$ this code can have [27]. Using the Gilbert-Varshamov lower bound and the sphere packing upper bound [27], it can be shown that for large values of n,

$$nH_2\left(\frac{2m+1}{2n}\right) \leq \ell \leq nH_2\left(\frac{2m+1}{n}\right), \tag{3}$$

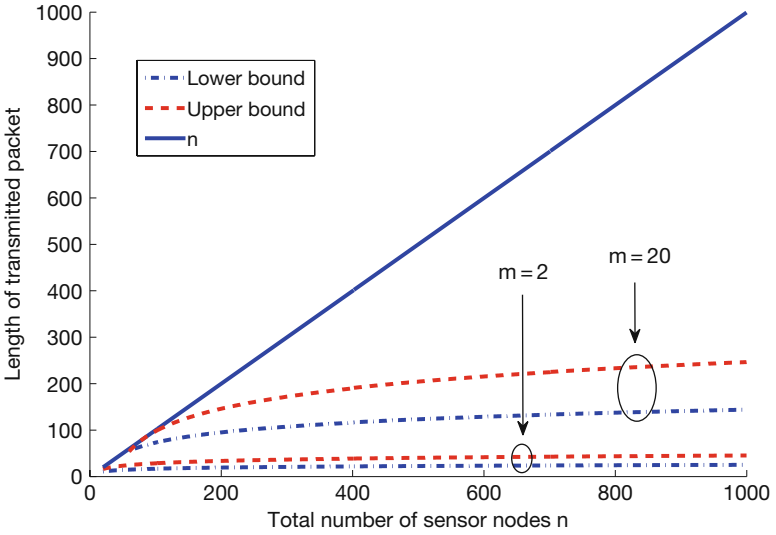


Fig. 15 Bounds on the length ℓ of the coding vectors when $m = 2$ and $m = 20$ sources get combined, as a function of the number of sensor nodes n

where H_2 is the binary entropy function, namely, $H_2(p) = -p \log p - (1-p) \log(1-p)$. Figure 15 plots the bounds in (3) as a function of n, for $m = 2$ and for $m = 20$. We conclude that our proposed code design results in using a fraction of the length n, that goes to zero as the ratio $\frac{2m+1}{n}$ goes to zero.

Example 8 Using a table of the best codes known [27] we can see for example that, there exist binary linear codes of length $n = 512$ with redundancy $\ell = 18$ and minimum distance $2m + 1 = 5$. Thus in a sensor network with 512 nodes if at most $m = 2$ source vectors get combined, we need to use vectors of length $\ell = 18$.

Example 9 An alternative approach to using shortened coding vectors, is to use smaller generations. In particular, we can assume that we divide the source nodes into groups, where each group contains m sources, and allow only the packets that originate from the same set of sources to be combined. A special counter attached to the header of the packet can specify the generation in which the packet belongs. This approach differs from our proposed approach in that, it may happen for a node to have multiple uncoded packets, but not be able to code them together as they belong in different generations.

6.2 Opportunistic Broadcasting with Network Coding

The idea is the following: when a node sends out a packet, apart from the intended receiver, some of the node's neighbors opportunistically overhear and receive the transmitted packet; each node forwards towards the sink a mix of the packets it

receives and the ones it opportunistically overhears. Thus, each piece of information may appear, linearly combined, in a large number of packets across the network. By propagating multiple packets that contain different linear combinations of the original data, we can create multiple opportunities for decoding the original data and achieve increased robustness to path failures.

We now outline a collection protocol that relies on this idea: it constructs a tree and implements opportunistic broadcasting on top. We first describe how the tree is used, then how it is constructed.

Consider a sensor network where nodes have formed a tree rooted at the sink. The network operates in rounds. During each round, each node stores the packets it receives from its children as well as any packets it opportunistically overhears from its other neighbors. It starts transmitting as soon as one of the following conditions is met:

- It has received packets from all its children.
- It has received a packet from at least one child and a time window has expired.
- It has overheard new information that exceeds a predetermined threshold.

A node continues to transmit until its parent signals that it has successfully received one of its packets, or an upper limit on the number of per-node transmissions is reached. Each transmission carries a different linear combination of the node's received/overheard packets – such that, if a neighbor happens to successfully overhear more than one transmissions, each one will bring new information to it.

Example 10 Consider the tree shown in Fig. 5 and assume we use it as described above. Recall that, in this tree, a node's parent is not necessarily its minimum-cost neighbor (it is the next hop on the node's minimum-cost path to the sink). For example, consider node a_{10}; to reach its parent, a_8, it must transmit on average twice; however, it can reach a_7 by transmitting on average once. Hence, if node a_{10} transmits until a_8 acknowledges receipt of a packet, it is likely to transmit twice, and a_7 is likely to successfully overhear both transmissions, i.e., collect two different linear combinations of the packets previously received and overheard by a_{10}. Similarly, if node a_2 transmits until its parent a_3 acknowledges receipt of a packet, its transmissions are likely to be successfully overheard by the sink.

We now discuss how to construct the tree. Assume that we want to ensure that each transmission be received by at least two neighbors of the transmitting node. We can implement this requirement in two ways: (i) *Hard*, where we explicitly mandate that two neighbors acknowledge reception. This effectively amounts to constructing two trees, which, as discussed in Example 2, is computationally hard. (ii) *Soft*, where we require that on average at least two neighbors receive each transmitted packet. We choose the latter and implement it as follows.

Consider a node that has multiple neighbors. In the Collection Tree Protocol (described in Sect. 3), each node chooses its parent so as to minimize the overall cost of communicating to the sink. We now revise this as follows: each node chooses its parent so as to minimize the overall cost of communicating to the sink, subject to the following *redundancy constraint*: a node is not allowed to choose as

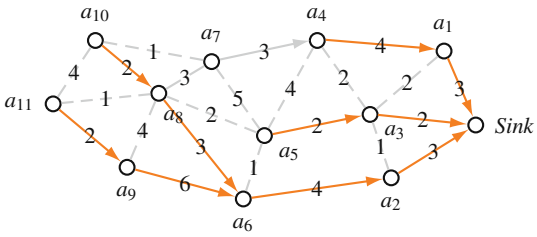


Fig. 16 A tree that satisfies the redundancy constraint: each node is connected to the tree through a link that is not its minimum-cost link, with the exception of the nodes that are connected directly to the sink

its parent its lowest-cost neighbor, unless it has no other choice. This protocol can be implemented as easily as CTP, by simply adding the redundancy constraint.

When a node a transmits a packet and its parent b successfully receives it, it is likely that so do a's neighbors that are connected to it through better links than b. The redundancy constraint ensures that, whenever possible, at least one such neighbor exists for each node. The following example illustrates this property.

Example 11 Figure 16 depicts a tree that satisfies the redundancy constraint. In this tree, node a_4 chooses node a_1 as its parent, whereas its lowest-cost neighbor is a_3. On average, the packets received by a_1 are also received by a_3. The cost of this tree is 34. The cost of the tree constructed by CTP (shown in Fig. 5) was 25. So, we ensure that on average each packet is received by at least two neighbors at the cost of increasing transmissions by approximately 35%.

7 Conclusions

In this chapter, we looked at the problem of designing a protocol that performs the simplest possible sensor-network task: communicate measurements from the sensors to the sink, without any in-network information processing. We described protocols that balance the competing goals of energy efficiency and reliability, ranging from a simple tree-based protocol to multipath protocols, and discussed their basic benefits and drawbacks; we also outlined an emerging protocol based on network coding. We close by noting that a rigorous study would be necessary to evaluate the relative value of each approach and how well each is matched to specific application needs.

References

1. H. Dubois-Ferrière, L. Fabre, R. Meier, and P. Metrailler, "Tinynode: a comprehensive platform for wireless sensor network applications," in *Proceedings of the IEEE International Symposium on Information Processing in Sensor Networks (IPSN)*, April 2006.
2. N. Burri, P. von Rickenbach, and R. Wattenhofer, "Dozer: ultra-low power data gathering in sensor networks," in *Proceedings of the IEEE International Symposium on Information Processing in Sensor Networks (IPSN)*, April 2007.

3. R. Fonseca, O. Gnawali, K. Jamieson, S. Kim, P. Levis, and A. Woo, "The collection tree protocol," http://www.tinyos.net/tinyos-2.x/doc/html/tep123.html, 2006.
4. Z. Li and B. Li, "Improving throughput in multihop wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 55, pp. 762–773, 2006.
5. S. De, C. Qiao, and H. Wu, "Meshed multipath routing with selective forwarding: an efficient strategy in wireless sensor networks," *Computer Networks*, vol. 43, no. 4, pp. 481–497, 2003.
6. D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing for wireless sensor networks," in *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, October 2001.
7. R. W. Yeung and N. Cai, "Network error correction, i: basic concepts and upper bounds," *Communications in Information and Systems (CIS)*, vol. 6, pp. 19–35, 2006.
8. N. Cai and R. W. Yeung, "Network error correction, ii: lower bounds," *Communications in Information and Systems (CIS)*, vol. 6, pp. 37–54, 2006.
9. C. Fragouli and E. Soljanin, *Network Coding: Fundamentals*, ser. Foundations and Trends in Networking. Now Publishers, Delft, 2007, vol. 2, pp. 1–133.
10. C. Fragouli and E. Soljanin, *Network Coding: Applications*, ser. Foundations and Trends in Networking. Now Publishers, Delft, 2008, vol. 2, pp. 135–269.
11. T. Ho and D. S. Lun, *Network Coding: An Introduction*. Cambridge University Press, Cambridge, UK, 2008.
12. T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, pp. 4413–4430, 2006.
13. P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proceedings of the Allerton*, October 2003.
14. R. Koetter and F. Kschischang, "Coding for errors and erasures in random network coding," in *Proceedings of the IEEE International Symposium on Information Theory*, June 2007.
15. Y. Wu, P. A. Chou, and K. Jain, "A comparison of network coding and tree packing," in *Proceedings of the IEEE International Symposium on Information Theory*, June 2004.
16. D. Silva and F. R. Kschischang, "Using rank-metric codes for error correction in random network coding," in *Proceedings of the IEEE International Symposium on Information Theory*, June 2007.
17. M. Jafari Siavoshani, C. Fragouli, and S. Diggavi, "Non-coherent network coding for multiple sources," in *Proceedings of IEEE International Symposium on Information Theory*, 2008.
18. M. Jafari Siavoshani, L. Keller, C. Fragouli, and K. Argyraki, "Compressed network coding vectors," in *Proceeding of the IEEE International Symposium on Information Theory*, June 2009.
19. M. Jafari Siavoshani, S. Mohajer, C. Fragouli, and S. Diggavi, "On the capacity of non-coherent network coding," in *Proceedings of the IEEE International Symposium on Information Theory*, June 2009.
20. S. Mohajer, M. Jafari Siavoshani, S. Diggavi, and C. Fragouli, "On the capacity of multisource non-coherent network coding," in *Proceeding of the IEEE Information Theory Workshop*, June 2009.
21. D. Silva, F. R. Kschischang, and R. Koetter, "Capacity of random network coding under a probabilistic error model," in *Proceeding of the 24th Biennial Symposium on Communications*, June 2008.
22. A. Montanari and R. Urbanke, "Coding for network coding," arXiv:cs.IN/0711.3935, 2007.
23. C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "A network coding approach to energy efficient broadcasting: From theory to practice," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, April 2006.
24. S. Katti, S. Gollakota, and D. Katabi, "Embracing wireless interference: analog network coding," in *Proceedings of the ACM SIGCOMM*, August 2007.
25. S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "Xors in the air: practical wireless network coding," in *Proceedings of the ACM SIGCOMM*, August 2006.

26. C. Gkantsidis, W. Hu, P. B. Key, B. Radunovic, P. Rodriguez, and S. Gheorghiu, "Multipath code casting for wireless mesh networks," in *Proceedings of the ACM CoNEXT*, December 2007.
27. F. J. MacWilliams and N. J. A. Sloane, *The theory of error correcting codes*, ser. North-Holland Mathematical Library. North Holland, Amsterdam, Netherlands, 1983.

Part II
Theory and Practice

Data Aggregation in Wireless Sensor Networks: A Multifaceted Perspective

Sergio Palazzo, Francesca Cuomo, and Laura Galluccio

Abstract Wireless sensor network applications usually do not require knowledge of each individual sensor measurement at the sinks; rather, an overall aggregate of the data monitored by different sensor nodes could be sufficient. Dealing with aggregated data is potentially highly convenient in the network perspective because it leads to a reduction in the overall energy consumption, even if the needed processing involves resource consumption. The reason is that through aggregation the unnecessary packets overhead is avoided and the waste of energy and bandwidth resources are reduced. Moreover, since during in-network data aggregation some pre-elaboration of the data can be made, several wireless sensor network applications may benefit from having this pre-elaborated data and may be facilitated in handling and processing the sensor measurements. Thus, data aggregation is a promising approach for improving the performance of sensor networks and has recently attracted the interest of researchers. In this chapter we provide a survey of the main approaches at data aggregation recently proposed and compare different potential perspectives for a data aggregation taxonomy in wireless sensor networks. Then, we introduce a more comprehensive classification which allows to embrace the existing ones and to incorporate the different phases of the data aggregation process: information description, propagation and preservation.

1 Background

The concept of "data aggregation" has been widely used within the computer science and communication fields with different meanings. For the only sake of mentioning a couple of example, we can refer to the process of gathering personal account information ("screen scraping") from various sources; likewise, the concept can also be applied to advertisement purposes, referring to capture different pieces

S. Palazzo (✉)
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni, University of Catania, Catania, Italy
e-mail: sergio.palazzo@diit.unict.it

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_6,

of data, summarizing them according to specific user application requirements, for purposes such as statistical analysis, content personalization, data mining, etc.

In this networking context, *aggregation*, which can be defined as "collecting of units or parts into a mass or whole", is a generic concept to be applied to various types of network scenarios, where it might be interpreted as the process of combining and compressing messages/packets into a smaller number of them, though containing the same information. The intrinsic characteristics of wireless sensor networks (WSNs) make them a really suitable framework to apply these techniques.

In wireless sensor networks, due to the high density of deployed devices and the type of monitoring or alerting applications being supported, the interest is usually not in the knowledge of each individual sensor reading but, rather, on the overall estimation of the parameters of interest monitored within a particular area. To accomplish this, data aggregation techniques can be effectively used. Hence, data aggregation and/or fusion for WSNs can be specialized as the process of forwarding and fusing together data coming from different network nodes to provide a unified feedback to the sink(s).

Numerous application scenarios for data aggregation have recently emerged. As an example, in chapter "Body Sensor Networks for Sport, Wellbeing and Health", body sensor networks (BSNs) are considered and their performance in terms of the inference process adopted in applications like sports, wellness and health is discussed. Also in chapter "Body Sensor Networks for Sport, Wellbeing and Health", data aggregation is proposed to be combined so as to allow the reduction in the magnitude of data traveling into the network through the selection of the most relevant features from specific sensors for decision tasks related to the BSN.

Data aggregation usually implies introducing some additional processing at intermediate network nodes so as to reduce the amount of data packets traveling throughout the network. This is made to counteract redundancy effects, i.e., packets with redundant information which would uselessly be forwarded throughout the network, overloading intermediate nodes and wasting energy and bandwidth resources. This is particularly true in case of event monitoring applications, where sensor nodes are required to send data to the sink(s) when they detect unusual behavior in the network. In such scenarios, due to the potentially high density of devices, it is likely that nodes located in the same area forward almost the same data to the sink, thus overloading and unnecessarily stressing intermediate network nodes. Another issue is related to periodic monitoring applications, in which sensor nodes, time by time, report their data to the sink(s), thus causing a temporal correlation in the received measurements, when no significant variations in the monitored metrics happen within consecutive monitoring periods.

It should be pointed out that the impact of useless transmissions is critical in terms of power consumption, because transmission of a single bit over the radio causes a consumption at least three orders of magnitude higher than the one required for executing a single typical instruction at the device processor [46].

Moreover, when considering data aggregation together with reduction in energy consumption, positive effects on the limitation of network congestion are also observed. It is well known, in fact, that due to the so-called funneling effect, when

propagating data towards the sink, nodes few hops away from the sink are the most stressed, both in terms of energy consumption and amount of traffic to manage. Thus, aggregation brings about a reduction in the amount of traffic that these funneled nodes should forward and, therefore, lessens the chances of incurring in congestion at these nodes [51, 12], so avoiding possible network partitioning due to nodes' misbehavior. Another problem which data aggregation can help to counteract is related to the effects of error readings. In fact, it is common that potentially compromised readings within a sensor network might cause unnecessary actions, when propagated to the sink. As an example, if a damaged sensor, say D, detects that the monitored parameter exceeds a predefined threshold and sends the corresponding alert message, this could cause the sink to transmit unnecessary backward control messages which are propagated to other nodes in the proximity of D, asking for confirmation of the previous alert. On the contrary, the use of data aggregation can be really efficient in avoiding such harmful effects, as a single wrongly operating device's piece of data is weighted and averaged with others sent by properly working devices and, thus, unnecessary actions are not invoked. This error reading effect could become quite relevant in certain scenarios, e.g., chemical monitoring, fire alert and disaster relief networks, etc. Hence, avoiding these unnecessary alerts is important and data aggregation can be very beneficial to this.

However, data aggregation could also cause some drawbacks in network behavior. First, by implying the need for additional processing at network nodes, it leads to an increase in the complexity of the operations performed by network nodes and, consequently, in the latency for data delivery. This latency is originated by the need of intermediate nodes to perform some aggregation, which implies additional time, mostly consumed during the waiting process, for gathering the required number of packets before aggregating data and sending them towards the sink. It is worth highlighting that it is believed (see, for example [46]) that the level of complexity is tightly related to the delay incurred in data delivery. In most of the application scenarios, the use of rather simple operations, such as averaging might be enough, but there may be other cases in which the loss of data is crucial, and aggregation embraces more complicated (and thus costly) metrics, such as median, consensus, percentiles, etc. It is clear that when performing aggregation, strict synchronization among network nodes is required to avoid misinterpretation of data and use of older information. Also nodes are expected to employ strict synchronization in terms of appropriate delay to wait before aggregating and sending out the aggregated data. This is because waiting too long would lead to excessive delivery delay and the possibility to bring to the sink stale and useless information. On the contrary, by waiting too short time, we could risk not to exploit aggregation in an efficient way, thus sending only few aggregated data.

By considering the specific time constraints being required, it looks evident that aggregation cannot be applied to all application scenarios. As an example, in case of alarm applications, the additional delay that is a consequence of aggregation might be detrimental to the timeliness of the data delivery.

When aggregating data packets, attention should also be paid to the way processing of data is performed. In fact, on the one hand, increasing the number of packets

whose contents can be aggregated together is useful to reduce energy consumption and bandwidth overloading; on the other hand, this could imply increase in distortion and possible loss of information, since compressing too many packets leads to losses in terms of data integrity [44]. Hence, an appropriate trade-off in the way aggregation is achieved is needed.

Summarizing, the main advantages of using data aggregation over wireless sensor networks are (see Table 1):

- Reduction in energy consumption and consequent increase in network lifetime.
- Decrease in the overhead of redundant packets traveling throughout the network, thus avoiding unnecessary storage and processing.
- Reduction of the overall network load thus giving rise to a mitigation of congestion effects on highly loaded nodes in the network.
- Smart filtering of data sent to the sink(s) and delivered to the control center.

On the other hand, data aggregation also faces some critical aspects, among which the following ones can be highlighted:

- Increase in data latency and need for strict synchronization between an aggregator node and its neighborhoods, which could become critical in case of delay sensitive applications.
- Accuracy and data integrity reduction due to inappropriate aggregation; in this sense, without proper preservation criteria, the data finally delivered to the destination could be unreliable and useless, due to the increasing level of distortion.
- Need to consider the impact of duplicates in case of use of more complex aggregation metrics; this requires distinguishing between aggregation metrics that are sensitive to duplicates and those which are not.
- Need of appropriate coding because of the additional processing required by sensor devices.

Table 1 Advantages and disadvantages of using data aggregation

Advantages	Disadvantages
Reduction in energy consumption	Increase in data latency and need for synchronization
Decrease in the overhead of redundant packets	Possible loss of accuracy and integrity of data
Mitigation of congestion effects	Need for minimization of the duplicates impact
Smart filtering of data	Need for appropriate coding

1.1 Terminology

Data aggregation is the process through which a combination of the messages traveling into the network is used to represent all the information which was originally conveyed by the initial packets. This methodology brings about several improvements for the operation of the WSN, especially in terms of energy and

bandwidth saving. Due to the typical characteristics of data aggregation which requires that intermediate nodes perform aggregation of the en-route packets traveling towards the sink, aggregation is often referred as *in-network aggregation* [10].

With respect to data aggregation, WSNs offer an ideal scenario because of *data-centricity*, i.e., the focus on the information itself received by the sink, and not on the messages traveling the network themselves or the nodes which emit them.

Different aspects should be addressed in the data aggregation process: querying from remote users, collection of nodes measurements, their combination and concatenation, compression and mining at the final destination. Accordingly, different terms and associated functionalities can be discussed (see Fig. 1):

- Data fusion
- Data querying
- Data gathering
- Data concatenation
- Data compression
- Data mining

The term *data fusion* refers to the combination of different pieces of data into a single one; one can imagine, e.g., a use case in which the interest is in monitoring the minimum temperature within a room; in this case, the fusion would be as easy as taking the minimum value from all the received messages and forwarding a unique packet. Usually in the literature data fusion is distinguished from data aggregation in that in the former the focus is mostly on information extraction, while in the latter the main focus is on data manipulation. This implies that data aggregation can be seen as a process included in data fusion.

In order to implement a *data aggregation* procedure, a database representation is usually employed. The collection of sensing devices is thus seen as a complete database, on which traditional management operations are carried out. The most relevant one is *data-querying*, where data is requested from the network (either fulfilling any particular criteria or not). In this sense, it is quite likely the use of SQL-type semantics when describing the required data aggregation operations.

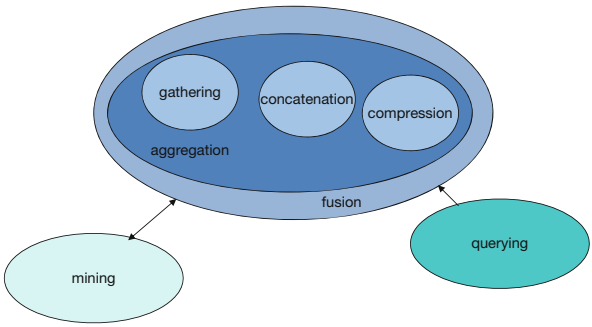


Fig. 1 Relationship among the data aggregation terminology

Data gathering refers to the process of sending and collecting the various sensor nodes measurements at the sink. Consequently the term gathering mostly refers to the methodology of collection, thus encompassing routing aspects rather than focusing on the process of information elaboration itself.

Data concatenation refers to elaboration of single messages into a bigger one (provided that the maximum transfer units of the underlying protocol stack are compatible with it). This implies reducing the number of packets to be processed, the energy consumption as well as other side-effects (e.g., the possibility of having collisions).

Data compression, refers to the use of compression techniques to reduce the amount of bytes required to code the different pieces of information and, thus, the traffic load which needs to be processed within the network. An an example, Slepian-Wolf coding [48] is a promising distributed source coding technique that can completely remove the data redundancy caused by spatially-correlated observations. This technique is based on the assumption that each sensor node has an a priori knowledge of the correlation structure, which depends on the distances between sensor nodes and the characteristics of the observed phenomenon.

Finally, at the sink, *data mining* refers to the process of selecting useful information from the huge amount of data collected in a sensor network and processing and filtering them.

It is worth highlighting that the use of any of the aforementioned mechanisms is not exclusive and a combination of them might be used; in fact, they are normally employed together, as parts of the same overall process which, in the rest of this chapter, will be denoted as data aggregation.

1.2 Typologies of Data Aggregation

The performance of the data aggregation procedures depends on the way the data collected by sensor nodes are worked out and propagated to the sink. To this purpose we can distinguish between a data aggregation process which reduces the amount of information propagated throughout the network or not. More specifically, when aggregation is performed by some intermediate nodes, the aggregators can

- *Combine and compress the data packets by preserving their payloads*. This means that there is only a reduction in the header size while the two payloads are merged together. As an example let us suppose that node 1 sends a temperature measurement, let's say T_1, to the sink, and node 2 sends measurement T_2. Accordingly node 3, being an aggregator, will emit a data packet with the payload containing $T_1 + T_2$. This kind of aggregation is referred to as *lossless aggregation* since it does not result in any loss of information. This approach results particularly useful when the data readings are small in size as compared to the maximum allowed data packet size, and thus aggregation can be efficiently performed by merging more packets. Observe that this approach is appropriate when sensor readings related to different types of measurements are received. As

an example, if node 1 sends a temperature measurement to the sink, and node 2 sends a pollution measurement, data cannot be merged but can be inserted in a unique packet.
- *Combine and compress the data packets by fusing their payloads*. In this case, both the overhead and the payload of the packet are reduced by putting and compressing together their contents. As an example let us suppose that node 1 sends a temperature measurement, let's say T_1 to the sink, and node 2 sends measurement T_2. Accordingly node 3, being an aggregator, will emit a data packet with the payload containing the "fused" T_1 and T_2 measurements. The result of the "fusion" process depends on the aggregation function being chosen. As an example, if the aggregation function is average, the average of the two measurements is computed and sent. When considering this kind of aggregation, it is evident that a *lossy* process happened since the initial values cannot be recovered at the sink. However, in the majority of the sensor network applications, knowing the aggregate at the sink is more than sufficient to keep an updated vision of the network status. This approach is useful in case of homogeneity in the data received by the aggregator node.

The performance of the aggregation process is also strictly related to the aggregation rate, i.e., the number of packets being aggregated per time unit. To this purpose, different categories can be distinguished as in [49, 10]:

- *Periodic simple aggregation*, when an aggregator node collects all the received readings for a fixed amount of time and then puts together the received data without any consideration of any time constraints. This approach, although simple, can be not very efficient when only few not significant packets are received by the aggregator which is then forced to send a packet, even when it is not necessary.
- *Periodic per-hop aggregation*, when a collection tree is considered and an aggregator node issues the aggregate only when it has received a packet from all its children. Again this approach is very simple, but can lead to inefficient waiting times when some nodes are less active than others and send data quite rarely.
- *Periodic per-hop adjusted aggregation*, when a more adaptive tuning of the frequency at which the aggregator nodes issue the aggregate packet is obtained by adjusting a time out based on the position of the node along the collection tree.

Finally, the performance of the aggregation also depends on the type of aggregation function being used. The classification proposed in [31] is concerned with the way the aggregation threats the measurements received to produce an aggregated data having the value of representative sample. The properties considered for the classification of the aggregates are:

- *Duplicate sensitive* where the sensitivity of the aggregates to duplicate readings being received is considered.
- *Exemplary* where a representative value over the set of all values is generated but is unpredictable, thus not giving any guarantees on the reliability of the value.

- *Summary* where a representative value is estimated over all values thus giving a more reliable estimate that derives from considering a more stable value whose contribution is estimated over all values.
- *Monotonic* where two partial state records are such that the aggregate is always higher or lower than the partial state values used for estimating it.

Table 2 Classes of aggregates defined in [31]

	Duplicate sensitive	Exemplary (E), Summary (S)	Monotonic	Partial state
Max, Min	No	E	Yes	Distributive
Count, Sum	Yes	S	Yes	Distributive
Average	Yes	S	No	Algebraic
Median	Yes	E	No	Holistic
Count Distinct	No	S	Yes	Unique
Histogram	Yes	S	No	Content-sensitive

In Table 2, the partial state of the aggregates can be in one of the following states:

- *Distributive* where the partial state records represent the aggregate for the partition of data where they are computed. Accordingly, the size of the partial state records is the same as for the final aggregate.
- *Algebraic* where the partial state records are of constant size and are not themselves aggregates for the partitions.
- *Holistic* where the partial state records are proportional to the size of the set of data in the partition.
- *Unique* that are similar to holistic aggregates with the exception that the amount of state that must be propagated is proportional to the number of distinct values in the partition.
- *Content sensitive* where the partial state records are proportional in size to some (perhaps statistical) property of the data values in the partition.

2 Perspectives for a Taxonomy of Data Aggregation

Several chapters have proposed different approaches to taxonomy of data aggregation techniques, some of them providing comprehensive summaries and comparisons of existing data aggregation solutions. In general it is not possible to identify a comprehensive and complete classification but different schemes can be envisaged. The possible perspectives are the following ones:

1. *Layer-centric taxonomy*, where a classification of the aggregation techniques is considered in accordance to the specific layer of the protocol architecture of the WSN which the technique impacts.
2. *Ingredient-centric taxonomy*, where the key elements of a data aggregation procedure, referred to as *ingredients*, i.e., the networking protocols, the aggregation functions and the data representation [10], are considered.

3. *Performance-centric taxonomy*, where the impact of the data aggregation solutions on performance metrics like latency, accuracy, energy, fault tolerance and security [2] is taken into account.
4. *Information-centric taxonomy*, where the solutions proposed in the literature are considered in the perspective of their fitting into the dimensions of the information processing performed during data aggregation, i.e., description of information, propagation of information and preservation of information, as shown in Fig. 2.

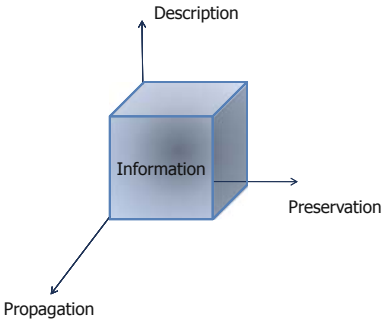


Fig. 2 Information processing dimensions being related to data aggregation

In the following sections we will discuss the different classifications and understand the reasoning behind each of them. In doing this, we will privilege the information-centric taxonomy since it allows to embrace different aspects which are disregarded by other taxonomies. However, in an attempt of clarity, at the end of this chapter, Table 3 summarizes how several proposed solutions for data aggregation can be classified with respect to different taxonomies.

3 Layer-Centric Taxonomy

Although data aggregation is a complex procedure requiring a joint action performed across different layers of the network architecture, a first classification of the aggregation techniques can be made from a layered perspective. More specifically, aggregation techniques can be distinguished on the basis of the specific architectural layer mainly involved. As a consequence, we can distinguish an aggregation approach operating at the link layer from a protocol operating at network or higher layers. This classification may facilitate the understanding of the main building blocks needed to perform the aggregation at each layer and, on the other hand, may highlight the need for cross layer interactions.

When considering aggregation solutions proposed in the literature, we can mainly distinguish between approaches specifically working at the link layer and solutions involving a network layer action. Together with this, few solutions implying an action performed at the application layer have also appeared.

When performing data aggregation at the link layer, solutions proposed in the literature are typically based on aggregating data packets at the link level by reducing the number of contending nodes, taking into account possible data correlation, and allowing only to a node to send data packets at each time on behalf of other nodes. This has been proposed for example in [50].

When performing aggregation at the network layer, the focus is usually on aggregator nodes positioning and appropriate design of routing protocols which allow to force traffic to go through aggregator nodes distributed throughout the network.

Finally, when considering the application layer, techniques are usually related to use of appropriate semantics to code the data emitted by sensor nodes while also considering the possibility to exploit spatial and temporal data correlation.

4 Ingredient-Centric Taxonomy

A different perspective in classifying the aggregation techniques is the one proposed in [10]. In this chapter three basic *ingredients* of the data aggregation are discussed, i.e., routing protocols, aggregation functions and representation of data, which correspond to the different phases in the design of the aggregation process.

Concerning the first aspect, i.e., the routing protocols, it represents the basic ingredient for data propagation into the network. To this purpose, the most appropriate path to aggregate data in the most efficient way using the highest number of data packets should be designed. However, optimal aggregator nodes placement is still an open problem since algorithms for aggregation tree construction are NP hard and, thus, heuristic solutions are typically considered. Also, there is need for more appropriate routing protocols, different from those used in classic ad hoc networks which do not rely on identification of the shortest path connecting source and destination so as to minimize delivery delay, but, instead, can be opportunistically tuned to collect the highest possible number of correlated data packets. This is mainly related to the data-centric paradigm employed in sensor networks.

Accordingly, when focusing on routing, mainly three routing approaches can be considered:

- *Tree-based* approaches where a classic tree scheme rooted at the sink is built usually following a shortest path methodology.
- *Cluster-based* approaches that are a traditional alternative to tree-based schemes and have been proposed for reducing energy consumption in sensor networks.
- *Multipath* approaches which are a more reliable and robust alternative to both cluster and tree-based approaches. In this case, data are propagated to the sink(s) through multiple paths, thus avoiding to have a unique failure point and increasing the chance to have at least one copy of the packet delivered to the sink.

Once the choice on the networking paradigm to be used has been done, the next step is choosing the aggregation function. With respect to this, the choice is mostly related to the applications running into the network. In fact, if different choices

are available, e.g. average, max/min, standard deviation, sum, count, histogram, etc., the choice of one aggregation function rather than another makes sense only if the application is known. As an example, let us consider an application aiming at identifying possible fires in a forest. If the sink sends requests about environmental temperature measurements to the sensor nodes, providing average values does not give any significant insight into occurrence of a fire in a specific area.

A third fundamental ingredient for the data aggregation is data representation, i.e., how to represent the data at the final destination maintaining a suitable insight into the original data itself. This aspect, as stressed by [10], mainly depends on the applications. In some cases it may be fundamental to have insight on the exact location from where a fraction of the aggregated data comes from, as well as on the time instant such part of data has been captured and processed. Spatial, temporal and semantic correlation of the readings may be captured by sophisticated data aggregation protocols.

5 Performance-Centric Taxonomy

An alternative classification of the data aggregation algorithms is proposed in [2]. Here the authors mainly discuss how the aggregation protocols can impact on network performance. In fact, as already discussed, data aggregation protocols can have significant advantages and drawbacks and strongly impact on network performance. The following performance metrics are considered:

- *Energy efficiency*: reducing energy consumption is a primary target in sensor networks. This is because such a reduction can lead to an increase in network lifetime. Accordingly, data aggregation can help to reduce the amount of redundancy traveling into the network. To this purpose one of the main concerns is about how to build the delivery trees to the sink while increasing the chances to efficiently aggregate at some network nodes. Innovative solutions addressing the use of more sophisticated sensors to be placed in appropriate locations could be considered. In this way simple wireless sensors will be leveraged from more complex processing. However, attention should be paid to the increase in energy consumption at aggregator nodes which could cause a decrease in their lifetime.
- *Latency*: performing aggregation implies introducing some processing and consequent latency in network delivery. A trade-off between energy and delay is to be identified. In fact, on the one hand, an increase in the number of nodes where packets are aggregated leads to a reduction in the global amount of energy depleted to relay packets but, on the other hand, this would imply waiting longer at the aggregation points and delaying the final delivery, so causing decrease in data “freshness”. Observe that minimization of network delay is tightly related to the choice in the aggregation function. In fact, the higher the complexity in the aggregation function, the longer the time needed to process the data at each step.

- *Accuracy*: when considering accuracy as an indicator of the number of nodes who contributed to a specific measurement delivered to the sink, it is evident that, the higher is the number of nodes, the higher is the delay incurred by the delivered packet to wait for the collection of the contributions of neighbor nodes at an aggregator. Consequently, choice of the most appropriate number of nodes contributing to an aggregate packet can impact on delay and energy consumption performance. Another aspect related to the accuracy in the delivered information is the number of data packets fused together which cannot be compressed without any limitation because of information reconstruction problems and possible loss of information.
- *Fault tolerance*: due to the intrinsic features of wireless communications such as collisions, channel time variability, channel quality degradation and asymmetries, transmission of data in this scenario can incur into errors. Also, use of single path approaches leads to the possibility to loose data when either nodes' failures, misfunctioning or propagation errors are met. Conversely, reliability can be improved by means of multipath topologies which, however, would cause increase in energy consumption and possibility to count more than once the same sensor reading. This can become critical depending on the kind of aggregation function being considered.
- *Security*: one of the most critical aspects when performing aggregation is related to the resilience to different security attacks, the most simple of which is misbehavior of aggregators. In fact, aggregators can send compromised data towards the sink. To cope with this problem, traditional authentication mechanisms can be proposed as well as techniques to aggregate encrypted data without decryption. This however implies that complex procedures should be managed by limited sensor nodes.

In addition to the performance metrics identified in [2], another key parameter to be taken into account in the performance evaluation is the *network load*. A good aggregation solution should in general reduce the network load and mitigate the effects of overload that can arise in correspondence to specific network nodes as the ones close to the sink (funneling effect). To this aim, in [9] the normalized number of transmissions is used as the metric to compare different protocols. A normalized number of transmissions represents how effective a protocol is in aggregating packets and in [9] is defined as the number of transmissions in the network divided by the number of contributing sources. The number of contributing sources represents the real number of pieces of information that are generated by all sources in the network and arrive at the sink.

6 Information-Centric Taxonomy

An information-centric classification of the data aggregation algorithms can also be proposed. To this purpose, different aspects can be considered:

- Description of information.
- Propagation of information.
- Preservation of information.

Usually, in the data aggregation literature the above mentioned aspects are separately addressed meaning that no overall solutions addressing the various aspects of a data aggregation information-centric paradigm have been proposed.

When focusing on *information description* we usually refer to the way sensor readings, once collected by wireless sensor devices, are processed and formalized in order to be sent to the sink. Along the path(s) this data will require to be re-elaborated, modified and processed in the aim of fusing the data at some intermediate aggregators. Information description in wireless sensor networks typically embraces problems of query processing, data storage, data filtering and mining.

Once the sensor readings have been described, they should be propagated up to the sink(s). *Information propagation* can be successful, provided that solutions of medium access and networking problems are addressed. This implies that also aspects related to packet scheduling and propagation tree construction are considered as well as wireless resource sharing, cluster identification or multipath set up in the aim of increasing robustness and reliability. The above topics could be considered both under a theoretical and an algorithmic perspective.

Finally, when the aggregated data are received by the sink(s), attention should be paid to the fidelity of the collected information. In the aim of *information preservation*, data should be aggregated with the aim of reducing the distortion rate while maximizing the amount of aggregated information delivered to the sink(s). Also, considerations on security level aspects should be done. In fact, as discussed above, aggregation is a process which incorporates data manipulation and, thus, is more prone to security and/or privacy violations. Accordingly, security solutions for data preservation should be proposed.

It is evident that the above taxonomy calls for the identification of a trade-off between contrasting targets. For example, it is intuitive that, if the information has to be accurately described, i.e., a significant amount of data has to be transmitted, then propagation might suffer. Similarly, to preserve information, redundancy has to be added, thus decreasing data propagation efficiency.

In the next sections we will recall protocols and solutions for data aggregation in accordance to the information-centric classification. This choice is motivated by the fact that, although the other three paradigms allow to capture interesting perspectives of the data aggregation process, the information-centric paradigm better encompasses the key information processing aspects determined by the overall data aggregation procedure. The information-centric classification in fact, on the one hand, allows a detailed understanding of the impact of the data aggregation process on the information in terms of relevant description, propagation and preservation; on the other hand, it captures the main functional elements and protocols in the view of understanding the implementation issues behind each solution and the impact on performance.

On the contrary

1. the tentative to associate a data aggregation process to a specific layer of the protocol stack may result an enforcement, especially in a WSN context, where cross-layer solutions are typically established; as a consequence the layered paradigm may not represent comprehensively the main solutions present in the literature;
2. the classification of data aggregation techniques in accordance with the functional paradigm may not capture some aspects related to (i) layers (e.g., some schemes that mainly operate at the link level are not considered in the functional paradigm presented in [10]); (ii) some application requirements in capturing the information (e.g., related to the semantic that can be associated to the sensor readings);
3. the performance-driven perspective gives a detailed insight into the behavior of aggregation procedures but does not allow a detailed view of the aspects related to the algorithms and protocols used to implement such procedures and how they fit into an architectural perspective.

Observe that the information-centric classification allows to embrace many aspects not considered by the other data aggregation paradigms. However, the latter can be superimposed in the sense that the majority of the literature solutions discussed in the following can be classified according to all the different paradigms. To this purpose, at the end of the chapter, Table 3 shows how the discussed data aggregation solutions fit into the different aggregation paradigms.

6.1 Description of Information

In order to properly design the data aggregation process, the semantics needed to correctly disseminate queries and collect and fuse sensor data should be specified. With respect to this, only recently some effort has been devoted to these issues. Chronologically, the first work where aggregation has been discussed in the view of a wireless sensor network encompassing also semantic aspects is [23]. In directed diffusion data generated by sensors are named as an attribute-value pair. Interests are disseminated by a user via the sink node and employed to request specific data from sensor nodes. Intermediate nodes along the path towards the sink(s) are expected to aggregate the data. However, in [23], aggregation is considered as an application-related operation which is assumed to be coded in low level languages like C, thus implying severe requirements on the user programming competencies. An example of an interest being disseminated in the directed diffusion paradigm is shown in Fig. 2 in case of vehicle tracking application.[1]

Data descriptors too are in a similar form. An effort in the direction of a simplification in the semantics used by the user to notify queries is presented in [53]. In this work a semantic stream framework is proposed, which allows users to diffuse some

[1] The pseudo-code is taken from [23].

```
TYPE = wheeled vehicle    detecting vehicle location;
INTERVAL = 10ms    notify events every 10 ms;
DURATION = 60s    overall duration of the monitoring;
RECT = [−200, 200, 100, 350]    area where sensors are selected for performing the task;
```

Fig. 3 Directed diffusion interest

queries from the network in the form of declarative statements. Also constraints on the quality of the measurements, i.e., confidence intervals, can be supported. However, this framework does not support the possibility to perform aggregates into the network.

To cope with complexity problems arising in directed diffusion, a simpler framework where consideration of the semantics behind an aggregation process is presented in [31]. Aggregation is regarded as a core service to be provided through system software. In this perspective the authors propose to use high-level programming abstraction to let users request data from the sensor network without taking into consideration any programming problem and without any correlation to routing functionalities. To this purpose, the Tiny Aggregation (TAG) Service for sensor networks has been introduced. In [31] TAG provides an interface for data collection and aggregation, driven by selection and aggregation facilities in database query languages. Moreover, TAG allows to evenly distribute the aggregation procedures among network nodes, thus also reducing energy consumption. Moreover, TAG is conceived as independent from both the routing details and the programming procedures. The SQL-like syntax on a single table identified with the one owned by each sensor node is inspired by TinyDB [33], which is a query processor for sensor networks. Queries are in the form of the pseudo-code of Fig. 4.

```
SELECT {agg(expr), attrs} FROM sensors;
WHERE {selPreds};
GROUP BY {attrs};
HAVING {havingPreds};
EPOCH DURATION i;
```

Fig. 4 Generic TAG query[2]

As in the SQL language, in Fig. 4 the *SELECT* clause is used to specify the expression (i.e., *expr*) by which sensor readings are selected. The parameter *attrs* specifies the attributes by which sensors are partitioned; *agg* is the aggregate function being considered among those discussed in Sect. 1.2. The *WHERE* clause is used to filter specific sensor readings based on location criteria. The *GROUP BY* clause specifies a partition of the sensor readings chosen according to the *attrs*

[2] The pseudo-code is taken from [31].

attributes. The *HAVING* clause filters some groups which do not satisfy specific predicates (i.e., *havingPreds*). The *EPOCH DURATION* clause specifies when the updates should be delivered.

As an example, a query iterated for 3 min requesting average sensor temperature measurements on the rooms located on the fourth floor of a hotel and exceeding 27°C temperature could be in the form shown in Fig. 5.

```
SELECT {AVG(temperature), room} FROM sensors;
WHERE floor = 4;
GROUP BY room;
HAVING AVG(temperature) > 27C;
EPOCH DURATION 180s;
```

Fig. 5 Example of a TAG query about temperature measurements

Basically, only few differences between TAG and SQL queries exist. The output of TAG queries is a stream of values and not a single value, since sometimes in monitoring applications it is preferred to have more data than a single value which better characterize network behavior. The Aggregation, i.e., the *agg* function, is here performed using three functions: a merging function f, an initializer i and an evaluator function e. f has the structure $< z >= f(< x >, < y >)$ where $< x >$ and $< y >$ are partial state records that register intermediate computation states at certain sensors which will be required to compute the aggregate; $< z >$ is a partial state obtained by applying f to $< x >$ and $< y >$. The initializer i is needed to say how to instantiate a state record for a sensor value; the evaluator e instead takes a partial state record and estimates the aggregate. TAG queries can be characterized through attributes, where each node is provided with a catalog of attributes. Attributes represent sensor monitored values, such as acoustic volume, temperature, or sensor parameters such as the residual energy. When a TAG sensor receives a query, named fields are changed into local catalog identifiers. If a node does not know the attribute it labels this attribute in its query as NULL.

TAG aggregation is performed in two steps: in the first one, called *distribution phase*, aggregation queries are inserted throughout the network. In the second phase, called *collection phase*, aggregated values are routed into the network.

The main advantages of TAG are: decrease in communication overhead with respect to a centralized aggregation approach and reduction in the number of messages transmitted by a node during each epoch, independently of its depth in the tree. This allows to save energy resources at nodes closer to the sink.

It is worth mentioning that also the Cougar project [4] discusses the definition of a sensor database system and illustrates some details about the system developed within the project itself. However, aggregation is not considered in this scenario and each sensor is modeled as an abstract data type.

Taking as inspiration the TAG approach, in TiNA [45] temporal correlation is considered so that energy consumption can be reduced while also preserving the

quality of data by suppressing sending readings which do not differ much from the recent ones previously sent by a sensor node.

To this purpose, each query contains an additional field denoted as *VALUES WITHIN tct* which is used to let the user specify the *temporal coherence tolerance* for the specific reading. As an example, if the *VALUES WITHIN tct* field is set to 0.2, it means that a sensor node will report back to the user only readings which differ more than 20% from the previous ones.

As an evolution of the semantic approach proposed in [31], a new Semantic Sensor data fusion is presented in [21]. In this work the synthesis between the use of the MicroToPSS middleware constrained by some knowledge on the application expressed through an appropriate application ontology is discussed. The focus is here on a methodology for processing and providing low-level sensor data to high-level applications. Accordingly, low-level data should be appropriately filtered, aggregated and correlated from heterogeneous and various sensor network sources. The final target would be to allow applications to exploit sensor data while disregarding low-level languages and interfaces. Accordingly, applications will express their interest in semantic events that are transformed into sensor network level events. The employed middleware, denoted as MicroToPSS, allows management of data flows to support specifically distributed automation applications. The most interesting aspects of this middleware are the use of an SQL-like language, the abstraction of tasks which results similar to object oriented abstraction, and the distributed run-time data flow optimizer which works transparently from the application.

A system for semantic data fusion in sensor networks based on MicroToPSS is proposed in [54]. In this chapter the authors propose to use a content-based publish/subscribe (CPS) technique [3] to relate application scenarios to semantic data fusion of low-level sensors data. Their approach takes inspiration from the MicroToPSS middleware and defines semantic level events independently of the sensor interfaces and applications. The CPS is a messaging model where notifications are delivered to nodes based on their interests. Different roles are distinguished in CPS: *subscribers, publishers* and *brokers*. Subscribers emit subscriptions, that are filtering constraints on t-uples, which describe the kind of data they are interested in. These subscriptions are stored by brokers. Publishers issue publications, that are sets of attribute-based t-uples, to brokers who take care of forwarding publications to subscribers based on the received notifications. To overcome the limitations of CPS which is based on a strict and rigorous syntax, a semantic evolution of the CPS is implemented. The complete scheme of the data flow in the system from sensors to applications is shown in Fig. 6.[3]

Each sensor node runs a Micro-ToPSS middleware used to perform data aggregation by acting at the sensor level. Sensor Gateways and CPS clients are used to provide a bridge between the sensor network and the application domain. Sensor data are sent to CPS clients and are then re-emitted as publications and issued to

[3] The figure is taken from [54].

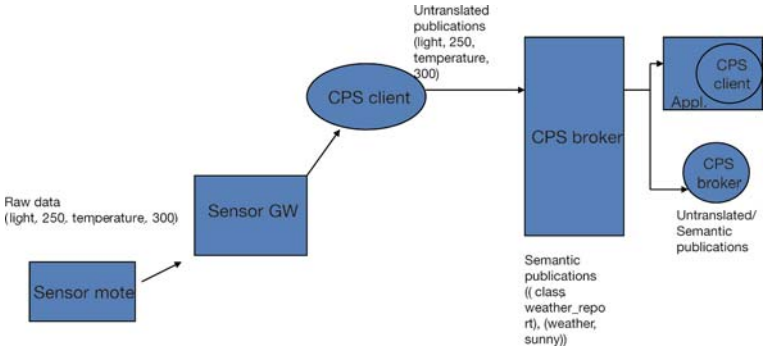


Fig. 6 Data flow in the Micro-ToPSS based system from sensors to application

the CPS broker. The broker then takes care of translating sensor data in high level semantic information through the use of ontologies as defined in [54]. Raw sensor data can be translated into semantic publications by exploiting the semantic engine in the brokers. An example of untranslated raw data issued by a sensor could be *(temperature, 300K)*. This raw data reading can be translated into a semantic publication in the form *(weather, hot)*. Also aggregated translations are considered. As an example two readings *(temperature1, 300K)* and *(temperature2, 400K)* lead to an aggregated estimate *(temperature, 350K)* and a semantic publication in the form *(weather, very hot)*.

6.2 Information Propagation

The information propagation of aggregated data requires the availability of nodes to process, transmit and route packets towards the sink(s). Different aspects should be taken into account and can be combined to suitably aggregate data in the propagation phase. These aspects are:

1. The way the medium access is managed.
2. The way packets are scheduled.
3. The way propagation path are selected.

Suitable combinations of these aspects lead to efficient data aggregation mechanisms.

6.2.1 Medium Access Management

Authors of [9] highlight that efficient aggregation requires packets to converge both spatially and temporally, i.e., packets to meet at the same node (spatial convergence) and at the same time instant (temporal convergence). For spatial convergence, an anycast-based approach called Data-Aware Anycast (DAA) has been proposed in [9]. The DAA approach uses anycast at the MAC layer for determining the next-hop

node at each transmission. Anycasting employs RTS packets to stimulate CTS responses from neighbors before packet transmission. An Aggregation ID (AID) is used to associate two packets to be aggregated. This AID of the transmitting packet is inserted within the RTS and any neighbor that has a packet with the same AID can respond with a CTS. However, in the DAA approach packets may not get aggregated if they are spatially separated (more than one hop). For such a case, authors study a temporal convergence technique to improve performance. Authors notice that deterministically assigning the waiting time to nodes such that nodes closer to the sink wait longer before transmission, can avoid the problem of interference and increase the chance of aggregation. However this results in a fixed delay for all packets wherever the packets are generated, either close to or far away from the sink, and the delay incurred by the waiting time is proportional to the size of the network, which could turn into a disadvantage for large sensor networks. Therefore, in order to introduce artificial delays and increase temporal convergence Randomized Waiting (RW) is proposed to be used at the source for each data packet. Each node when generating a new packet to transmit, delays it by an interval chosen from 0 to τ, where τ is the maximum delay. The combination of DAA with RW has shown to improve the normalized load as much as 73% while RW can significantly reduce the normalized overhead in terms of number of transmissions.

Differently from [9], in [50] in order to increase network performance, spatial correlation is exploited to set up an aggregation-like mechanism working on medium access control. The authors propose to exploit spatial correlation so as to decrease the number of transmitted packets using regulation of medium access. To this purpose, they develop an analytical framework to estimate the distortion suffered at the sink, depending on the number of sensor nodes collecting information and their correlation parameters. Then, they apply this methodology to the definition of a MAC scheme which limits the amount of redundant information traveling throughout the network. This MAC protocol, denoted as CC-MAC, consists of two components: the *Event MAC (E-MAC)*, which is in charge of filtering correlated components, and the *Network MAC (N-MAC)*, which manages high priority given to route-thru packets (i.e., packets that are not newly generated but are crossing router nodes). The basic functioning of E-MAC and N-MAC is similar to CSMA/CA, but some modifications are introduced. In E-MAC correlated information regions are identified where a representative sensor node transmits data for a certain time while other nodes remain silent. After each transmission a new representative is chosen as shown in Fig. 7.[4] Nodes, upon hearing the representative transmission, do not transmit if they belong to the same correlation area; hence, redundancy is avoided and thus overhead reduced. The N-MAC, instead, takes care of prioritizing route-thru packets with respect to those generated by other concurrent events in other correlation areas traversed during propagation into the network. Priority is managed using an IEEE 802.11 PCF like mechanism. Results show the good performance

[4] Figure taken from [50].

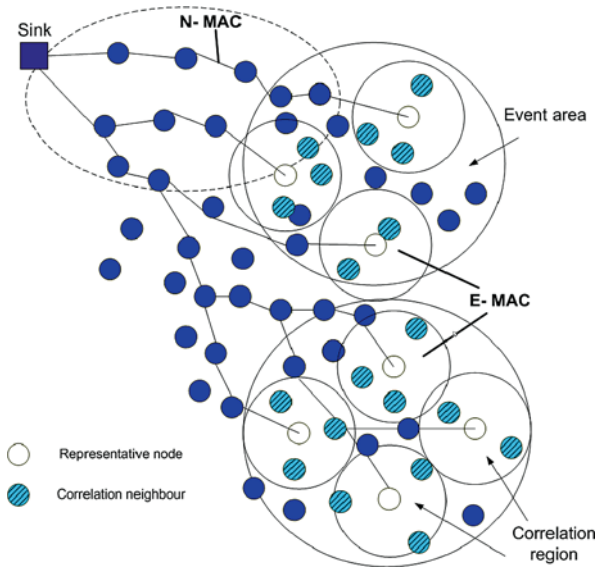


Fig. 7 E-MAC and N-MAC

achieved by CC-MAC in terms of increase in energy efficiency, reduction in packet drop rate and latency.

6.2.2 Packet Scheduling

The way packet transmissions are scheduled also plays an important role when considered jointly with data aggregation. In [39] an application-aware data aggregation (AADA) scheme, which adaptively applies different aggregation policies, depending on the type of packets (i.e., real-time (RT) and non real-time (NRT)), is proposed. Specifically, on the one hand, NRT packets wait at a queue until the number of accumulated packets is equal to the maximum aggregation limit; on the other hand, RT packets, which are sensitive to the aggregation latency, are transmitted as soon as possible, without waiting at the queue. The AADA scheme consists of three modules: queueing, aggregation, and transmission. In the queueing module, a FIFO queue with a size of D packets is considered. The RT packets are delivered to the aggregation module without any queueing. NRT packets, instead, are first queued in the queueing module and then accumulated in the aggregation queue and flushed when the FIFO queue is full or a RT packet arrives. At the data aggregation module, RT and NRT packets can be aggregated into a single packet. The number of the aggregated packets depends on how soon an RT packet arrives. Namely, an aggregation packet is composed of zero or one arrived RT packet and zero or multiple NRT packets that have been accumulated in the aggregation queue. In [39] an analytical model to evaluate the AADA scheme in terms of the average aggregation degree and aggregation latency is proposed. The numerical results show the

significant energy saving gain, the effects of the arrival rate of RT packets, and the impact of the choice of the aggregation threshold. Due to its simplicity, requiring a simple queueing scheme, the AADA protocol can be easily implemented in resource constrained wireless sensor nodes.

In [18], the authors propose an adaptive application-independent data aggregation (AIDA) paradigm where aggregation decisions are made in accordance with an adaptive feedback-based packet-scheduling scheme that dynamically controls the degree of aggregation in accordance with time-varying conditions.

The main idea is to have an aggregation module that resides between the data-link and the network layer which aggregates packets through network unit concatenation. The aggregation component combines network units into a single AIDA payload to reduce the overhead incurred during channel contention and acknowledgment. The use of a separate aggregation layer allows to isolate aggregation decisions from application specifics. This component is generalized enough to be utilized over a wide range of applications without incurring the costs of rewriting components to support application-specific logic. AIDA takes the timely delivery of messages as well as the protocol overhead into account to adaptively adjust aggregation strategies in accordance with assessed traffic conditions and expected sensor network requirements. Simulation results show that AIDA can adapt to varying traffic situations and reduce network congestion and transmission energy consumption. AIDA performs lossless aggregation allowing the upper layer to decide whether information compression is appropriate at the time. Finally, the proposed design enables AIDA to remain complementary to other data aggregation strategies while providing significant time saving benefits in the lower layers of the communication stack. In [18] an extensive simulation analysis is provided which shows benefits of AIDA in terms of reduction of end-to-end transmission delay under heavy traffic loads and reduction in transmission energy consumption. A physical implementation of AIDA on the Berkeley testbed provides initial evidence of the savings obtainable by an application-independent aggregation scheme.

6.2.3 Propagation Path Structure

When dealing with information propagation emphasis should be also given to the path structure used to convey data towards the sink. Data aggregation techniques are typically implemented by using hierarchical structures. Traditionally, two approaches have been considered: tree-based and cluster-based. However, for some specific applications, it may be better to implement techniques and protocols which do not require an explicit maintenance of a structure, thus leading to efficient data aggregation. These techniques are usually referred to as hybrid-based approaches (or structure-free). Also, in order to improve reliability, multi-path solutions are explored.

Tree-Based Approaches

Tree-based protocols are usually built exploiting traditional shortest path routing trees. The research is focused on how to choose a good routing metric to build the

trees based on attributes to facilitate data aggregation. Based on directed diffusion, the authors in [22] propose the Greedy Incremental Tree (GIT) algorithm. GIT establishes an energy-efficient path and greedily attaches other sources to the established path.

In [28] tree based data-centric routing is modeled and its performance is compared with traditional end-to-end routing schemes. With a data-centric protocol, the sources send data to the sink(s), but routing nodes can inspect the content of the data, performing aggregation on multiple input packets. Simple aggregation functions (such as duplicate suppression, min, max, etc.) are considered, and multiple input packets can be aggregated into a single output packet. The impact of source-destination placement and network density are analyzed with respect to energy costs and delay associated with data aggregation. Moreover, since the problem of building an optimal data aggregation tree is generally NP-hard, some suboptimal data aggregation tree generation heuristics are presented, discussing the existence of particular polynomial cases. The proposed modeling shows that whether the sources are clustered close to each other or randomly located, significant energy gains are possible by using data aggregation. This gain increases when the number of sources is large, when they are located relatively close to each other and far from the sink(s). However, the modeling suggests that aggregation latency could be not negligible and should be taken into consideration during the entire design process. In [28] three data-centric routing schemes, denoted as Center at Nearest Source (CNS), Shortest Path Tree (SPT) and a new data-centric version of Greedy Incremental Tree (GITDC) are compared to illustrate the advantage of data aggregation for energy saving. The authors observe that GITDC performs better in terms of average number of transmissions. In [8] a shortest path tree with parent energy-awareness is proposed. Here the neighbor node having the shortest distance from the sink(s) and highest residual energy is chosen as parent. One of the drawbacks of the above tree-based data aggregation protocols is the need to exchange a lot of signalling in order to build and maintain the tree structure. Also, another serious drawback, common to the tree structures, is the lack of network connectivity brought about by the failure of some branches of the tree. This aspect becomes critical in case of unreliable networks where cluster or hybrid approaches are thus preferred.

Most of these tree-based aggregation routing protocols are not designed for event tracking applications. GIT can be used in such a scenario, but it suffers from the cost of pruning branches, which might lead to high cost in moving event scenarios. In [56] the Dynamic Convoy Tree-Based Collaboration (DCTC) is proposed. Essentially, DCTC tries to balance the tree in the monitoring region to reduce energy consumption; however it assumes knowledge of distances from the center of the event to the sensor nodes, which may not be a sensible assumption in all tracking applications. In addition, DCTC incurs in heavy message exchanges, which is not desired when the data rate is high, and furthermore, its performance highly depends on the accuracy of the mobility prediction algorithms.

In [47] a mechanism for data collection, aggregation and dissemination in wireless sensor networks is proposed. The key idea is sweeping over the network using a wavefront which propagates throughout the network, visiting each node only once

and performing the required computations and scheduling. The sweep is achieved using some active nodes moving within the network and sending invitations to other devices to join the wavefront. At each node a potential is considered which, in conjunction with gradients defined by the Directed Diffusion paradigm, can be used to guide the sweep. Upon passing the sweep, aggregation of data is performed (however, no details on the way aggregation can be realistically done are given). Aggregation inside the sweep region is performed by exploiting a local tree, thus avoiding the drawbacks of global tree solutions in terms of possible lack of connectivity. Simulation results show that, as compared to TAG, sweeping allows to decrease the amount of undelivered data and exhibits higher robustness.

Aggregation Tree Construction in a Nutshell

Concerning optimal aggregation tree construction it is a well known NP hard problem [15]. In fact, it would require the identification of the, so-called, Minimum Steiner Tree. To understand what the Minimum Steiner Tree is, let us consider a given graph $G = (V, E)$, where V is the set of nodes, E is the set of possible interconnections between pins, (u, v) is an edge, $w(u, v)$ is its weight and A is a subset of the set of vertices $A \in V$. The Minimum Steiner Tree problem consists in finding the smallest tree connecting all the vertices of the tree A where aggregator nodes can be located at intermediate positions with respect to given network nodes.

The solution of the Steiner Tree problem has many applications in network design and wiring layout problems but can be also applied to aggregation tree construction problems. The Steiner Tree problem can be associated to the minimum spanning tree problem that will be discussed in the following but presents some differences. In fact, by solving the Steiner Tree problem we find intermediate node locations where aggregator nodes can be positioned while this is not the case when solving the spanning tree problem. Unfortunately, as said above, the Steiner Tree problem is NP hard and, thus, heuristics can be used to solve it. Most typical approach relies on consideration of the minimum spanning tree problem and on its solution. Another heuristic relies on identification of the shortest path. If this is the case, traditional approaches, such as the Bellman-Ford or the Dijkstra algorithms, can be used for building the tree.

The Minimum spanning tree problem applied to a graph relies on the identification of an acyclic subset of the interconnections $H \subseteq E$ that connects all vertices, while minimizing the total weight. This problem can be typically solved by using the Kruskal [29] and the Prim [41] algorithms. The Kruskal's algorithm uses a greedy approach to solve the spanning tree problem. It creates a forest, i.e., a set of trees; then it selects a set containing all the graph edges and step-wise removes an edge with minimum weight from this set. If an edge connects two different trees it is added to the forest and the tree get unified. Finally, the forest forms the searched minimum spanning tree. The complexity of the Kruskal's as well as the Prim's algorithm is $O(ElogV)$ and the algorithm can be run using binary heaps.

The Prim's algorithm keeps on increasing the size of the tree starting with a single vertex and spanning over all vertices, until the set of vertices V' becomes

equal to V. In order to add the vertices, an edge (u, v) where $u \in V'$ and $v \in V$ is chosen if it has the minimal weight. When multiple edges with the same weight can be identified, an arbitrary choice can be done. Finally, when $V' = V$, the algorithm stops.

However, due to the limited energy capabilities of the sensor network nodes as well as the kind of application data, it can happen that the aggregation tree changes in time. Accordingly, it could be expensive to compute multiple multicast trees and, thus, a variation of the Steiner Tree problem, called universal Steiner tree problem, can be considered [24]. Similarly, in [25], the authors propose a single group independent spanning tree algorithm, denoted as GIST. This algorithm builds a tree A so that any sensor group S uses the subtree induced by S on A as its group aggregation tree. By comparing the performance of this algorithm to those of the minimum spanning tree and the shortest path in terms of aggregation cost and delay it can be observed that GIST achieves $O(\log n)$ approximate aggregation cost and $O(1)$ approximate delay, for all groups S. This shows that the algorithm works fine with respect to traditional approaches.

Cluster-Based Approaches

Differently from tree-based approaches, clustering consists in grouping nodes which are relatively close to each other while choosing a representative, typically denoted as clusterhead, to perform packet forwarding and processing on behalf of the entire group. LEACH [20] and PEGASIS [30] are two representative protocols working in the perspective of cluster-based approaches. LEACH consists of a distributed cluster formation technique that enables self-organization of large number of nodes, algorithms for adapting clusters and rotating cluster head role to evenly distribute the energy load among all the nodes, and techniques to enable distributed signal processing to save communication resources. As described in [20], nodes organize themselves into local clusters, with one node acting as the cluster head. All non cluster head nodes transmit their data to the cluster head which, once received, performs signal processing functions on the data (i.e., data aggregation), and transmits them to the remote sink(s). Due to the overhead required to be a cluster head node, leadership should randomly rotate so as to avoid that these nodes quickly use up their limited energy and die out. The operation of LEACH is divided into rounds. Each round begins with a set-up phase when the clusters are established, followed by a steady-state phase when data are transferred from the nodes to the cluster head and then to the sink(s).

In [19], authors propose a modified version of LEACH, named LEACH-C, which uses the base station, i.e., the sink(s), to broadcast the cluster head assignment so as to extend network lifetime. Based on LEACH, some modifications have been proposed to refine the cluster head election algorithm by letting every node broadcast and count neighbors at each setup stage, where qualified potential nodes bid for the cluster head position. This modification scatters cluster heads more evenly across the network without requiring the participation of the sink(s) but, due to packet

broadcasting performed at the highest transmission power, only slight improvements over LEACH are obtained.

Reducing the number of cluster heads is critical to conserve energy, since these nodes stay awake and transmit to the sink(s) using high power. Accordingly, in [30] PEGASIS was designed. PEGASIS organizes all nodes in a chain and lets them play the role of chain heads in turn. Since there is only one head node in PEGASIS and in addition there is no simultaneous transmission, latency becomes an issue. To address this, authors propose two chain-based PEGASIS enhancements in [26] and [5]. These two approaches usually save less energy than their predecessor, but outperform it in an energy per delay metric that they use. Based on both LEACH and PEGASIS, authors propose Hybrid Indirect Transmission (HIT) [16], a hybrid scheme which combines the above ones. HIT still uses LEACH-like clusters, but allows multi-hop routes between cluster heads and non-head nodes.

Limitations of LEACH- and PEGASIS-based protocols are related to the assumption that the sink(s) can be reached by any node in only one hop, which limits the size of the network where the approach is applicable. Furthermore, the combination of CSMA, TDMA and CDMA makes the design complex and cost-inefficient. In addition, in scenarios where data cannot be perfectly aggregated, LEACH-based protocols do not perform well, since the cluster head has to send many packets to the sink(s) using high transmission power. Last, the chain-based nature of PEGASIS-like protocols makes them suitable only for scenarios where packets can be perfectly aggregated into one packet of equal size.

Multipath-Based Approaches

If in-network aggregation is performed using a tree-based approach (such as in TAG [34] and [32]) a single failure results in an entire subtree of values being lost. In particular, if the failure is close to the sink(s) it could significantly affect the resulting aggregate. For this reasons solutions based upon multi-path routing were proposed in [7] and [37] in order to overcome these robustness problems. In these works it is pointed out that while, for duplicate-insensitive aggregation function, such as MIN and MAX (where the final results does not depend on the number of times the same value has been considered), a multi-path approach provides a fault-tolerant solution, for duplicate-sensitive aggregates, such as COUNT or AVG, multipath is not satisfactory. In particular in [7] approximate methods, exploiting the duplicate-insensitive counting sketches previously introduced in [11], are proposed. The authors extend the well-known approximate counting sketches for COUNT in order to handle SUM and AVG aggregates. The use of duplicate-insensitive data structures, which approximate a duplicate-sensitive aggregate, allows to achieve the robustness typically associated with multi-path routing. In the proposed algorithm, under the hypothesis of continuous queries, the aggregate is computed once for each time interval, called epoch. The algorithm consists of two phases. During the first one the query is distributed across the sensor network using some form of flooding; each node computes its hop distance from the root and learns the level values of

its neighbors. In the second phase, time is divided into a series of epochs specified by the query. An epoch is then sub-divided into a series of rounds, one for each level, starting with the highest one (the farthest level from the sink(s)). During each round, the nodes at the corresponding level compute their sketches, combine them with those received during the previous round and broadcast the aggregate. In the last round the sink combines the sketches received from its neighbors and produces the final aggregate. In [37] the problem of double counting of data when multi-path routing is exploited is addressed. The proposed approach, denoted as order and duplicate insensitive (ODI), exploits the idea of synopsis that is a small digest summarizing the partial aggregation obtained at intermediate nodes. Synopsis consists of generation, fusion and evaluation. The first algorithm allows to produce the synopsis of data for a given reading generated at a source. Fusion is used to put together data stored by two synopsis and evaluation is used to derive the final result as a summary of various synopsis. To preserve the network against duplicates, synopsis is designed so as to guarantee certain properties: commutativity, idempotence and associativity.

A multipath-based approach for achieving network coding in WSNs and guaranteeing robustness is described in the chapter "Multipath Diversity and Robustness for Sensor Networks". By using network coding each source packet may appear, linearly combined, in a large number of packets through the network. By propagating multiple packets that contain linear combinations of the source packets, it is possible to create multiple opportunities for decoding the source data, and achieve increased robustness to specific node or path failures. From this perspective network coding can be seen as a form of data aggregation performed by network nodes during the relay of packets towards the sinks.

Hybrid and Structure-Free Approaches

Although structured solutions, such as tree- and cluster-based approaches are suitable for data gathering applications, they incur in high maintenance overhead in dynamic scenarios, especially in case of event-based applications.

In [35] a hybrid approach to aggregate data in wireless sensor networks is proposed. The proposed algorithm, called Tributary-Delta, aims at combining the advantages of tree and multi-path schemes by simultaneously running them in different regions of the network. The motivation of this work lies in the fact that, in case of high loss rate, it is possible to significantly increase robustness by using multiple paths towards the sink(s). On the other hand, under low loss conditions, the tree-based scheme is more suitable. Keeping this in mind, in [35] the area of interest (where sensing devices are deployed) is divided in two regions: the Delta region, including the sink, where nodes (called M nodes) use a multi-path scheme to forward packets and the Tributary region where nodes (called T nodes) use a tree-based scheme (see Fig. 8[5]).

[5] Figure taken from [35].

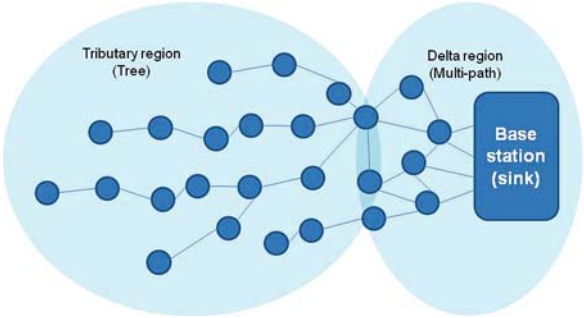


Fig. 8 Tributaries and deltas

In order to keep the entire network working properly, the authors introduce a correctness condition to be satisfied: a multi-path partial result can only be received by a node running the multi-path scheme. This condition implies that M nodes form a multi-path "delta", including the sink, which is fed by trees of T nodes ("tributaries") and that nodes running the tree-based scheme can assume that are no duplicates in their input. The goal of Tributary-Delta algorithm is to adapt the extension of the aforementioned regions according to the network conditions by dynamically shrinking or expanding the Delta region. This can be obtained by switching the operating mode of the nodes that lie in the border zone between the two regions. In particular, users are required to specify a threshold on the minimum percentage of nodes that should contribute to the aggregate answer. Depending on the comparison between this percentage and the specified threshold, the sink decides whether to expand or shrink the Delta region. A proper choice of this parameter is important since a wider Delta region results in an improvement of answer accuracy (in case of high loss rate), while shrinking usually results in more nodes running the tree-based algorithm, thus leveraging to a lower approximation error (in case of low loss rate). Simulation results show that Tributary-Delta performs better than existing pure tree- and multi-path-based approaches.

6.3 Preservation of Information

In this section we will discuss some solutions, which have been introduced in the literature with the aim of preserving the integrity of information, under two different points of view. The first one is related to the way aggregation is pursued by means of manipulating the data. More specifically, in Sect. 6.3.1, we will consider how the aggregation process impacts the reliability and integrity in the information delivered at the sink. Then, in Sect. 6.3.2, we will consider possible security threats met at aggregator nodes in the network which could compromise the reliability of the information delivered at the sink, possibly causing un-necessary actions into the network or preventing the sink from restoring the correct functionalities of the sensor network.

6.3.1 Preservation of Integrity

When focusing on preservation of integrity in the data aggregation process, attention should be paid to determine an appropriate trade-off between the need to reduce energy consumption through exploitation of spatial and temporal correlation in the transmitted data, and reduction in the distortion rate of the information delivered to the final user. In fact, if, on the one hand, reducing the amount of data delivered through aggregation is useful to increase network lifetime, on the other hand, extreme aggregation can imply the impossibility to recover the original information or incurring in an excessive distortion rate.

To this purpose in [14] the joint problem of aggregator nodes placement for deploying a power-efficient sensor network, provided that distortion constraints are given, is dealt with. To this purpose the authors first address the one-dimensional problem to figure out some insights on the process. Then, they move to a bi-dimensional problem, which has been shown in the previous literature to be NP complex and cannot be solved in polynomial time. Thus, the authors focus on a specific and simplified network scenario and derive some conclusions by re-addressing an extended version of the one-dimensional scenario. With respect to the relevant literature in the field, which addresses the problem of joint rate allocation and appropriate transmission structure construction, here the additional distortion constraint is considered.

The addressed problem can be described as follows. In the area $\mathcal{A}$, N sensor nodes are deployed. The sample generated by each node can be represented as a random variable $X(u, v, t)$ where (u, v) are used to describe the location of the node and t is the time instant. Multihopping is exploited to let sensor nodes send their samples to the sink. These samples will be delivered, provided that a maximum D_M, and an average D_A, distortion values are guaranteed. D_M is the maximum distortion allowed at each point of $\mathcal{A}$, while D_A is the distortion per unit of area $\mathcal{A}$.

To minimize the total cost, the quantity $\sum_{j=1}^{N} R_j \cdot C_C(j)$ should be minimized where R_j is the total amount of data transmitted by node j and $C_C(j)$ is the transmission cost per bit at node j. The sensing model is assumed to be Gaussian with zero mean. By denoting as $R_X(d)$ the covariance function of the stationary random field $X(u, v)$, the correlation function is in the form $R_X(d) = e^{-a \cdot d}$ where d is the Euclidean distance.

The chosen communication model is the classical path loss one where the exponent k varies in the range between 2 and 4. Finally, the aggregation model considers that the total data rate sent from a generic node j can be represented by means of the entropy as

$$H(j) = 1/2 \cdot \log(2\pi e)^{|T_j|} \det(R[T_j]) - |T_j| \log \Delta \tag{1}$$

where

- $|T_j|$ is the number of nodes in the subtree of node j, having assumed a pre-existing tree routed at the sink.
- $R(T_j)$ the covariance matrix of the nodes in j's subtree.
- Δ the quantization step in case of uniform quantization.

The sink, once received the quantized values sent by each of the N nodes, interpolates them and obtains the reconstructed value $\hat{X}(u, v)$. To solve the problem of minimizing the energy consumption, Voronoi regions are identified for each sensor j. An example of Voronoi cells in the linear and circular area case are shown in Figs. 9 and 10. The problem thus reduces to

$$\begin{cases} \min\{\sum_{j=1}^{N} R_j \cdot C_C(j)\} \\ \text{s.t.} \\ 1. \quad \bigcup_{j=1\ldots N} V_j = A \\ 2. \quad \text{MSE}(u, v) = 2\sigma^2(1 - e^{-ad_{j,i}}) \leq D_M \\ 3. \quad \frac{1}{A}\int_A \text{MSE}(u, v)\text{d}u\text{d}v \leq D_A \end{cases} \tag{2}$$

where

- $d_{j,i}$ is the distance between node j and node i,
- MSE is the distortion error between the quantized and the unquantized sample.

The optimized placement problem is solved in the one-dimensional case by showing that the shortest path allows to obtain the optimal performance and an associated Lagrangian optimization problem can be solved in closed form.

In the two-dimensional case, the wheel placement case is considered where nodes are located radially on the spokes of the wheel and data delivery is done by exploiting multihop towards the sink located in the center of the wheel. The distortion constraint is such that it is satisfied along the spoke and the optimal number of spokes and nodes on each spoke are heuristically chosen.

As an evolution of this work, the possibility to exploit Slepian-Wolf (SW) [48] coding can be considered. SW coding is a kind of source coding technique which can be exploited to allow un-coordinated communication among sensor nodes while reducing the amount of data packets delivered to the sink, and still preserving the entropy of the sensor measurement readings. SW coding can fully remove the data redundancy caused by the spatially-correlated observations in WSNs. This technique is based on the assumption that each sensor node has a priori knowledge of the information correlation structure with respect to other nodes. The Slepian-Wolf theorem allows to derive some insights into the lossless compression of two or more

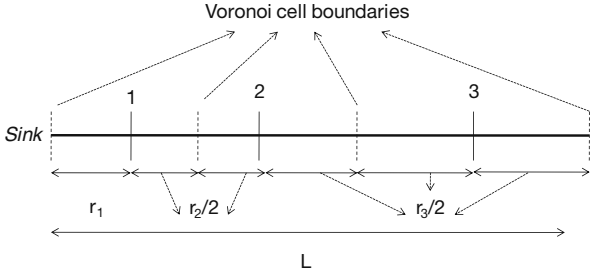


Fig. 9 Voronoi cells in case of linear placement

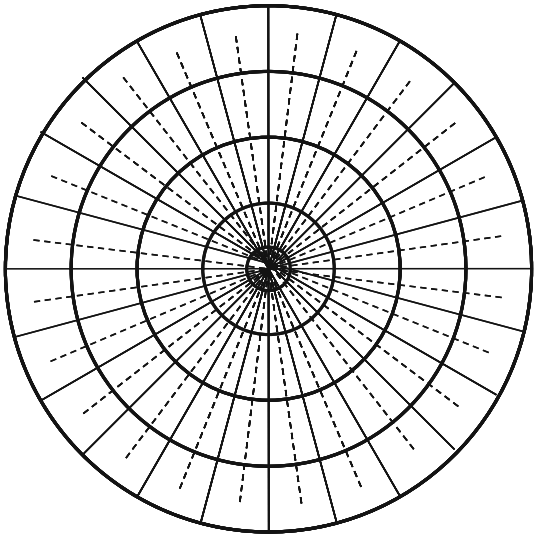


Fig. 10 Voronoi cells in case of sink placed at the center of a circular area

correlated data streams in that some correlated streams can be encoded separately, without any explicit communication performed between nodes, and the compressed data from all these encoders can be jointly decoded by a single decoder, provided that each node only knows the joint statistics. This theoretically guarantees that the output of the decoder is almost an exact reproduction of the correlated streams.

SW coding assumes two correlated random sources, say Z_1 and Z_2. Each source can code its data with a rate $R_j \geq H(Z_j)$, with $j \in [1, 2]$. If the two sources were able to communicate, they could send data with a total rate $R_1 + R_2 = H(Z_1, Z_2)$. Slepian and Wolf showed that, in the hypothesis that their individual rates are at least equal to the conditional entropies $R_1 \geq H(Z_1|Z_2)$ and $R_2 \geq H(Z_2|Z_1)$, two correlated sources can be coded with a rate equal to their joint entropy also if they cannot communicate. The SW derivation can be straightforwardly generalized to the case of k random sources. However, this is only a theoretical, although very significant, result and no indications about how such an ideal aggregation can be performed are given.

Due to this intrinsic feature of the SW coding, some evolutions of this technique applied to clustered systems have been proposed, e.g. [52]. In this work the authors study the major problems in applying Slepian-Wolf coding for data aggregation in cluster-based WSNs with an objective to optimize data compression so that the total amount of data in the whole network is minimized.

In [1], distortion due to possible loss of packets because of high packet loss rate, both in case of a simple relay node and an aggregator node is being considered. Two kinds of distortions are addressed: temporal and spatial as related to the consideration of the temporal or spatial correlation among samples. Accordingly the authors express the temporal distortion as

$$\bar{D}_n = 2\sigma^2 t_s \frac{1}{p_n} + \frac{2}{\gamma}\sigma^2 \frac{p_n e^{-\gamma t_s}}{1-(1-p_n)e^{-\gamma t_s}} - \frac{2\sigma^2}{\gamma} \tag{3}$$

being

- σ^2 the variance of the physical phenomenon modeled as a Gaussian random variable.
- t_s the sampling interval.
- p_n the probability that a sample received by node n is successfully delivered to the sink.
- γ the temporal correlation coefficient.

Analogously, the spatial distortion is expressed as

$$\hat{D}_l = \sum_{n=1}^{N}\sum_{m=0}^{\infty} p_n \cdot (1-p_n)^m \cdot 2 \int_0^{L/(2N)} \left(2\sigma^2 - 2\sigma^2 \cdot e^{-\left[\left(\sqrt{x^2+\theta_3^2 \cdot m^2 \cdot t_s^2}\right)/\theta_1\right]^{\theta_2}}\right) \mathrm{d}x \tag{4}$$

where

- L is the length of the considered line network.
- θ_1, θ_2 and θ_3 are correlation constants explicited in [1].

Interesting results are derived since authors find that there exist optimal densities of nodes such that the spatial distortion can be minimized. Also the distortion in case of aggregated data is, as expected, more critical than in case of simple relaying because the loss of some packets leads to loss of a significant portion of the network data. On the other hand, temporal distortion always increases and, in particular, in case of data aggregation since longer aggregated packets are sent and thus higher loss probability and higher temporal correlation are met.

In [13] a number of significant insights concerning the data aggregation process, which have not been discussed in the literature so far are derived. First, the conditions when aggregation is a costly process as compared to a no-aggregation approach are derived by considering a realistic scenario where processing costs related to aggregation of data are not neglected. Then, considering that aggregation should preserve the integrity of data, the entropy of the correlated data sent by sources is taken into account so as to reduce the amount of redundant data forwarded to the sink while performing an overall lossless process. The framework has been used to investigate the tradeoff between the increase in data aggregation required to reduce energy consumption, and the need to maximize information integrity.

The conditions when aggregation can be regarded as a non costly process with respect to no-aggregation are derived as a function of the power needed to transmit, code and keep on the circuitry, taking into account the number of downstream neighbors along the tree to the sink, as well as the maximum aggregation which can be applied, provided that the differential entropy of data is saved. This achieves the trade-off between need to increase the aggregated packets and preservation of information integrity.

A discussion on the design of suitable encoding strategies that leverage the correlation induced by cooperative coding in dense networks can be found in chapter "Cooperative Strategies in Dense Sensor Networks". In chapter "Cooperative Strategies in Dense Sensor Networks" it is described how to learn the structure of the data via queries and how to minimize the number of queries needed to obtain a lossless reconstruction of a source codeword. These coding approaches can be seen as solutions for propagating aggregated data in WSNs with attention to the preservation of the integrity on one side and the minimization of the network resources on the other side.

6.3.2 Preservation from Security Threats

Security is a major concern in wireless sensor networks. In fact, due to the deployment of sensors in open un-attended environments, and to the vulnerability of cheap network devices whose cryptographic keys can be kept by adversaries, false actions can be invoked, driven by unreliable data received by the sink(s). When considering the data aggregation process, security threats are even worsened. In fact, potential compromised aggregated data can imply that the sink receives wrong information about events happening in a portion of the entire network. This problem can be additionally worsened by the position of the aggregator node in the network. In fact, un-trusted high level nodes, i.e., nodes closer to the sink in a tree structure, cause more serious problems than compromised low level nodes, i.e., farther away nodes, because the aggregated result calculated by a high level device derives from the collection of measurements performed by a higher number of sensors. If we consider that an aggregator node collects the information sent by multiple sensors, processes them and sends again this information towards the sink(s), it is important that the final information collected by the sink is reliable. If this is not, two problems can arise

- un-necessary actions are activated by the sink(s). As an example, this could happen when the malicious nodes notify the sink(s) about alarms, when they should not, only in the perspective of consuming network energy resources, reduce network lifetime and increase future network vulnerability.
- necessary actions are not invoked by sink(s) because menaces are hidden by malicious nodes. As an example, this could happen when a security violation happens and malicious nodes are distributed in the network in the aim of hiding this threats, thus avoiding network protection.

In the literature aggregator nodes have been usually assumed to be honest, i.e., to send throughout the network reliable and secure data. However, in a realistic scenario it should be considered that aggregators can be malicious or very prone to physical attacks which could compromise the data reported by them, thus leading to false alarms. Accordingly, some techniques aimed at verifying the content of the data received by the sink should be devised.

Three types of security threats can be discussed when considering data aggregation:

- T_1: an aggregator node simply drops the information it is expected to relay to the sink.
- T_2: an aggregator node falsifies its readings so as to modify the aggregate value it is expected to relay to the sink.
- T_3: an aggregator node falsifies or modifies the aggregate generated by messages received by its children.

While the first threat can be easily identified, the second threat can be solved using well known approaches for fault tolerance. As an example in [36] an estimation algorithm to be performed at the fusion center for localizing events by combining the binary measurements received by many sensor nodes is presented. The main idea behind *Subtract on negative add on positive protocol (SNAP)* is to fuse the observations of all sensors (positive or negative) to efficiently construct a likelihood matrix by summing contributions of ± 1. In other words, sensors with positive observations add their contributions, while sensors with negative observations subtract theirs. The power of this algorithm lies in the simplicity. Also, by giving equal importance to the sensors that did and the ones that did not sense the event, they do not allow any particular sensor to change the estimation results and this is the basic reason for the fault tolerant behavior. Moreover, the algorithm is energy efficient since, for the construction of the likelihood matrix, only single bits need to be transmitted to the sink. The SNAP algorithm works in three major phases:

1. *Grid formation:* The entire area is divided into a grid. The number of cells is chosen as a trade-off between estimation accuracy and complexity. Each sensor node is associated with a cell (i, j) based on its position.
2. *Likelihood matrix (L) construction:* For each cell of the grid the likelihood of a source occurring in the particular cell is calculated based on the binary data received from each sensor node. To this purpose the Region of Coverage (ROC) is defined as a neighborhood of cells around (i, j). There are various ways that one can define the neighborhood and the optimal shape of the ROC depends on the signal propagation model. Given a uniform propagation model, one could assume that the ROC is the set of all cells whose centers are within a distance R from the center of cell (i, j). However, for computational efficiency, a square approximation is used. Each alarmed sensor adds a positive one $(+1)$ contribution to the elements of L that correspond to the cells inside its ROC. On the other hand, every non-alarmed sensor adds a negative one (-1) contribution to all elements of L that correspond to its ROC. Thus, the elements of the likelihood matrix are obtained by spatially superimposing and adding the ROC of all the sensor nodes in the field. Figure 11[6] shows an example of the resulting likelihood matrix after adding and subtracting the contributions of 8 sensors using SNAP.
3. *Maximization:* The maximum of the constructed likelihood matrix L, points to the estimated event location. If more than one element of the likelihood matrix

[6] Figure taken from [36].

has the same maximum value, the estimated event position is the centroid of the corresponding cell centers.

In Fig. 11 we observe 3 sensors located in the proximity of an event (identified by the white ellipse) to be notified. For each sensor the ROC is built as represented by the big squares associated to each point representing sensor node location. The area is divided in a grid and all ROC squares containing the event location are assumed to be alerted. The small squares in the alerted bigger squares will increase their indicator of $+1$ while the small non alerted squares will add -1 to the indicator. As a result the overall counters are shown in the figure. The event location will be identified with the small square with the higher indicator. Thus aggregation of the information collected by the sensors has been performed by equally weighting the contributions of the sensors. According to the case shown in Fig. 11, the event will be located at the baricenter between the centers of the two small squares having indicator $+2$.

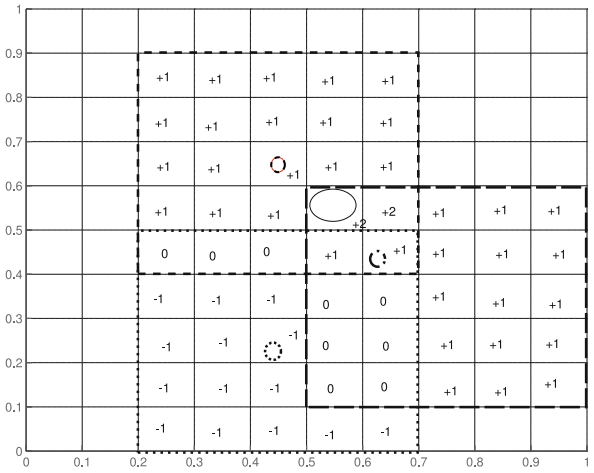


Fig. 11 Likelihood matrix construction resulting from SNAP in case of 3 sensor nodes, 2 of which are alarmed and are shown in *solid color*. The event is correctly localized at the baricenter between the two grid cells with the maximum value $+2$

The performance of SNAP was compared against two other estimators: a centralized Maximum Likelihood estimator (ML) proposed in [38] and a Centroid Estimator (CE) where the source position is estimated as the centroid of the alarmed sensor nodes positions. With three estimators, in the absence of faults, the performance of ML and SNAP is very similar. As the number of faults increases however, ML is very sensitive to sensor faults and looses accuracy continuously. For low sensor

faults, it starts loosing accuracy and for greater sensor faults its performance is worse than the CE. SNAP however, displays a fault tolerant behavior and looses very little in accuracy even when 25% of the considered 200 sensor nodes exhibit erroneous behavior.

CE is especially sensitive to the presence of false positives because of noise or misfunctioning due to overhearing. Since the algorithm essentially treats all alarmed sensor nodes with equal weight, it is especially sensitive to false positives that occur far away from the true event location. These can result in large errors when calculating the centroid of the alarmed sensor nodes' positions. On the other hand, ML is extremely sensitive to false negatives. Even a single faulty sensor node inside the neighborhood of the source can completely throw off the estimation results. This is a direct result of the construction of the likelihood matrix of ML.

Among the above discussed three security threats in data aggregation, however, the one which results more critical is the third, i.e., a modification of the data an aggregator node receives by its children. In fact, in this case no stable solutions have been yet devised. Some proposals, for example [43], use a key distribution approach to establish secure communication between aggregating sensors and the sink. Aggregation is intended in an incremental perspective, meaning that only the differential data is sent rather than the raw data. The key distribution approach randomly distributes the keys to all network nodes partitioned into groups. A key pool is distributed to each group and each node randomly selects a subset of these keys. Sensors use these keys to establish secure communication with group members. Other keys are used for allowing communication between neighboring grids. A discovery phase is initiated to let a node discover shared keys with other group members as well as with neighboring groups. Security level increases when going to higher levels of the hierarchy towards the sink, so that higher level nodes which are in charge of sending greater summaries of nodes readings, are more protected than low level nodes.

However, even if using this security mechanism, nothing guarantees that the data sent on a secure communication channel between the aggregator node and the sink is trusted and was not generated by a malicious aggregator node. In other words, this approach is thought to protect data against channel sniffing but not against aggregator threats.

Another work proposing a similar approach is the one presented in [27] where a self-configuring hierarchical clustered architecture for wireless sensor networks is presented. In this chapter security issues are dealt with in the perspective of a randomized data authentication procedure based on a unique ID and security code employed by sensor nodes and sink to randomly verify the trustfulness of the data received. However, here again the hypothesis of honest sensor nodes is assumed and threats are only identified with conditions where a security code is captured by an attacker.

Other approaches aimed at solving the third threat have been proposed in [6, 55, 42]. In [6] the authors deal with the problem of counteracting the, so-called, stealthy aggregate manipulation, that is a kind of security attack where the attacker wants to make the user accept false and erroneous aggregation results without

being recognized by the final user, i.e., the one who manipulates the information received by the sink(s). In [55] some techniques to allow the user to understand if the data received by the aggregator is deviating too much from an expected result, thus revealing possible attacks, is proposed. To this purpose cryptography is employed as well as a random sampling approach to let the user verify if what was sent by the aggregator has been compromised by an attacker action or not. Moreover, a methodology to let the aggregator prove its honesty in aggregating data is devised as well. In [42] a secure hop-by-hop data aggregation protocol is proposed. Differently from [55], which was one of the first works where malicious nodes were considered, here security relies on collaboration among network nodes. More specifically, in [55], by exploiting the divide-and-conquer paradigm, network nodes are probabilistically partitioned into multiple logical trees or groups. Accordingly, since a reduced number of nodes will be grouped under a logical tree routed at a high level aggregator node, the threats related to a compromised high level node will have a reduced impact. An aggregate packet is generated by each group and, once a user receives aggregates by all groups, it uses a bivariate multiple-outlier detection algorithm to figure out if suspicious packets were received. When a suspicious group is identified, it is forced to take part to an attestation process to prove its honesty. If a group fails in attesting its honesty, its packets are discarded by the user who, eventually, takes into consideration only packets received by verified groups. A similar problem is dealt with in [42]. In this chapter the focus is on use of multi-path networking paradigms for performing aggregation. Multipath, as discussed in previous sections, is more reliable in case of network failures with respect to tree approaches but still presents some problems of reliability in case of duplicate-sensitive aggregate metrics such as count and sum. To this purpose the well known synopsis diffusion [37] approach has been proposed. However, synopsis diffusion does not provide any support for security and, thus, in [42] an evolution of synopsis is presented. Here a subset of the network nodes provide an authentication code as well as data packets responding to a possible query. These authentication codes are propagated up to the sink and used to estimate the accuracy of the final result so as to filter and discard possible compromised packets.

In [40] the problem of validating aggregate information sent by sensors located on mobile vehicles is addressed. More specifically, authors focus on spoofing and bogus attacks related to injection of fake information into the network that are used to divert traffic. To cope with these problems especially in case of aggregated data sent by vehicles, a tamper-proof service is assumed for generating timestamps, random numbers and signing the records sent by nodes. Aggregator nodes are thus solicited to provide a proof that their aggregated record is not modified. For worth of validation, the aggregator sends the aggregate record as well as a record contributing to the aggregated one but chosen according to a random number. If this tamper-proof information fits into the aggregate record the packet is considered reliable. This mechanism should provide a deterrent to node's malicious behavior. However, here again tamper-proof elements are assumed which could be unrealistic in the majority of the scenarios.

Table 3 Summary of protocol solutions for data aggregation, classified in accordance to different taxonomies

		Description	Propagation	Protection
Layered	Link		[9, 50, 39, 18]	
	Network	[23, 31, 21, 54]	[18, 22, 28, 32, 34, 8, 56, 47, 20, 30, 19, 26, 5, 16, 7, 37, 11, 35]	[36, 38, 43, 27, 6, 55, 42, 51, 12]
	Application	[23, 31, 21, 54, 45]		[6, 40]
Functional	Routing	[23, 31]	[22, 28, 32, 34, 8, 56, 47, 20, 30, 19, 26, 5, 16, 7, 37, 11, 35]	[6, 27, 55, 42]
	Aggregation			[14, 52, 1, 36, 38, 43, 27, 51, 12]
	Representation	[21, 54, 45]		[36, 38, 43, 42, 40]
Performance	Energy	[23, 45]	[50, 39, 18, 22, 28, 56, 20, 30, 19, 26, 5, 16, 13]	[14, 52, 43, 42]
	Latency	[23, 31]	[9, 39, 18, 22, 28, 32, 34, 8, 30, 26, 5, 16]	
	Fault tolerance	[23, 31]	[32, 34, 8, 47, 7, 37, 11, 35]	[36, 38, 40]
	Security			[36, 38, 43, 27, 6, 55, 42]
	Accuracy	[45]	[13]	[14, 52, 1, 43, 27, 6, 55]
	Network load		[9, 51, 12]	

7 Conclusions

Data aggregation in wireless sensor networks is a promising approach for counteracting waste of resources caused by redundancy traveling throughout the network. In wireless sensor networks, due to the high density of deployed devices and the type of monitoring, the interest is typically in an overall view of the network status and not in each specific node's reading. More specifically, due to the spatial and temporal correlation among data collected by sensor devices, some useless information can travel into the network, thus causing spreading of both bandwidth and energy resources.

In this chapter we investigated on the use of data aggregation as a mean for counteracting this redundancy problems in sensor networks. We highlighted different approaches to taxonomy of data aggregation techniques, namely *layer-centric, ingredient-centric* and *performance-centric*. We then presented and discussed data aggregation by focusing on the *information-centric* approach which allows a detailed understanding of the impact of the data aggregation process in terms of information

description, propagation and preservation, and in the same time captures the main functional elements and protocols in the view of understanding the implementation and performance issues behind each solution. In Table 3 we reported the main solutions introduced in the literature by applying both the classical taxonomies appeared in the literature (rows of the table) and the various perspectives of the information-centric approach (columns).

As regards the description of the information, all the solutions discussed aim at facilitating WSN applications to query and collect the information on the basis of an associated semantic scheme. We pointed out how semantics can be used to achieve efficient data aggregation able to capture the meaning of the information that is aggregated. The propagation of the information is instead at the basis of aggregation functions that are tailored to the medium access, routing and scheduling schemes adopted in the network to propagate the information from sensors to sinks. Finally, the preservation of the information integrity can be accomplished by suitable aggregation methods that reduce the amount of information relayed towards the sink(s), and in the same time assure the preservation of the information itself; on the other side aggregation can also be accomplished with special care to the preservation of the information from security threats.

From this overview two main insights can be derived. On the one hand, a need for abandoning isolated solutions looking for a holistic view is needed. In this view we believe that the proposed information centric analysis intrinsically provides guidelines for approaching the data aggregation issues in WSNs. On the other hand, to satisfy this requirement, the various approaches for aggregation performed at different levels of the network architecture should be made compliant and interoperable so as to allow an improvement in network behavior under a cross-layer perspective.

References

1. U. G. Acer and A. A. Abouzeid. Poster abstract: reconstruction distortion in lossy wireless sensor networks. Proceedings of WiMesh, Reston, VA, September 2006.
2. K. Akkaya, M. Demirbas, and R. S. Aygun. The impact of data aggregation on the performance of wireless sensor networks. Wireless Communications and Mobile Computing, pp. 171–193, Vol. 8, No. 1, February 2008.
3. R. Baldoni, C. Marchetti, R. Virgillito, and R. Vitenberg. Content-based publish-subscribe over structured overlay networks. Proceedings of ICSCS, Columbus, OH, June 2005.
4. P. Bonnet, J. Gehrke, and P. Seshadri. Towards sensor database systems. Proceedings of MDM, Hong Kong, China, November 2001.
5. C. Borgelt and G. Rodriguez. FrIDA – a free intelligent data analysis toolbox. Proceedings of FUZZ-IEEE 2007, London, UK, July 2007.
6. H. Chan, A. Perrig, and B. Przydatek. SIA: secure information aggregation in sensor networks. Proceedings of ACM Sensys, Los Angeles, CA, November 2003.
7. J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation technique for sensor databases. Proceedings of ICDE, Boston, MA, April 2004.
8. M. Ding, X. Cheng, and G. Xue. Aggregation tree construction in sensor networks. Proceedings of IEEE VTC Fall, Orlando, FL, October 2003.
9. K. Fan, S. Liu, and P. Sinha. Structure-free data aggregation in sensor networks. IEEE Transactions on Mobile Computing, pp. 929–942, Vol. 6, No. 8, August 2007.

10. E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi. In-network aggregation techniques for wireless sensor networks: a survey. IEEE Wireless Communications, pp. 70–87, Vol. 11, No. 2, April 2007.
11. P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. Journal of Computer and System Sciences, pp. 124–130, Vol. 31, June 1985.
12. L. Galluccio, A. T. Campbell, and S. Palazzo. CONCERT: aggregation-based CONgestion Control for sEnsoR neTworks (poster abstract). Proceedings of ACM SenSys, San Diego, CA, November 2005.
13. L. Galluccio, S. Palazzo, and A. T. Campbell. Efficient data aggregation in wireless sensor networks: an entropy-driven analysis. Proceedings of IEEE PIMRC, Cannes, France, September 2008.
14. D. Ganesan, R. Cristescu, and B. Beferull-Lozano. Power efficient sensor placement and transmission structure for data gathering under distortion constraints. Proceedings of IEEE IPSN, Berkeley, USA, April 2004.
15. M. R. Garey and D. S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, New York, 1979.
16. C. Guestrin, P. Bodix, R. Thibaux, M. Paskin, and S. Madden. Distributed regression: an efficient framework for modeling sensor network data. Proceedings of IPSN, Berkeley, CA, April 2004.
17. D. L. Hall and J. Llinas. Handbook of Multisensor Data Fusion. CRC Press, Boca Raton, FL, 2001.
18. T. He, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. AIDA: adaptive application-independent data aggregation in wireless sensor networks. ACM Transactions on Embedded Computing Systems, pp. 426–457, Vol. 3, No. 2, May 2004.
19. W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. Proceedings of IEEE HICSS, Big Island, HA, January 2000.
20. W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. IEEE Transactions on Wireless Communications, pp. 660–670, Vol. 1, No. 4, October 2002.
21. MICROTOPSS http://www.microtopss.msrg.utoronto.ca
22. C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. Proceedings of ICDCS, Vienna, AU, July 2002.
23. C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. IEEE/ACM Transactions on Networking, pp. 2–16, Vol. 11, No. 1, February 2003.
24. L. Jia, G. Lin, G. Noubir, R. Rajaraman, and R. Sundaram. Universal approximations for TSP, Steiner tree, and set cover. Proceedings of STOC, Baltimore, MD, May 2005.
25. L. Jia, G. Noubir, and R. Sundaram. Group-independent spanning tree for data aggregation in dense sensor networks. Proceedings of IEEE DCOSS, San Francisco, CA, June 2006.
26. K. Kalpakis, K. Dasgupta, and P. Nainjoshi. Maximum lifetime data gathering and aggregation in wireless sensor networks. Proceedings of IEEE ICN, Atlanta, GA, August 2002.
27. M. Khan, B. Bhargava, S. Agarwal, L. Lilien, and Pankaj. Self configuring node clusters, data aggregation, and security in microsensor networks. Technical report, Purdue University, 2002.
28. B. Krishnamachari, D. Estrin, and S. Wicker. The impact of data aggregation in wireless sensor networks. Proceedings of ICDCS, Vienna, AU, July 2002.
29. J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. Proceedings of the American Mathematical Society, pp. 14–32, Vol. 7, No. 1, February 1956.
30. S. Lindsey and C. S. Raghavendra. PEGASIS: power-efficient gathering in sensor information systems. Proceedings of IEEE Aerospace Conference, Big Sky, MT, March 2002.
31. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a Tiny AG-gregation service for ad-hoc sensor networks. Proceedings of OSDI, San Francisco, CA, December 2002.

32. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The design of an acquisational query processor for sensor networks. Proceedings of SIGMOD, San Diego, CA, June 2003.
33. S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: an acquisitional query processing system for sensor networks. ACM Transactions on Database Systems, pp. 122–173, Vol. 30, No. 1, March 2005.
34. S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler. Supporting aggregate queries over ad-hoc wireless sensor networks. Proceedings of WMCSA, English Lake District, UK, June 2004.
35. A. Manjhi, S. Nath, and P. B. Gibbons. Tributaries and deltas: efficient and robust aggregation in sensor network streams. Proceedings of Sigmod, Baltimore, MD, June 2005.
36. M. P. Michaelides and C. G. Panayiotou. Subtract on negative add on positive (SNAP) estimation algorithm for sensor networks. Proceedings of IEEE ISSPIT, Cairo, Egypt, December 2007.
37. S. Nath, P. B. Gibbons, S. Seshan, and Z. Anderson. Synopsis diffusion for robust aggregation in sensor networks. Proceedings of ACM SenSys, Baltimore, MD, November 2004.
38. R. Niu and P. Varshney. Target location estimation in wireless sensor networks using binary data. Proceedings of ICASSP, Salt Lake City, UT, May 2001.
39. S. Pack, J. Choi, T. Kwon, and Y. Choi. Application aware data aggregation in wireless sensor networks. Proceedings of IEEE AWiN, St. Louis, MO, November 2005.
40. F. Picconi, N. Ravi, M. Gruteser, and L. Iftode. Probabilistic validation of aggregated data in vehicular ad hoc networks. Proceedings of IEEE VANET, Los Angeles, CA, March 2006.
41. R. C. Prim. Shortest connection networks and some generalizations. Bell System Technical Journal, pp. 14–19, Vol. 36, 1957.
42. S. Roy, S. Setia, and S. Jajodia. Attack-resilient hierarchical data aggregation in sensor networks. Proceedings of ACM SASN, Alexandria, VA, October 2006.
43. H. O. Sanli, S. Ozdemir, and H. Cam. SRDA: secure reference-based data aggregation protocol for wireless sensor networks. Proceedings of VTC Fall, Los Angeles, CA, September 2004.
44. A. Scaglione and S. Servetto. On the interdependence of routing and data compression in multi-hop sensor networks. Proceedings of ACM Mobicom, Atlanta, GA, September 2002.
45. M. A. Sharaf, J. Beaver, A. Labrinidis, and P. K. Chrysanthis. TiNA: a scheme for temporal coherency-aware in-network aggregation. Proceedings of ACM MobiDe, San Diego, CA, November 2003.
46. N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri. Medians and beyond: new aggregation techniques for sensor networks. Proceedings of ACM Sensys, Baltimore, MD, November 2004.
47. P. Skraba, Q. Fang, A. Nguyen, and L. Guibas. Sweeps over wireless sensor networks. Proceedings of IEEE IPSN, Nashville, TN, April 2006.
48. D. Slepian and J. K. Wolf. Noiseless coding of correlated information sources. IEEE Transactions on Information Theory, pp. 471–480, Vol. 19, January 1973.
49. I. Solis and K. Obraska. The impact of timing in data aggregation for sensor networks. Proceedings of IEEE ICC, Paris, France, June 2004.
50. M. C. Vuran and I. F. Akyildiz. Spatial correlation-based collaborative medium access control in wireless sensor networks. IEEE/ACM Transactions on Networking, pp. 316–329, Vol. 14, No. 2, April 2006.
51. C.-Y. Wan, S. B. Eisenman, and A. T. Campbell. CODA: congestion detection and avoidance in sensor networks. Proceedings of ACM SenSys, Los Angeles, CA, November 2003.
52. P. Wang, C. Li, and J. Zheng. Distributed data aggregation using clustered Slepian-Wolf coding in wireless sensor networks. Proceedings of IEEE ICC, Glasgow, UK, June 2007.
53. K. Whitehouse, F. Zhao, and J. Liu. Poster abstract: automatic programming with semantic streams. Proceedings of ACM Sensys, San Diego, CA, November 2005.
54. A. Wun, M. Petrovic, and H.-A. Jacobsen. A system for semantic data fusion in sensor networks. Proceedings of DEBS, Toronto, Canada, June 2007.

55. Y. Yang, X. Wang, S. Zhu, and G. Cao. SDAP: a secure hop-by-hop data aggregation protocol for sensor networks. Proceedings of ACM Mobihoc, Florence, Italy, May 2006.
56. W. Zhang and G. Cao. DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks. IEEE Transactions on Wireless Communications, pp. 1689–1701, Vol. 3, No. 5, September 2004.

Robust Data Dissemination for Wireless Sensor Networks in Hostile Environments

Jun Lu, Yi Pan, Satoshi Yamamoto, and Tatsuya Suda

Abstract An important research issue in large scale wireless sensor networks is to efficiently deliver data per user requests, which is referred to as data dissemination. This chapter studies data dissemination in wireless sensor networks deployed in hostile environments. In such type of networks, external catastrophic events can damage sensors at a large scale, which may cause data loss or disconnect data transmission paths. This chapter presents a novel data dissemination protocol, RObust dAta Dissemination (ROAD), which can sustain data dissemination when large scale sensor failures occur. In ROAD, events are differentiated based on types. For each type of event, data are replicated to a set of sensors forming a specific geographical trajectory. When facing catastrophic events, since sensors forming a trajectory are unlikely to be damaged simultaneously, surviving sensors can restore the trajectory and resume data dissemination. Simulation results show that ROAD is robust against large scale sensor failures while presenting low communication overhead, high reliability and short response time.

1 Introduction

Recent improvements in affordable and efficient integrated electronic devices have significantly advanced the technology of wireless sensors. Wireless sensors available today are equipped with sensing modules, radio transceivers, processing units, storage and possible actuators. Such sensors form a wireless sensor network (WSN) and collaborate with each other to perform various functionalities in different applications, such as environment surveillance, target tracking and disaster rescue. The utility of a sensor network in these applications derives primarily from the data it gathers. Thus, one of the key issues of a WSN application is to efficiently deliver data according to user requests in a large scale sensor network. This is referred to in the literature as data dissemination.

J. Lu (✉)
School of Information and Computer Sciences, University of California, Irvine, CA, USA
e-mail: lujun@ics.uci.edu

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_7,

While there has been extensive research (e.g., [3, 9, 11, 13, 17, 18, 20]) on data dissemination for WSNs, most of the existing work assumes a relatively stable deployment environment, in which large scale sensor failures are unlikely to occur. This chapter considers WSNs deployed in hostile environments, where robustness is recognized as an essential issue when network operations become highly challenging due to large scale sensor failures. While A5 studies multi-path routing in improving sensor network robustness in data transmission, this chapter focuses on another aspect – robustness in data dissemination. One example of robust data dissemination in a hostile environment is battlefield surveillance where a WSN is deployed to monitor enemy units including soldiers and vehicles. In this scenario, many allied soldiers are scattered in the field and each may request information on various enemy units at different times. The WSN should be able to deliver the requested information even when many sensors have been damaged by external events such as bomb explosions or fire blasts.

A novel data dissemination scheme will be presented for WSNs in hostile environments, RObust dAta Dissemination (ROAD). In ROAD, events are differentiated based on their types (e.g., events of wildfire or bomb explosions). A global hash function shared by sensors and sinks maps different types of events to different geographical trajectories (e.g., lines or curves) as to where data for the types of events should be stored. As randomly deployed sensors may not precisely reside on a trajectory, a trajectory is actually formed by a set of sensors with their locations best approximating the trajectory. Data for a certain type of event are replicated on the set of sensors forming the corresponding trajectory. Because a trajectory spans a relatively large range, the probability of all sensors forming the trajectory failing simultaneously is low. When a few sensors on a trajectory fail, other surviving sensors can repair the trajectory and restore data dissemination. By replicating data to sensors that form geographical trajectories, ROAD achieves the following advantages: (1) robustness against large scale sensor failures, (2) low communication overhead and short response time of data retrieval, and (3) high reliability under heavy query loads.

The rest of this chapter is organized as follows: Sect. 2 examines existing work on data dissemination in the Internet as well as WSNs. Section 3 describes ROAD in detail, followed by simulation evaluation in Sect. 4 and conclusion in Sect. 5.

2 Related Work

A spectrum of approaches has been attempted for data dissemination over the Internet, including constructing distributed indexing services in overlay networks. Examples of such services include the Internet Domain Name System (DNS), MERCURY [2] and the Distributed Hash Table (DHT) systems exemplified by Freenet [5], CAN [16] and Chord [19]. These approaches, however, assume an Internet environment in which two nodes can establish a logical connection over multiple physical links using transport protocols. Thus, they are not applicable to

the context of WSNs, in which a logical connection between two sensors may not be possible due to physical limitations on sensor capabilities or resources (e.g., sensors may not have enough memory or power to run complex network or transport layer protocols).

There have also been numerous studies on data dissemination in WSNs.

Intanagonwiwat et al. [11] proposed a scheme called Directed Diffusion to allow sinks to retrieve data of interest. In their scheme, each sink broadcasts a specific interest to the network and establishes a gradient to specify the forwarding direction toward itself. Each sensor matching the interest sends low-rate data, possibly along multiple paths, following the gradient toward the sink. The sink then reinforces one good path in order to draw higher quality (higher data rate) events.

Rumor routing [3] presents a simple solution for data dissemination at low cost. Both event notifications and queries are forwarded randomly through the network. Each sensor relaying event notification stores the route to the source of the event. A query is forwarded toward the source sensor when it reaches a sensor storing event information that matches the query. Since both data publishing and queries are randomly forwarded, rumor routing cannot guarantee the retrieval of all data requested by a sink.

Distributed Index for Multi-dimensional data (DIM), proposed in [13], describes a distributed indexing service for sensor networks to support multi-dimensional range queries. The sensor field is divided into zones, each owned by a single sensor. Events are published to and retrieved from the destination zone hashed from the event attributes. Using a geographic locality-preserving hash function that maps similar data to physical vicinity, DIM is able to efficiently support multi-dimensional range queries.

Ratnasamy et al. [17, 18] proposed Data-Centric Storage (DCS) for WSNs. DCS stores data in sensors in a manner dependent on the type of event. To publish data for an event, a source sensor hashes the type of the event into a geographical storage location using a Geographic Hash Table (GHT). It then employs geographic routing to publish data to the sensors graphically nearest the storage location. To retrieve data for a type of event, a sink uses the same GHT to obtain the storage location and sends a query to the storage location. DCS uses a perimeter refresh protocol to replicate stored data locally to improve robustness against sensor failures and mobility. Structured replication is utilized to balance the work load of data storage among sensors. Ghose et al. [9] proposed Resilient Data-Centric Storage (R-DCS) to improve the scalability and resilience of DCS to sensor failures by strategically replicating data.

In Two-Tier Data Dissemination (TTDD) [20], upon detecting an event, each sensor proactively builds a grid structure by storing forwarding information to sensors geographically nearest to the grid points, called data dissemination nodes. To request data, a sink floods a query within the scope of a grid square (a cell). When a data dissemination node with the requested data receives such a query, it forwards the query toward the source, following the grid structure. A sink issues a new query when it moves more than a cell size from its previous location. The forwarding of the new query stops at a data dissemination node that has been receiving the data.

The data dissemination node then forwards the data toward the new location of the sink. In this way, TTDD can minimize the jitter of received data, even when a sink is moving continuously.

Tiered Storage ARchitecture (TSAR), proposed in [6], addresses the problem of data dissemination in a heterogeneous sensor network composed of normal sensors and powerful proxy nodes. The basic idea of TSAR is to separate the storage of data and metadata (i.e., abstract information describing data). Data are stored locally at each sensor, and metadata are sent to a nearby proxy node, which communicates to other proxy nodes by external high-speed links. Interconnected proxy nodes construct a distributed index, called the Interval Skip Graph, for the metadata reported from all sensors. The Interval Skip Graph enables a sink to query and read data efficiently. By eliminating in-network indexing, TSAR is able to reduce energy consumption of sensors.

Fang et al. [7] presented a landmark-based data dissemination scheme for sensor networks in which location information is not available or expensive to obtain. With a set of pre-selected sensors as landmarks, the network is divided into tiles using the GLADER protocol [8]. To publish or retrieve data, each sensor hashes the content of the data to a destination tile and routes the data or query toward the destination tile using the GLADER protocol. The published data are replicated at intermediate tiles on the path to the destination tile in order to accelerate later data retrieval.

All of the above mentioned existing approaches for data dissemination in WSNs assume a relatively stable environment, although some of them can handle individual sensor failures by replicating data to several mirror sensors. However, none of these approaches explicitly address data dissemination in an environment where large scale sensor failures may destroy all the mirror sensors or disconnect the path of data retrieval.

3 RObust dAta Dissemination (ROAD)

3.1 Assumptions

In a real sensor network deployment, sensors may be able to identify the detected event through *in-network data processing*, which usually involves the collaboration of multiple nearby sensors. This chapter focuses on data dissemination and simply assumes that a single sensor, upon detection of an event, can process and identify the type of the event. Sensors are configured with a global hash function to map detected types of events to pre-defined trajectories as to where data for detected types of events should be stored.

This chapter assumes that sensors are static and location-aware. Although equipping a GPS device with each sensor is expensive and thus unrealistic, there are many sensor localization schemes (e.g., [1, 4, 15]) that allow sensors to obtain their own locations without GPS. It is also assumed that sensors can obtain their neighbors' locations through one-hop communication.

Sensors are assumed to have certain processing capabilities to run ROAD protocol. The computation in ROAD is quite simple and affordable for small sensors with limited processing capabilities.

3.2 Scheme Description

The design of ROAD applied the widely-accepted concept of data-centric storage [17]. In data-centric storage, data are stored within the network based on their types. Data for different types of events are stored to sensors at different locations. To retrieve the data of interest, queries are forwarded to the corresponding locations. ROAD further develops this idea by distributing data for a certain type of event to a continuous geographic trajectory instead of isolated locations. Data for different types of events are stored to sensors approximating different trajectories. To retrieve the event of interest, a sink issues a query to reach the corresponding trajectory following the shortest path. An overview of ROAD is shown in Fig. 1 to disseminate data for two types of events.

Although ROAD is designed to accommodate various forms of trajectories, the current implementation uses straight lines as trajectories because straight lines are easy to compute, represent, and maintain. The end of this section will discuss how to extend ROAD to generic trajectories.

ROAD has three major component processes: data publishing, data retrieval and trajectory maintenance.

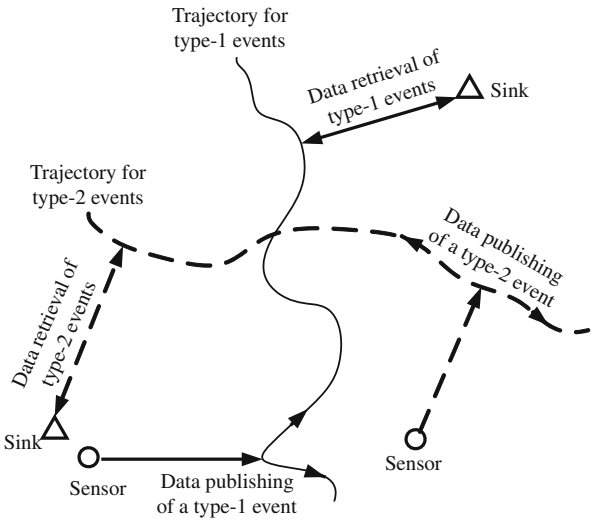


Fig. 1 An overview of data publishing and retrieval through continuous trajectories in ROAD

3.2.1 Data Publishing

In order to publish data to sensors approximating a trajectory, data are forwarded in two phases: forwarding data to a sensor on the trajectory and propagating data to sensors along the trajectory. In phase I, a sensor detecting an event forwards the data to reach the trajectory. The destination of data forwarding in phase I is the trajectory, or more precisely, *any* sensor on the trajectory. In phase II, data are forwarded to propagate along the path that best approximates the trajectory. The destinations of data forwarding are *all* the sensors that form such a path. Note that existing geographic routing algorithms (e.g., [12]) cannot be applied here as they forward data to a particular location as the destination.

Phase I – Forwarding Data to a Sensor on the Trajectory

A source sensor identifies the event type upon detection of an event. It then uses a global hash function to map the event type to a trajectory. Different event types are hashed to different trajectories such that data for different event types are stored separately. The event type and data are included in a `DATA` message sent by the source sensor. A `DATA` message can be forwarded in two modes: the greedy mode and the hole mode.

To forward a `DATA` message to reach a trajectory, a sensor calculates its own distance to the trajectory as the distance between its own location and the perpendicular foot (i.e., the intersecting location of the perpendicular line from the sensor to the trajectory). Since it also knows its neighbors' locations, the sensor can calculate their distance to the trajectory and decide whether a neighbor is closer to the trajectory than itself. If closer neighbors exist, the sensor forwards the message in the greedy mode to the neighbor with the shortest distance to the trajectory.

In phase I, it is possible that a sensor may not find any neighbor closer to the trajectory than itself. In that case, a hole is detected. As some existing geographic routing algorithms (e.g., [12]), ROAD applies the right hand rule to circumvent the hole, but with a different termination condition. The sensor that initiates the hole mode sets a hole bit (to notify other sensors of the hole mode) and includes its own location (i.e., the location of the sensor that started the hole mode) in the `DATA` message. It then sends the `DATA` message to the neighbor chosen using the right hand rule. The neighbor receiving the `DATA` message checks whether its own location is closer to the trajectory than the location of the sensor that started the hole mode. If its location is closer to the trajectory, it terminates the hole mode by resetting the hole bit and resumes forwarding of the message in the greedy mode. Otherwise, it continues forwarding the `DATA` message using the right hand rule. A TTL (Time-To-Live) value is used in the hole mode to stop message forwarding when the trajectory is unreachable.

If a sensor has a neighbor located at the other side of the trajectory, it forwards the `DATA` message to the neighbor closest to the trajectory. The neighbor receiving the `DATA` message then ends Phase I.

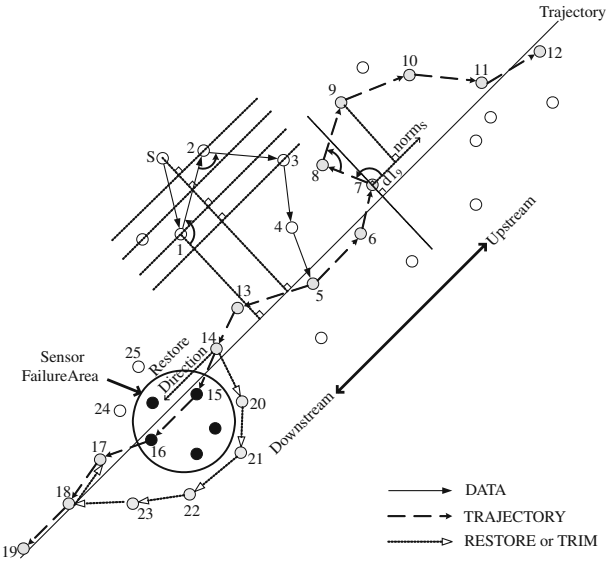


Fig. 2 An illustration of the major processes of ROAD – data publishing, data retrieval and trajectory maintenance

A complete scenario of phase I data publishing is shown in Fig. 2. Sensor *S* has data to publish and chooses sensor 1 as the next hop, which is the closest to the trajectory among the neighbors of *S*. Sensor 1 faces a hole and uses the right-hand rule to find sensor 2 as the next hop. Sensor 1 then sets the hole bit, stores its own location as the location of the sensor that started the hole mode, and sends the `DATA` message to sensor 2. Sensor 2 realizes it is farther from the trajectory than the sensor that started the hole mode (i.e., sensor 1) and finds sensor 3 as the next hop using the right-hand rule. Sensor 3 is closer to the trajectory than the sensor that started the hole mode (i.e., sensor 1). It resumes forwarding of the message in the greedy mode by resetting the hole bit, and forwards the `DATA` message to sensor 4. Sensor 4 finds the neighbor sensor 5 at the other side of the trajectory and forwards the `DATA` message to the neighbor closest to the trajectory, which is sensor 5. Sensor 5 stops the forwarding of the `DATA` message and ends phase I.

Phase II – Propagating Data to Sensors Along the Trajectory

A sensor that ends phase I becomes the sensor that initiates phase II. To propagate new data toward both ends of the trajectory, two `TRAJECTORY` messages are sent by the initial sensor, one for each forwarding direction. Each `TRAJECTORY` message carries the type of the event, the data, and one bit representing the forwarding direction. `TRAJECTORY` messages can also be forwarded in the greedy mode or in the hole mode.

A `TRAJECTORY` message is forwarded such that (a) the message advances in the forwarding direction along the trajectory and (b) the message is forwarded by a

path that best approximates the trajectory. To achieve the above two goals, a sensor forwarding a TRAJECTORY message evaluates each of its neighbors based on the following two measures: the relative progress of the neighbor along the forwarding direction and the deviation of the neighbor from the trajectory. Specifically, it first calculates the direction of the TRAJECTORY message as $norm_s$. For example, for a given trajectory $y = ax + b$, $norm_s$ is $\langle 1, a\rangle$ for one direction (e.g., for the direction where the direction bit is 0) and $\langle -1, -a\rangle$ for the other direction (e.g., for the direction where the direction bit is 1). For each neighbor, the sensor calculates the relative progress of the neighbor as the projection of the vector from itself to the neighbor onto $norm_s$. If the projection is positive, in other words, if the neighbor makes progress along the forwarding direction, the neighbor is regarded as a candidate to forward the TRAJECTORY message. For each candidate sensor, the sensor calculates its deviation from the trajectory as the distance to the trajectory. Once candidate sensors are identified, and deviation is calculated for each candidate sensor, the sensor evaluates each candidate based on a weight calculated by the following equation,

$$w_i = \alpha \times d1_i - d2_i, \tag{1}$$

where α ($\alpha \geq 0$) is a system parameter, $d1_i$ denotes the progress of candidate sensor i relative to the current sensor along $norm_s$, and $d2_i$ represents the deviation of sensor i from the trajectory. The neighbor with the larger w_i represents a better approximation to the trajectory, since it must be closer to the trajectory or make greater progress along the trajectory according to Eq. (1). Therefore, the neighbor with the largest w_i is chosen as the next hop. Ties are broken by choosing the neighbor with a larger sensor ID.

The value of α provides a choice for the relative importance of $d1_i$ and $d2_i$ in weight calculation. Specifically, a larger α accentuates the relative importance of $d1_i$, and thus gives higher priority to sensors with significant progress toward the forwarding direction. To prevent too much deviation of the forwarding path from the trajectory, α is adjusted to 0 if the current sensor is more than a Communication Range (CR) away from the trajectory. This adjustment quickly "pulls" the forwarding path back to the trajectory when it deviates too far.

If all of the neighbors have negative projections onto $norm_s$, a sensor may not find any next hop candidate. In this case, a hole is detected. The sensor initiates the hole mode by setting the hole bit, and storing its own location (i.e., the location of the sensor that started the hole mode) in the TRAJECTORY message. Similar to phase I, it then uses the right-hand rule to choose the next hop, starting from the direction of $norm_s$. However, the termination condition of the hole mode in phase II is different. Since the hole is detected when a sensor that started the hole mode cannot find a neighbor with positive progress toward the forwarding direction $norm_s$, a sensor terminates the hole mode if its own location has a positive projection relative to the location of the sensor that started the hole mode; otherwise, it continues forwarding in the hole mode according to the right hand rule.

The propagation of the TRAJECTORY messages stops at the end of the trajectory.

Figure 2 depicts the process of forwarding a TRAJECTORY message. To forward the TRAJECTORY message toward the upstream direction, sensor 5 that initiates phase II calculates the weight for the two next hop candidates (i.e., sensors 4 and 6) and chooses sensor 6 with the larger weight as the next hop. Sensor 6, in turn, relays the message to sensor 7, which detects a hole. Sensor 7 sets the hole bit, stores its own location in the TRAJECTORY message and chooses sensor 8 as the next hop using the right-hand rule. The message is relayed in hole mode from sensor 8 to sensor 9, which stops the hole circumvention and resumes greedy forwarding. Afterwards, the message is forwarded in greedy mode until it stops at sensor 12, which is located at the end of the trajectory. Note that the TRAJECTORY message toward the downstream direction is forwarded in the same way, except for a different forwarding direction.

When forwarding a TRAJECTORY message, sensors store the data and record the upstream and downstream neighbors on the trajectory. These sensors are called *storage sensors*. The relationship between neighboring storage sensors is maintained through periodic ALIVE messages. All one-hop neighbors of these storage sensors cache the types of events stored at the storage sensors. These neighbors are called *caching sensors*, which only store one byte of the type of event for each nearby trajectory. Since the maximum distance between two neighboring storage sensors is CR, a later DATA message is guaranteed to encounter a caching sensor or a storage sensor when crossing the trajectory. When the DATA message reaches a caching sensor, it will be directed to a nearby storage sensor. A storage sensor receiving a DATA message then propagates the data along the established trajectory using the recorded neighbor information. As a result, later data for the same type of event are merged to the existing trajectory.

3.2.2 Data Retrieval

Data retrieval forwards a query from a sink to the corresponding trajectory and sends a reply including stored data back to the sink.

A sink applies a global hash function to identify a trajectory. It then issues a QUERY message including the sink's location and events of interest through a nearby sensor. A QUERY message is forwarded in the same way as phase I of data publishing, which is described in Sect. 3.2.1. For the same reason, if an established trajectory exists at the expected position for the queried event type, the QUERY message always encounters a caching sensor or a storage sensor. When the message reaches a caching sensor, it will be directed to a nearby storage sensor. The storage sensor then issues a REPLY message with the data of the queried events to the sink through generic geographic routing (e.g., [12]).

3.2.3 Trajectory Maintenance

Trajectory maintenance provides robustness against sensor failures.

A storage sensor detects the failure of a neighboring storage sensor when the ALIVE beacon from the neighbor is absent for a certain period. Upon detecting the

failure, the storage sensor starts a random back-off timer to avoid transmitting simultaneous repair messages from different storage sensors. When a storage sensor's timer expires, the storage sensor sends a RESTORE message toward the direction of the failed neighboring storage sensor. The RESTORE message includes the event type, data, forwarding direction and location of the current storage sensor.

The RESTORE message is forwarded along the trajectory in the same manner as phase II of data publishing, which is described in Sect. 3.2.1. The intermediate sensors forwarding a RESTORE message become storage sensors, which store the data and record the upstream and downstream neighbors. A RESTORE message stops at a storage sensor that already has a neighboring storage sensor in the forwarding direction or at the end of the trajectory. As depicted in Fig. 2, a catastrophic event caused the failures of storage sensors 15 and 16. The RESTORE message from sensor 14 is forwarded in the downstream direction until it reaches storage sensor 18, which already has a downstream neighbor 19.

However, it is not guaranteed that the trajectory is repaired exactly where it breaks. For example, in Fig. 2, the RESTORE message hits sensor 18 instead of sensor 17, causing sensor 18 to have two upstream neighbors, sensors 23 and 17. In order to eliminate the redundant old branch, the storage sensor terminating the RESTORE message sends a TRIM message to the old neighbor in the opposite direction of the RESTORE message. The TRIM message is forwarded along the old branch until there are no more sensors on the old branch. All sensors receiving the TRIM message resign from the trajectory by clearing their information on the event type. As shown in Fig. 2, sensor 18 sends a TRIM message to sensor 17, causing sensor 17 to delete all the stored information for the event type and to cancel the back-off timer for the RESTORE message.

3.3 Extensions

3.3.1 Double-Sided Hole Circumvention

When a hole is detected during phase II of data publishing or trajectory maintenance, part of the resultant trajectory may deviate far from the expected position. When a query tries to reach such a trajectory, it may not be able to hit the trajectory when crossing the expected position of the trajectory. In this case, the query fails to find the data. This case is rare since it only happens when the formed trajectory is concave and there is a path that crosses the expected position of the trajectory and protrudes into the concave area. To avoid this problem, we propose double-sided hole circumvention in which a trajectory is formed or restored along both sides of the hole. Obviously, no query can reach the expected position of the trajectory without hitting a storage sensor since the restored trajectory encloses the expected position for the trajectory. Double-sided hole circumvention incurs more communication overhead to circumvent a hole than single-sided hole circumvention previously described. For example, in Fig. 2, while sensor 14 sends a RESTORE message in downstream direction to restore the trajectory, sensor 17 also issues a RESTORE message in upstream

direction to restore the trajectory through sensors 24 and 25 to sensor 14. Back-off timers or TRIM messages are not needed in the double-sided hole circumvention.

3.3.2 ROAD for Generic Trajectories

The data publishing and retrieving algorithms in ROAD can be easily extended to a trajectory of a simple curve without crossing to itself. In phase I of data publishing, distance from a sensor to a generic trajectory can be calculated as the radius of the smallest circle centered at the sensor and tangent to the curve. The neighbor with the shortest distance to the curve is chosen as the next hop. The same right hand rule is applied: a hole is detected when no neighbor with closer distance to the curve exists, and a DATA message in the hole mode is forwarded according to the right-hand rule until it reaches a sensor closer to the curve than the sensor that started the hole mode. In phase II of data publishing, in addition to the distance to the curve, the relative progress of a neighbor along the curve can be calculated as the length of the curve between the tangent points of the current sensor and the neighbor. The progress is positive if the tangent point of the neighbor is toward the advancement direction of the current sensor, and negative otherwise. Together with the distance to a curve, the weight of each next hop candidate can be calculated using Eq. (1). The next hop candidate with the largest weight is chosen as the next hop. The same right hand rule can be applied: a hole is detected when no neighbor with positive progress exists, and a TRAJECTORY message in the hole mode is forwarded according to the right-hand rule until it reaches a sensor with positive progress relative to the sensor that started the hole mode. Data retrieval and trajectory maintenance can also be extended since they also follow phase I or phase II algorithms.

Note that a similar idea to the above forwarding algorithm was discussed in [14], although with different hole circumvention mode.

3.3.3 Time-Based Load Balancing

One possible criticism of ROAD and many other existing schemes (e.g., [13] and [17]) is the unbalanced load of data storage. Specifically, sensors near the predefined storage places have a greater chance of becoming storing and distributing data, and thus, they consume more energy. Our solution to this issue is time-based load balancing – both the time and type of event are used as hash keys to identify data storage places. For ROAD, in different time slots, data for the same type of event are stored to different trajectories. Correspondingly, queries for a certain type of event are also associated with time – for the same type of event happening in different time slots, queries are sent to different trajectories. Each type of event has a timeout period, indicating how long the data should be kept. By carefully designing the hashing function of time and event type, the load of data dissemination can be well distributed among sensors.

4 Simulation Evaluation

The ROAD implementation evaluated here is based on the MIT uAMPS NS-2 extension [10]. The ROAD extensions (i.e., double-sided hole circumvention, generic trajectories and time-based load balancing) are not evaluated and will be part of the future work. In the simulation, ROAD is compared with data-centric storage denoted by DCS ($d = 0, 1, 2$) [17, 18], where d means the depth of the structured replication hierarchy. In structured replication, the entire area is divided into 4^d sub-regions, each containing a storage location for a single type of event. These storage locations are organized into a tree structure of height $(d + 1)$. DCS ($d = 0$) refers to the original DCS design with no structured replication (i.e., only one storage location for each event type). With structured replication, a sensor detecting an event stores the data for the event at the closest storage location for the event type. A query is first routed to the root of the tree and then forwarded to leaves following the tree hierarchy in a recursive way. Query replies are transmitted following the same path as queries, but in the reverse direction. Structured replication distributes data storage over sensors at different locations, and thus improves the scalability. DCS applies a perimeter refresh protocol to restore data storage from local replications when sensor failures occur.

In the simulation, 1600 sensors are uniformly deployed in a square area of 200 m×200 m. Fifty sensors are randomly chosen as source sensors and each publishes data for a random type of event. Data publishing is modeled as a Poisson process at the arrival rate of 1 data publishing per 30 ss. Afterwards, 400 randomly chosen sensors each issue a query to retrieve data for a random type of event. The queries are also characterized as a Poisson process at the arrival rate of 1 query per second if not otherwise specified. When a query reaches a storage sensor, all the data for the queried event type are sent to the sink in one message. Four possible event types are simulated, each hashed to one of the four trajectories crossing the center of the area. Two of the trajectories are diagonal lines and the other two are parallel to x and y axes, respectively. The data size for one event is fixed at 100 bytes. α in Eq. (1) is configured as 1.0.

ROAD is compared with DCS in terms of communication overhead, response time, reliability and robustness. Note that for either scheme, the communication overhead does not include the overhead to initialize the network such as the energy consumed to exchange location information among one-hop neighbors.

4.1 Communication Overhead

Figure 3 shows the average communication overhead per data publishing as measured in *byte*×*hop*. We can see that ROAD incurs more communication overhead to publish data than DCS protocols. This can be easily understood since ROAD publishes data to form a trajectory while DCS protocols only store data to a single location. Furthermore, increasing the depth d of the replication hierarchy reduces

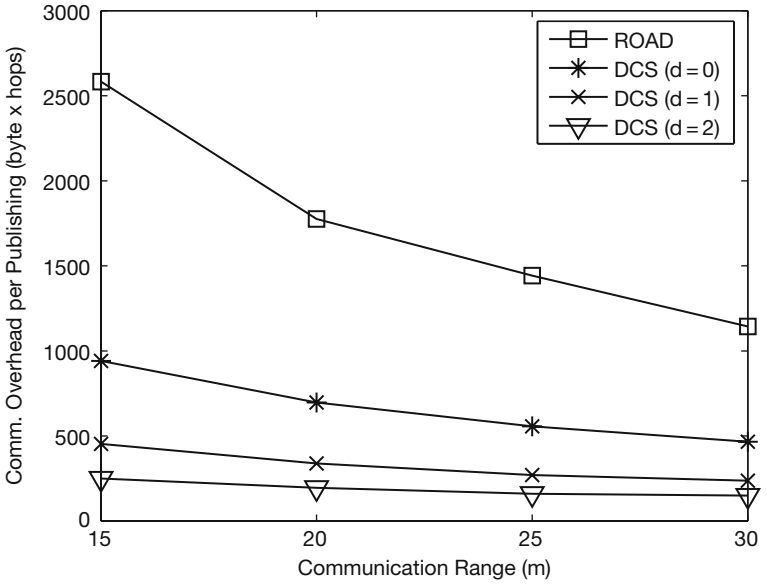


Fig. 3 Communication overhead per publishing

the cost of data publishing in DCS. This is because the distance between a sink and the closest mirror sensor becomes smaller when d is larger.

Figure 4 presents the average communication overhead per query as measured in $byte \times hop$. ROAD has the minimal communication overhead per query. In ROAD, each sink only needs to reach a nearby storage sensor on the trajectory to retrieve data, while in DCS a sink needs to reach all storage locations. Since queries and replies need to follow the replication structure recursively, the cost of query grows with the depth d of the replication structure as shown in Fig. 4. As expected, in all the schemes, the communication overhead of both data publishing and query decreases with the increase of CR because of less communication hops.

Based on the results in Figs. 3 and 4, the total communication overhead for both data publishing and queries can be derived as shown in Fig. 5. With 50 events published and CR of 15 m, ROAD outperforms DCS protocols when the number of queries is at least 20. Although not shown, similar results can also be obtained for larger CRs. For example, when CR is 30 m, ROAD outperforms DCS when there are more than 22 queries. This implies that ROAD is efficient for applications with many sinks retrieving data. The examples of such applications include battlefield surveillance and disaster rescue missions, in which many soldiers or rescue team members are interested in various types of events. For applications with few sinks requesting data, ROAD incurs more communication overhead than DCS protocols because most of the communication overhead in such applications is caused by data publishing.

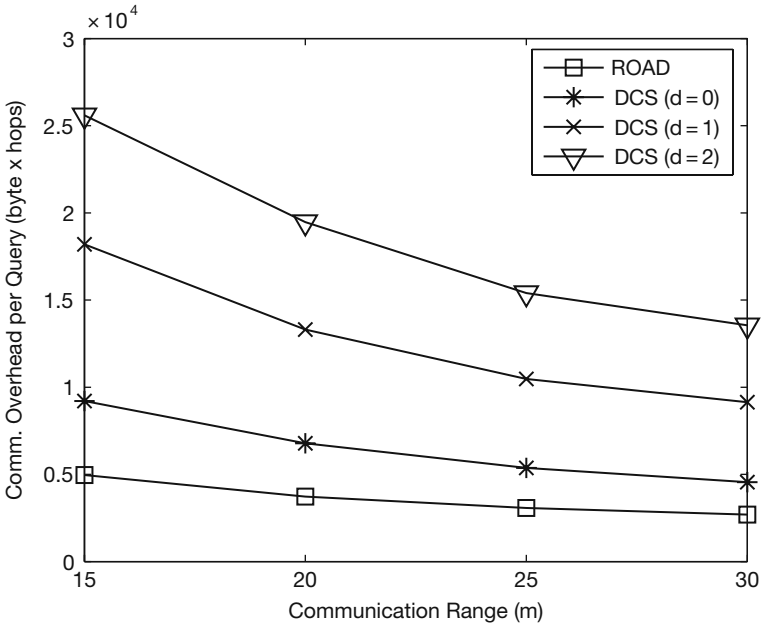


Fig. 4 Communication overhead per query

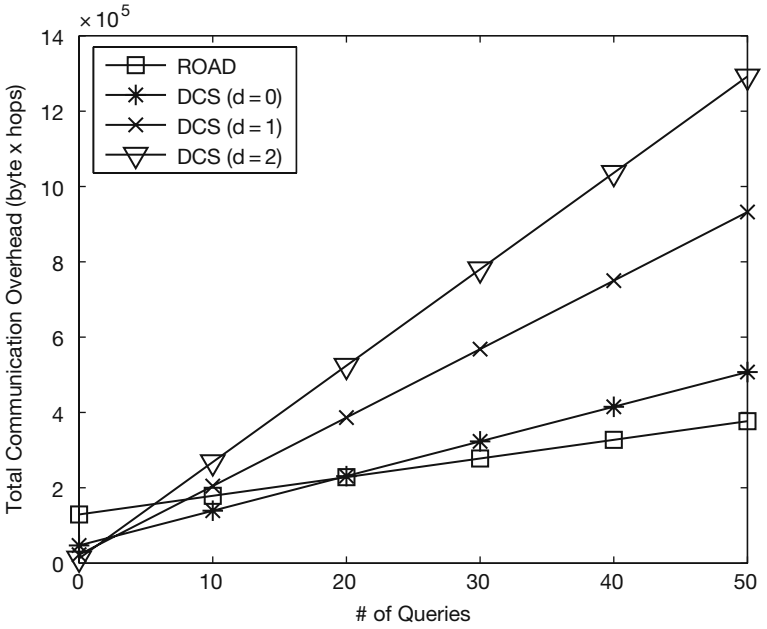


Fig. 5 Total communication overhead including data publishing and queries (50 events)

4.2 Response Time

For sensor network applications in hostile environments, the system response time is critical. For example, based on the retrieved data, a solider in a battlefield may need to take action immediately in order to avoid potential dangers. The response time is measured as the latency between the time when a query is issued and the time when a reply for the query is received. Figure 6 compares the response times for all protocols. ROAD has a response time about 50% faster than the best value among all DCS protocols, DCS ($d = 0$). This is because in ROAD, a sink can access a nearby sensor on the trajectory to retrieve data, while in DCS a sink needs to reach all possible mirror storage locations following the replication structure.

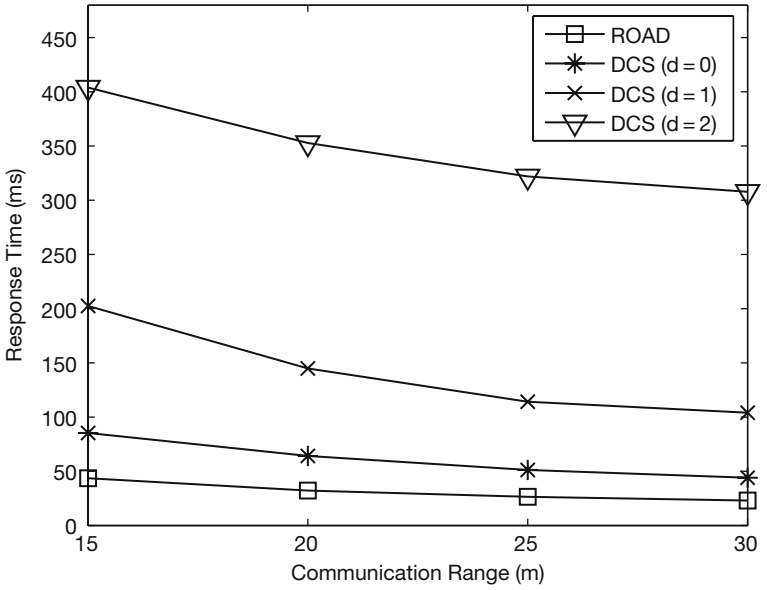


Fig. 6 Response time over various communication ranges

4.3 Reliability of Data Retrieval

The reliability of data retrieval is also important. The reliability performance can be evaluated using the query success rate under various query loads. In the simulation, the query success rate is measured as the ratio of the amount of data retrieved to the amount of published data that should be retrieved. For example, if there are 100 type-1 events published and a query for type-1 events retrieves the data for 95 type-1 events, the query success rate is calculated as 95%.

In Fig. 7, the query arrival rate is varied from 1 query per second to 200 queries per second to simulate a system under different query loads. Under the light load of

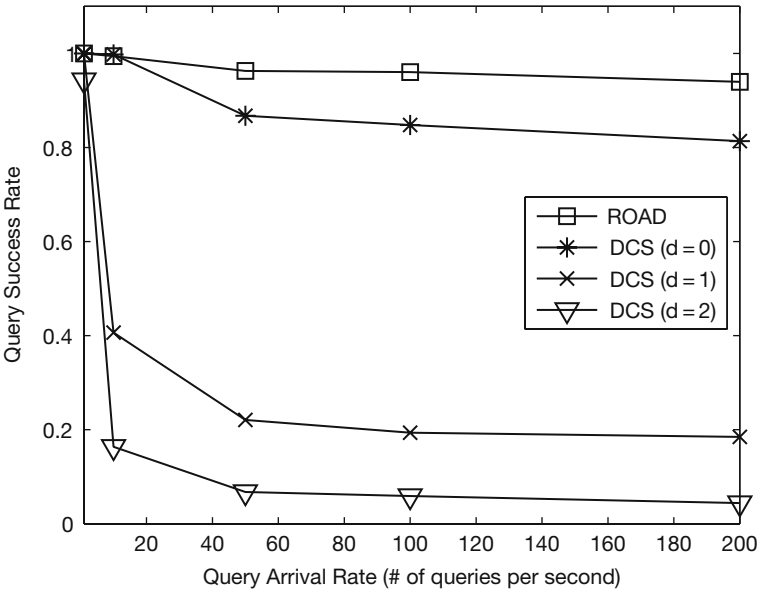


Fig. 7 Reliability of data retrieval as evaluated by query success rate over query arrival rates

1 query per second, all protocols perform well and deliver more than 94% of data. However, the query success rate of DCS ($d = 1$) and DCS ($d = 2$) suffers greatly when the query arrival rate reaches 50 queries per second. DCS ($d = 0$) performs best among the DCS protocols, delivering about 81% of data at the maximal query arrival rate. Again, ROAD outperforms DCS protocols by providing a success rate of about 94% at the maximal query arrival rate. The reason for data loss is network congestion caused by numerous queries and replies that are transmitted simultaneously in the network. In DCS ($d = 1$) and DCS ($d = 2$), congestion happens when a large number of queries and replies are forwarded following the replication hierarchy. Congestion is alleviated since there is no replication hierarchy in DCS ($d = 0$). In ROAD, the load of the queries is distributed over different storage sensors because each sink can retrieve data from a nearby sensor on the trajectory. As a result, the chance of congestion is reduced. The high reliability of ROAD favors applications where queries are frequently issued by many sinks.

4.4 Robustness Against Large Scale Sensor Failures

In order to evaluate the performance of ROAD and DCS in hostile environments, large scale sensor failures were simulated by randomly generating holes in the field. A hole is a circular area within which all sensors are regarded to have failed. Two large scale sensor failure scenarios were simulated.

In the first scenario, data for 50 events were published by 50 randomly chosen source sensors. After that, 10 holes were randomly generated with various sizes.

After a certain restoration period, queries were sent from 400 randomly chosen sensors each for a random type of event.

Figure 8 demonstrates the query success rate with various mean hole radii. ROAD has a higher query success rate than DCS protocols, especially for large holes. For example, with the mean hole radius of 32.5 m, ROAD provides about a 71% query success rate while the best DCS protocols can only deliver about 24% of data. Figures 9 and 10 confirm that ROAD has lower communication overhead and latency per query than DCS protocols under severe environmental conditions. Note that the communication overhead and latency are only counted for successful queries (i.e., queries that retrieve some data from the network).

The second scenario fixed the mean hole radius at 32.5 m but varied the number of holes. As shown in Figs. 11, 12, and 13, similar results were obtained that ROAD is more robust and has lower communication overhead and latency per query than DCS protocols under severe environmental conditions.

When large scale sensor failures occur, query failures can be caused by (1) loss of published data due to the failures of storage sensors or (2) unsuccessful routing incurred by damage of intermediate sensors. By publishing and retrieving data through a trajectory, ROAD relieves the effects of both causes. Since the chance of all sensors on a trajectory failing simultaneously is low, the data published onto a trajectory are unlikely to be destroyed. In contrast, DCS protocols deal with sensor failures through a perimeter refresh protocol – when a storage sensor fails, other storage sensors on the perimeter will restore the perimeter with new sensors. The

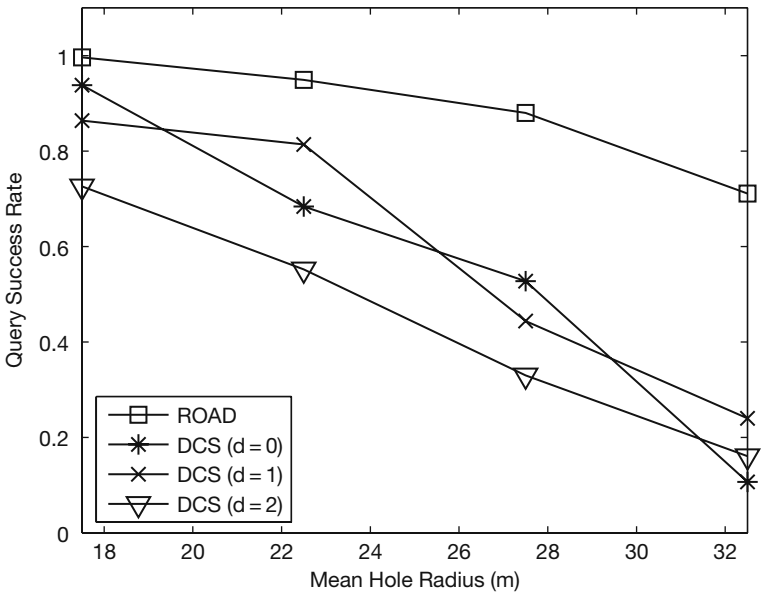


Fig. 8 Robustness against large scale sensor failures: query success rate over a range of mean hole radius (hole # = 10)

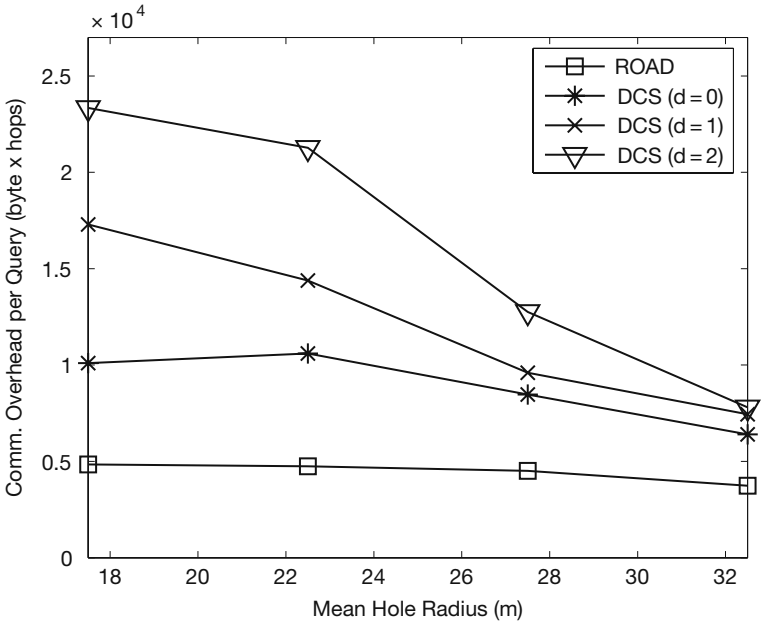


Fig. 9 Robustness against large scale sensor failures: communication overhead per query over a range of mean hole radius (hole # = 10)

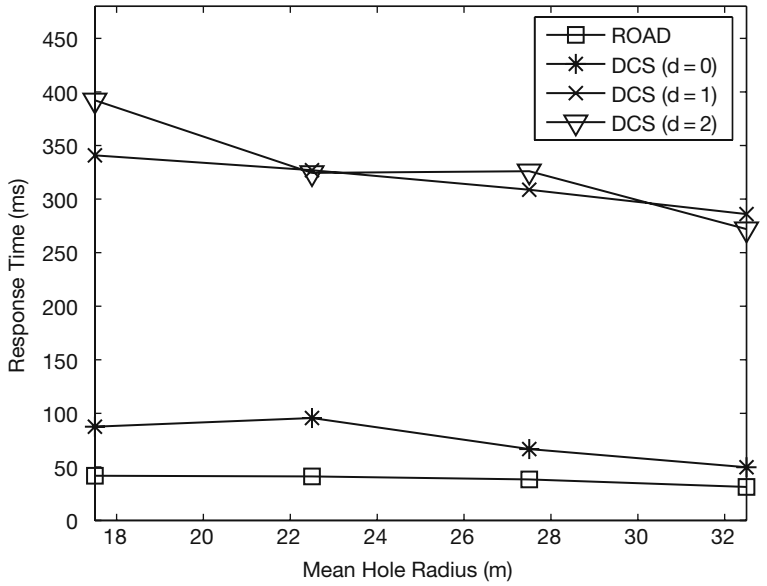


Fig. 10 Robustness against large scale sensor failures: response time over a range of mean hole radius (hole # = 10)

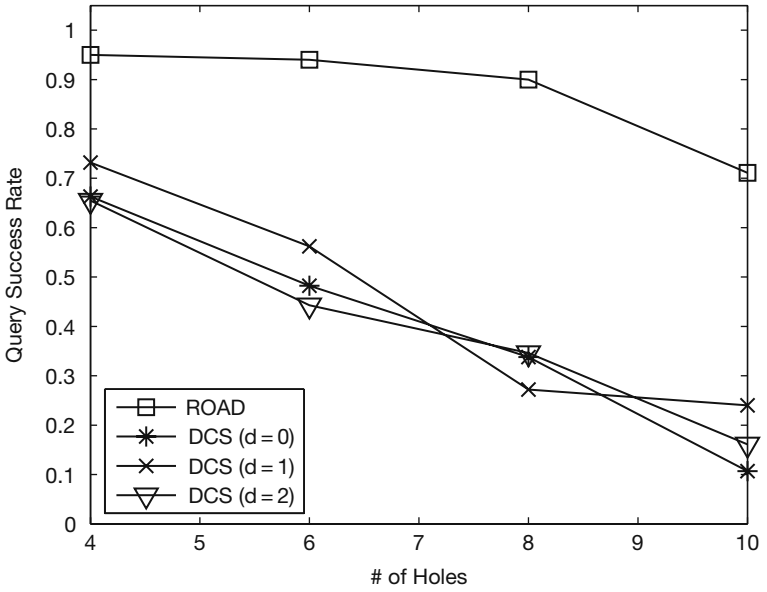


Fig. 11 Robustness against large scale sensor failures: query success rate over a range of hole # (mean hole radius = 32.5 m)

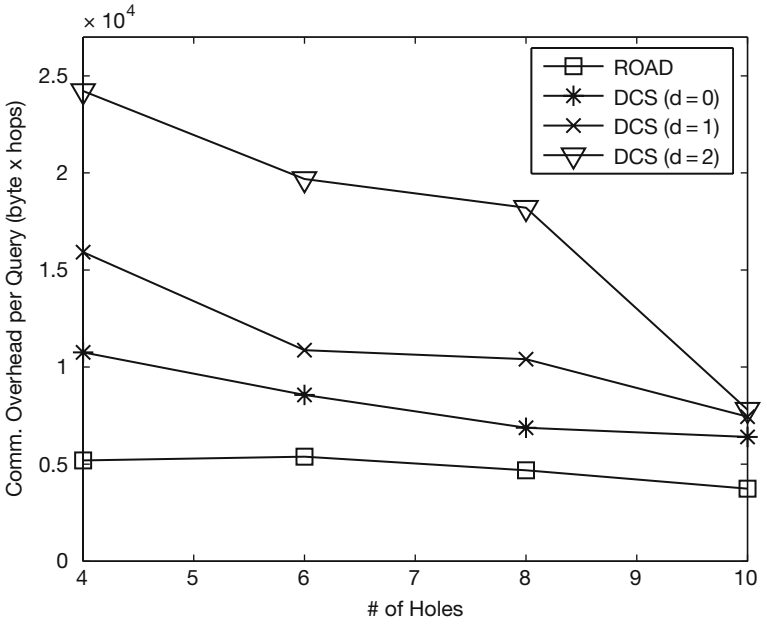


Fig. 12 Robustness against large scale sensor failures: communication overhead per query over a range of hole # (mean hole radius = 32.5 m)

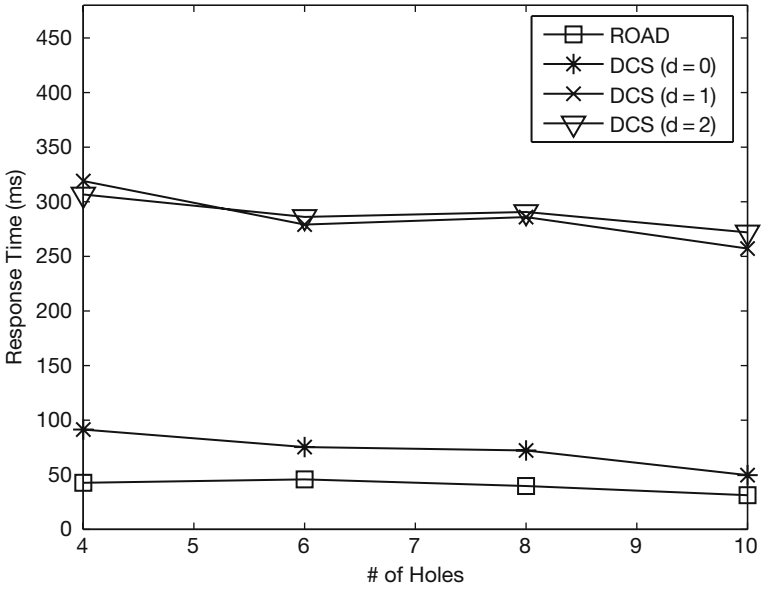


Fig. 13 Robustness against large scale sensor failures: response time over a range of hole # (mean hole radius = 32.5 m)

perimeter refresh protocol has been proven effective against individual sensor failures. However, since all the sensors on a perimeter are located close to each other, they may be all destroyed by a large scale sensor failure event, causing data loss. On the other hand, routing a query to access a trajectory is always easier than reaching a single location because accessing a trajectory only requires finding a path to any sensor on the trajectory.

5 Conclusion

This chapter proposes ROAD, a novel data dissemination scheme that publishes and retrieves data through sensors forming a path that best approximates a geographical trajectory. ROAD is robust against large scale sensor failures, which makes it suitable for sensor networks deployed in hostile environments. ROAD also presents high reliability and low communication overhead of data retrieval, and thus, it is efficient for applications with a large number of sinks that frequently issue queries. Simulation results provided a comparison between ROAD and DCS protocols in terms of communication overhead, response time, reliability and robustness against large scale sensor failures, and established ROAD as an effective data dissemination solution for WSN applications in hostile environments.

References

1. Albowicz, J., Chen, A., Zhang, L.: Recursive position estimation in sensor networks. In: Proceeding of the International Conference on Network Protocols (ICNP). Riverside, California (2001)
2. Bharambe, A., Agrawal, M., Seshan, S.: Mercury: Supporting scalable multi-attribute range queries. In: Proceeding of ACM SIGCOMM. Portland, Oregon (2004)
3. Braginsky, D., Estrin, D.: Rumor routing algorithm for sensor networks. In: Proceeding of WSNA. Atlanta, Georgia (2002)
4. Bulusu, N., Heidemann, J., Estrin, D.: GPS-less low cost outdoor localization for very small devices. IEEE Personal Communications **7**(5), 28–34 (2000)
5. Clarke, I., Sandberg, O., Wiley, B., Hong, T.: Freenet: A distributed anonymous information storage and retrieval system. In: Proceeding of the ICSI Workshop on Design Issues in Anonymity and Unobservability. Berkeley, California (2000)
6. Desnoyers, P., Ganesan, D., Shenoy, P.: Tsar: A two tier sensor storage architecture using interval skip graphs. In: Proceeding of ACM SenSys. San Diego, California (2005)
7. Fang, Q., Gao, J., Guibas, L.: Landmark-based information storage and retrieval in sensor networks. In: Proceeding of IEEE INFOCOM. Barcelona, Spain (2006)
8. Fang, Q., Gao, J., Guibas, L., Silva, V., Zhang, L.: Glider: Gradient landmark-based distributed routing for sensor networks. In: Proceeding of IEEE INFOCOM. Miami, Florida (2005)
9. Ghose, A., Grossklags, J., Chuang, J.: Resilient data-centric storage in wireless ad-hoc sensor networks. In: Proceeding of the International Conference on Mobile Data Management (MDM). Melbourne, Australia (2003)
10. Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: uAMPS ns code extensions. http://www-mtl.mit.edu/researchgroups/icsystems/uamps/research/leach/. Accessed 21 July, 2008
11. Intanagonwiwat, C., Govindan, R., Estrin, D.: Directed diffusion: A scalable and robust communication paradigm for sensor networks. In: Proceeding of ACM MobiCom. Boston, Massachusetts (2000)
12. Karp, B., Kung, H.: GPSR: Greedy perimeter stateless routing for wireless networks. In: Proceeding of ACM MobiCom. Boston, Massachusetts (2000)
13. Li, X., Kim, Y., Govindan, R., Hong, W.: Multi-dimensional range queries in sensor networks. In: Proceeding of ACM SenSys. Los Angeles, California (2003)
14. Niculescu, D., Nath, B.: Trajectory based forwarding and its applications. In: Proceeding of ACM MobiCom. San Diego, California (2003)
15. Priyantha, N., Chakraborty, A., Balakrishnan, H.: The cricket location-support system. In: Proceeding of ACM MOBICOM. Boston, Massachusetts (2000)
16. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content-addressable network. In: Proceeding of ACM SIGCOMM. San Diego, California (2001)
17. Ratnasamy, S., Karp, B., Shenker, S., Estrin, D., Govindan, R., Yin, L., Yu, F.: Data-centric storage in sensor nets with GHT, a geographic hash table. Mobile Networks and Applications **8**(4), 427–442 (2003)
18. Ratnasamy, S., Karp, B., Yin, L., Yu, F., Estrin, D., Govindan, R., Shenker, S.: GHT: A geographic hash table for data-centric storage in sensornets. In: Proceeding of ACM WSNA. Atlanta, Georgia (2002)
19. Stoica, I., Morris, R., Karger, D., Kaashoek, M., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: Proceeding of ACM SIGCOMM. San Diego, California (2001)
20. Ye, F., Luo, H., Cheng, J., Lu, S., Zhang, L.: A two-tier data dissemination model for large-scale wireless sensor networks. In: Proceeding of ACM MOBICOM. Atlanta, Georgia (2002)

Markov Decision Process-Based Resource and Information Management for Sensor Networks

David Akselrod, Thomas Lang, Michael McDonald, and Thiagalingam Kirubarajan

Abstract In this chapter, we consider the problem of managing a network of sensors with particular application to multisensor multitarget tracking. We study the problem of decision based control of a network of sensors carrying out surveillance over a region that includes a number of moving targets. The objective is to maximize the information obtained and to track as many targets as possible with the maximum possible accuracy. Uncertainty in the information obtained by each sensor regarding the location of the targets is addressed in the problem formulation. The chapter presents a number of solutions for centralized and decentralized tracking involving sensor management and distributed information flow control. We consider a distributed data fusion system consisting of sensors that are decentralized, heterogenous, and potentially unreliable. The objective function for sensor management is based on the Posterior Cramer-Rao lower bound and constitutes the basis of a reward structure for Markov decision processes that are used, together with decentralized lookup substrate, to control the data fusion process. In distributed sensor network fusion, we analyze three distributed data fusion algorithms: associated measurement fusion, tracklet fusion and track-to-track fusion. The chapter also provides a detailed analysis of communication and computational load in distributed tracking algorithms. In centralized sensor network fusion, we introduce a multi-level hierarchy of MDPs to control each of the sensors in the network. Simulation results are presented on a representative multitarget tracking problem using a network of sensors showing a significant improvement in performance compared to the existing algorithm.

1 Introduction

The problem considered in this chapter is the optimization of the information obtained by a number of sensors tracking a number of confirmed and suspected targets. We also consider a distributed data fusion system consisting of sensors that are decentralized, heterogenous, and potentially unreliable.

D. Akselrod (✉)
ECE Department, McMaster University, Hamilton, ON, Canada
e-mail: akselrd@mcmaster.ca

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_8,

Several sensor management algorithms have been presented in the literature. In [12] a system architecture for the target assignment and coordinated intercept problem was developed. The path-planning problem was solved via a Voronoi diagram and Eppstein's k-best paths algorithm. As an approach to finding an optimal solution for the team as a whole, a decomposition strategy that lets team-optimal solutions to be calculated in a decentralized manner was presented in [62]. A decision and control scheme that creates a model in which the gain is based on maximizing the expected number of targets found given some a priori information was presented in [43]. In that approach a feasible method of cooperation is achieved by considering other vehicles as stochastic elements. The problem of sensor management and coordinated sensor control in decentralized sensing networks was addressed in [42]. It builds on techniques established for the related problem of decentralized data fusion. The methods are based on the use of mutual information gain measures to formulate local and global objective functions for the sensor network. The approach to cooperative control presented in [66] aims to decouple, in part, the central problems of formation maintenance and maneuver management. For this purpose it introduces to the group a virtual body that is a collection of linked, moving reference points. The approach in [44] presents a decentralized on-line control strategy that lets cooperative sensors to engage multiple targets time-optimally. A UAV placement algorithm described in [79] and [80] proposes a Fisher information based approach for optimal Ground Moving Target Indicator (GMTI) sensor placement. An information based criterion, is used to select the path of an sensor such that the total information, obtainable by the sensors in the sensors as a group, corresponding to the detected targets, is maximized. A single-level MDP approach for sensor management was presented in [2]. The criterion used for the policy update provides a robust solution without compromising the precision of the approach. The approach of using information-based objective function yielded good results when incorporated in the MDP reward structure. In [6] a multi-level hierarchy of MDPs with each level in the hierarchy solving a problem at a different level of abstraction was presented. The problem of sensor selection in multitarget tracking with unknown associations of measurements to targets, and also with unknown and potentially time-varying number of targets was considered in [87].

In this chapter, we are interested in considering the sensor management problem for multitarget tracking using a tractable approach. In many of the solutions presented above, polynomial convergence of the proposed algorithms is not guaranteed or the computational complexity is not presented. The approach proposed in this chapter considers sensor placement task as a goal-oriented decentralized Markov Decision Process (Dec-MDP) with independent transitions and observations, solved by decomposing it into a number of Markov Decision Processes (MDPs). The individual Markov Decision Processes communicate with each other before they update their individual policies. Under certain conditions, the optimization problem, presented as a Dec-MDP may be decomposed into a number of MDPs that can be solved in polynomial time. Therefore, our algorithm will scale up better in larger environments with a high number of targets than those with higher complexity. The criterion used for the policy update provides a robust solution without

compromising the precision of the approach. The use of information based objective function yielded good results when incorporated in the MDP reward structure.

Although a single-level MDP sensor management approach is certainly attractive in case of multiple targets as the optimization problem need not be solved for each one of the targets individually, the decision regarding a sensor trajectory may be suboptimal. The proposed solution is multi-level hierarchy of MDPs controlling each of the sensors. Each level in the hierarchy solves a problem at a different level of abstraction. As a result, a sensor will navigate more precisely in a populated MDP sectors of the higher level as they will be represented by a lower-level MDP process. However, even using a multi-level MDP structure, the regular solution of MDP introduces a number of drawbacks to the practical problems solved using an MDP framework. Choosing just a single action for the given state out of several possible actions exposes the issue of choosing such action based merely on its yielding the highest value of state and eliminating another possible actions which yield the same or somewhat lower value. Also, because of inevitable noise components present in the system, the action yielding the highest value of state, would not necessarily be the one yielding the highest value of state if the noise were eliminated from the system. In addition, choosing a single action for the given state causes a number of problems in practical applications. One of them is loop formation, especially taking into account loops mistakenly formed because of processing noisy observations data on the basis of which the policy was calculated. Another issue is decreasing the potential coverage of the area under surveillance thus reducing the likelihood of discovering new targets. In this chapter, we present a Dynamic Element Matching (DEM) based modification of the classic Value Iteration (VI) algorithm to calculate the optimal policy of an MDP. A number of newly introduced performance metrics, such as Mean Opportunity Index, Maximal Opportunity Index and Area Coverage Index, verify the effectiveness of an MDP policy, especially for quantifying the impact of the modified DEM-based VI algorithm on the overall performance. The modified algorithm is applied to control a group of sensors carrying out surveillance over a region that includes a number of moving targets and demonstrates accuracy improvement in position and velocity RMSE of the targets under surveillance. The proposed method provides robust performance while guaranteeing polynomial computational complexity.

In this chapter we also consider the problem of transferring significant amounts of data through communication channels that are subject to certain capacity limits, associated costs of data transfers, reliability and security issues. A number of multisensor tracking systems and data fusion techniques have been introduced in the literature [9, 7, 34, 56, 78]. Several practical distributed tracking algorithms, including distributed track fusion, track fusion using tracklets and distributed composite tracking have been identified [7]. What is common to many of the above is that the data from each platform, which may correspond to a different type of data fusion or data format, are passed to all the other platforms. The assumption that the available data channel capacity will suffice to carry all the required data will not hold in applications that feature a large number of platforms with high volumes of data to be exchanged among them (e.g., a network of airports). A number of works

present solutions for distributed sensor networks applications and network-centric data fusion where limited communication capacity is considered. An information graph approach is introduced in [32]. Using the information graph model, common information can be identified and removed to produce the optimal estimate. The approach of using graphical models was also used in [30] to develop distributed fusion algorithms reducing communication. An agent-based fusion model is presented in [68], where agents model fusion entities' capabilities, expertise and intentions and perform fusion based on their intentions. They cooperate with each other to extract the relevant information in order to achieve their intentions. Issues associated with distributed multiple target tracking for ad hoc sensor networks are discussed in [35], which examines the applicability of tracking algorithms developed for traditional networks of sensors. A decision fusion rule based on the total number of detections made by local sensors, for wireless sensor networks with a large number of sensors was proposed in [64]. In [31], distributed fusion and communication management algorithms for target identification, where information graphs are used to select fusion architectures that minimize the effect of information double counting due to communication, were presented. Strategies to determine when a fusion agent should send its estimate to another fusion agent were also developed. Techniques to solve data association problems arising in distributed sensing scenarios were presented in [28], which introduced a communication-sensitive message-passing algorithm for achieving near-optimal performance with substantial savings in communication. A comparison of two different approaches for sensor selection for distributed tracking was presented in [22]. An approach for fusing information from diverse sources based on the quality of the source was presented in [96]. An algorithm for determining the quality of sensor data in the fusion process was presented in [19]. An information valuation metric for sensor networks was described in [74].

The chapter describes a distributed data fusion system consisting of platforms that are completely decentralized, independent and heterogenous. The platforms are also assumed unreliable, i.e., they may join or leave the network at any time, are not committed to share the information, and may have unreliable communication channels with limited capacity. When requesting specific information from such a platform we do not know if the requested information will be provided as we have only the statistical characteristics describing the ability of the platform to provide such information. Data transfer from a platform in the system may be interrupted at any moment. Similarly, a refusal to supply the specific data from a platform does not mean that the next request will be refused as well. We are considering a situation where the decisions regarding the data flow of the information in the distributed data fusion system should be made sequentially based on the currently observable state that reflects fully or partially the state of the environment. The result of each decision cannot be fully predicted, but can be accounted for using available statistical information before the next decision is made.

This chapter also presents a decision mechanism that provides each platform with the required data for the distributed data fusion process subject to the available channel capacities and reducing redundancy in the information flow in the overall system. The proposed approach, which is based on Markov decision processes

(MDP) and decentralized lookup substrate (i.e., the way to identify efficiently all the platforms in the distributed network that possess the required information), will control the data fusion and information exchange process based, among the other parameters, on information gain metrics of individual platforms, enhancing the total distributed system's reliability as well as that of each participating platform. The chapter includes a complete solution for the distributed data fusion architecture timely providing participating platforms with specific decisions regarding obtaining information that is needed for the data fusion.

We also present three distributed data fusion algorithms – associated measurement fusion, tracklet fusion and track-to-track fusion [37, 81] – to be applied with the MDP-based data fusion system. We compare the performance based on the type of the data fusion used. In track-to-track fusion [26, 59] sensor measurements are used to update the tracks of each of the local trackers. The updated tracks are then communicated to the other platforms based on the pre-defined policy. Since compressed information (tracks, not measurements) is shared, using track-to-track fusion results in suboptimal performance. In addition to that, in the presence of process noise, this technique requires computation of the filter gains at the update instances of the received tracks. This step is required to compute the cross-correlation between the local and received tracks. Hence each local tracker would be required to compute the filter gains for all tracks at all local trackers. This makes the algorithm computationally burdensome. In [8, 59] an approximation technique is presented for the computation of cross-covariance matrices between tracks from different sources, which is used in this work. In the second fusion approach, the local tracks are first updated using the measurements available to the corresponding platform. After a few updates these tracks are decorrelated with the prior information and communicated for fusion to the other platforms. The decorrelated tracks are called tracklets in the literature [40].

The architecture involving tracklets is considered to be a special case of track fusion. Since they are decorrelated from the local tracks, the tracklets can be treated as measurements of the track states. In this case no further computation of the correlation between local tracks and tracklets is required. However, in the presence of process noise, the tracklets communicated by various platforms cannot be totally decorrelated from the local tracks. The degree of the residual correlation depends on maneuvering index [27] of the tracks. In the third fusion architecture, the sensor measurements are associated with the local tracks maintained in individual platforms and only then the associated measurements reports (AMRs) are communicated to the other platforms [7]. This architecture is easier to implement than the track fusion algorithms due to the independence of information from different sources. That stems from the fact that, unlike local tracks, measurements from different local platforms can be considered to be independent of each other. We present convergence and complexity analysis and demonstrate the proposed algorithm using simulation results. The proposed approach does not require any modification to the individual platforms or communication network connecting them beside the ability to fuse external data containing required information such as tracks, tracklets or AMRs with the platform's own data.

2 Decision-Based Resource and Information Management

2.1 Problem Formulation

Figure 1 shows the area under surveillance divided into sectors that may contain a number of targets as well as sensors that need to track these targets. A sensor actions are contained within a set of possible actions $A = \{a_i | i = 1, \ldots, N\}$. In this chapter, possible action set includes moving one sector to the North, West, East, or South. However, it may include much more sophisticated actions. We assume that with each such action is associated a probability P of reaching the intent of that action, that besides the action itself depends on the current state of the sensor (e.g., the physical location of the sensor) and on the goal to be reached. It means that the sensor moving to the North could actually have moved to the East.

In each of its possible states the sensor receives a certain reward, and the performance of the sensor will be measured by a sum of rewards for the states visited. The abbreviated objective function is thus formulated as:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}\left[\Sigma_{t=0}^{\infty} U_h([s_0, s_1, \ldots])|\pi\right] \tag{1}$$

where π^* is the policy of actions maximizing the expectation of the utility function U_h of all the sequence of states $[s_0, s_1, \ldots]$ which resulted from following that policy.

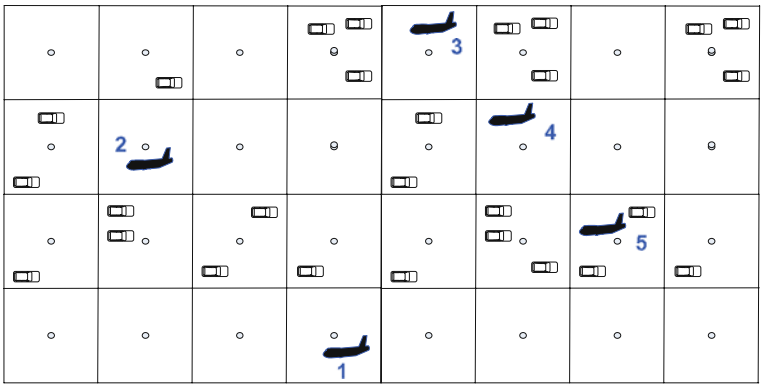


Fig. 1 Area under surveillance containing a number of targets and sensors

2.2 Sensor Management as a Decision Mechanism

Choosing to take a specific move introduces a new decision to be made regarding the one to follow. The decision to take a move in a certain direction may not be carried out as planned because of external conditions or errors in navigation. After this situation is discovered and the current state is updated, the sensor will have to make a new decision taking its current state into account. What is important to note

is that in such a case, the optimal configuration is not achievable by a single decision or solution. It is rather an outcome of a multi-stage process where to each state of the system corresponds the optimal action (taking into account the information available to the sensor) that can be taken in the given state.

In the problem we are considering, we do not seek a solution at a particular time only. That is, the approach, once having solved an optimization problem we have formulated, will provide the optimal decision for the next step taking into account the outcome of the previous one. It means that we are considering a situation where the decisions should be made sequentially or in a recursive manner. The result of each decision cannot be fully predicted, but can be accounted for using available statistical information before the next decision is made. The objective is to minimize the cost of our actions (or, alternatively, maximize a utility function) [18, 51]. One of the approaches that could be utilized is using a Markov Decision Processes – stochastic processes controlled by a decision maker. An infinite-horizon fully observable MDP is defined by the model $M = < S, A, P, R >$ where S is the finite set of world states, A is the finite set of actions, P is the transition probability function, and R is the real-valued reward function. In an MDP, a policy $\pi \in \Pi$ is a mapping from states to actions. $\pi : S \rightarrow A$. Because of the Markov property, the action taken depends on the current state only and not on any of the previous states. The MDP model is solved using dynamic programming approaches, which have been described in [13, 16] as well as in many other works. Given a stationary discrete-time process,

$$s_{t+1} = f(s_t, a_t, w_t), \quad t = 0, 1, \ldots, \tag{2}$$

where s_t is the state at time t, a_t is the action at time t that depends only on the current state s_t, and w_t is the random disturbance; the problem is to find a stationary policy π. This policy maps each $s_t \in S$ to $a_t \in A$, which, given an initial state s_0, maximizes the following value function given by

$$\begin{aligned} V_\pi(s_0) &= \lim_{N \longrightarrow \infty} \mathbb{E} \left\{ \sum_{k=0}^{N-1} \gamma^k R(s_{t+k}, a_{t+k}) | s_t = s_0 \right\} \\ &= \sum_{s_{t+1} \in S} P(s_{t+1} | s_t = s_0, a_t) \left[R(s_t, a_t) + \gamma V_\pi(s_{t+1}) \right] \end{aligned} \tag{3}$$

where γ is a discount factor, $0 < \gamma < 1$. The random disturbance w_t is represented by the transition probability P in an MDP. Then, the optimal value function is defined as

$$V^*(s) = \max_{\pi \in \Pi} V_\pi(s), \quad s \in S \tag{4}$$

The policy π is optimal if $V_\pi(s) = V^*(s)$ for all the states $s \in S$. The optimal V^* is unique and is the solution to the Bellman equation [13]

$$V^*(s_t) = \max_{a \in A} \mathbb{E}\{R(s_t, a_t) + \gamma V^*(f(s_t, a_t, w_t))\}, \ \forall s_t \in S \tag{5}$$

Discount factor γ expresses the dependence of the value function on current rewards over future rewards. Using the elements of the MDP, (5) can be expressed as

$$V^*(s_t) = \max_{a \in A}(R(s_t, a_t) + \gamma \sum_{s_{t+1} \in \mathcal{S}} P(s_t, a_t, s_{t+1}) V^*(s_{t+1}), \ \forall s_t \in \mathcal{S} \tag{6}$$

Computational complexity of MDPs was analyzed in [67] and it was shown that, under any of the three cost criteria [60, 67], namely, the expected cost to target, expected discounted cumulative cost, and average expected cost per stage, the problem is P-complete.

2.3 Policy Update and Termination Criteria

When a certain policy is calculated, it is assumed that during its execution the conditions and the state of the environment that contributed to the specific policy do not change significantly, except for the changes in the environment as a direct consequence of the actions that are taken. For example, transition probability P and reward R matrices are supposed to reflect the environment at all times during the policy execution. It is assumed that the locations of all information sources have not been changed. Obviously, these assumptions may not hold, at least not for a long period of time, which means that the policy should be re-evaluated from time to time as shown in Fig. 2, taking into account any new information that might has arrived.

This approach is similar to sampling an analog signal, where the sampling frequency reflects the dynamics of the signal, specifically its highest harmonic. In our case, the environment change rate will also influence the policy update rate. In some cases, *premature policy termination* will be performed. We define a criterion for a premature policy change as exceeding the maximum allowed difference between the state of the world as we observed it at the beginning of the current policy and the updated one during the policy execution.

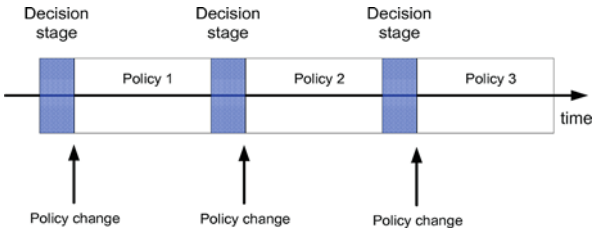


Fig. 2 Policy update stages

3 Decision-Based Multitarget Tracking

3.1 MDP-Based Structure for Multisensor Multitarget Tracking

The overall problem of sensor management can be formulated as a Dec-MDP process possessing the qualities mentioned above, which allows us to decompose it into a number of MDPs. We specify below all the elements of these MDPs that are solved optimally to attain the most desirable solution to the decentralized optimization problem. The problem solved by each of the composing MDPs is an optimization of the information obtained by a moving sensor carrying out surveillance over a region which includes a number of confirmed and suspected targets. The overall surveillance region is divided into a number of MDP sectors of which the corresponding MDP is responsible for a certain part.

In this chapter, we divide the area of surveillance into 25 equal sectors, which we refer to as MDP sectors. Any target that is situated within the boundaries of the area will belong to one of these 25 sectors. All the sensors act within the boundaries of these sectors when each one is responsible for the designated area. The division to MDP sectors is independent of the division to sectors to be scanned by a sensor. It is assumed that the sensor is able to detect the targets situated within the MDP sector only when the sensor is located close to the sector boundaries. The sensor has to decide on the optimal path to follow in order to cover as many targets as possible during its operation at the lowest cost. A separate MDP is allocated to each sensor at each of the levels.

- Set of states S:
  The state of the MDP, corresponding to each of the sensors, consists of its location combined with the states of all the populated sectors that the sensor is responsible for. A sector is considered populated if it contains at least one target. Once a certain sector has been visited by a sensor, its status is updated (cleared). Once the status bits of all the populated sectors are cleared, they will be set again as the sensor continues the surveillance.
- Set of actions A:
  The set A contains all the possible actions that a sensor can take in order to make its next move. In our case, we have four actions in the set, each one instructing it to move in either North, South, West, or East direction.
- Transition probabilities $Pr(s_{t+1}|s_t, a_t)$:
  The sensor has a priori information regarding the targets it needs to visit as well as other factors which may reduce its chance of survival, chance of getting to the correct location, etc. For example, if we know that high winds blow in the vicinity of a certain sector, the chance of the sensor to arrive to the desired location will be lower. All such information is eventually mapped to the transition probability matrix Pr that specifies the probability of transition from state s_t to state s_{t+1} while performing an action a_t.

- Real-valued reward function on states $R(s)$:
  $R(s)$ contains the value of the immediate reward associated with state s. It is a tool to specify priorities in getting to specific information source. We can express R as

$$R(s) = \Theta r(s) - (1 - \Theta)c(s), \quad 0 \leq \Theta \leq 1 \tag{7}$$

  where $r(s)$ is the revenue associated with being in state s and $c(s)$ is the cost associated with it. Coefficient Θ balances between considerations of priority to reach the state s and the cost of reaching it. For example, in mission-critical or military applications, Θ will be higher than that in commercial applications.
  In the above, $r(s)$ and $c(s)$ are expressed as

$$r(s) = \sum_{i=1}^{N} r_i^w(s) r_i^{re}(s) = R_s^{wT} R_s^{re}, \quad r_1^w(s) + \ldots + r_N^w(s) = 1 \tag{8}$$

$$c(s) = \sum_{j=1}^{M} c_j^w(s) c_j^{ce}(s) = C_s^{wT} C_s^{ce}, \quad c_1^w(s) + \ldots + c_M^w(s) = 1 \tag{9}$$

  where $R_s^{re} = [r_1^{re}(s), \ldots, r_N^{re}(s)]^T$ and $C_s^{ce} = [c_1^{ce}(s), \ldots, c_N^{ce}(s)]^T$ are vectors containing contributing elements of the state revenue and state cost, respectively. Vectors $R_s^w = [r_1^w(s), \ldots, r_N^w(s)]^T$ and $C_s^w = [c_1^w(s), \ldots, c_N^w(s)]^T$ contain weights, which control the influence of R^{re} and C^{ce} elements on the revenue $r(s)$ and the cost $c(s)$, respectively. In general, both cost and revenue weights may depend on the state, but in many applications they can be independent of the system state s.

Figure 3 shows the pseudo code of the sensor control algorithm. It has the following key elements:

- Initialization stage, during which the area under surveillance is determined and the initial parameters are set.
- Sensor movement control stage. During this stage an sensor is controlled according to the evaluated MDP policy. After the current policy is calculated, a new action is determined. This action may not necessarily result in a flight direction change. If it does, a new maneuver is initiated. Sensor movement direction changes take place in pre-defined regions, designated as Maneuver zones. When in a Maneuver zone and in need to change its flight direction, a sensor will engage maneuver control (i.e., resulting in coordinated turn start).
- Scan decision control. Each sensor decides on the sectors to scan in a process separate from path selection. In this stage, a fixed number of surveillance sectors to be scanned are selected.
- In the last stage, information obtained by sensors is processed.

function SensorControlAlgorithm

1. **Initialization:** Designate surveillance area. Set initial parameters
2. Sensor movement control.

   a. **If** (NoPolicySet *Or* ReachedPolicyUpdate)
   ChangePolicy:

      i. Build updated list of sectors: (a) populated *AND* (b) unscanned over time limit
      ii. Calculate updated TransitionProbabilityMatrix P
      iii. Calculate updated RewardFunction R
      iv. Calculate MDP policy π

   b. **ExecuteCurrentAction**

   - **NewManeuverCheck**
     If($a(s_i) \neq a(s_{i-1})$)
     ExecuteManeuver: perform movement direction change according to $a(s_i)$
     Else
     ContinueCurrentAction: according to $a(s_i)$. No direction change.

   - **BoundaryCheck**
     Until(*ReachedNewSector*)
     ContinueCurrentAction: Perform action according to $a(s_i)$
   - **NewSectorDecisionPoint**
     If(*ReachedPolicyUpdate*)
     ChangePolicy: Perform steps **i** to **iv** from **2-a** above.
     Else
     DetermineNewAction: $a(s_i)$
     ExecuteCurrentAction: Perform steps in **2-b** above.

3. Scan decision control.
4. Process obtained measurements.

Fig. 3 Sensor control algorithm. After the current policy is calculated, a new action is determined. This action may not necessarily result in a movement direction change. If it does, a new maneuver is initiated

3.2 Expected Information Gain-Based Reward Structure of MDP for Sensor Management

In this work, the authors utilize the approach of using information-based objective function. The objective function is based on the Posterior Cramér-Rao lower bound (PCRLB) [88]. Let $\hat{X}(Z)$ be an unbiased estimate of a r-dimensional random state vector X, based on measurement vector Z. The PCRLB for the estimation error is defined to be the inverse of the Fisher Information Matrix (FIM) [91], J, providing a lower bound of the estimation error:

$$C(k) \triangleq \mathbb{E}\{[\hat{X}_k(Z_k) - X_k][\hat{X}_k(Z_k) - X_k]'\} \geq J(k)^{-1} \tag{10}$$

The $r \times r$ dimensional Fisher information matrix J is defined as:

$$J \equiv \mathbb{E}\{[\nabla_x \log(p(Z, X))][\nabla_x \log(p(Z, X))]'\} \tag{11}$$

where $p(Z, X)$ is the joint probability density function of (Z, X) and $\mathbb{E}$ denotes expectation over (Z, X).

Assuming independently moving targets, the linear state equation can be decomposed into M equations [86]:

$$X_{k+1}^t = F_k^t X_k^t + v_k^t \quad t = 1, 2, \ldots, M \tag{12}$$

In the equations above, M is a number of targets, F_k^t is the state transition matrix and v_k^t is the process noise of target t. For the linear system (12), the sequence $\{J_k\}$ of posterior information for estimating X_k is given by the following recursion [88]:

$$J(k+1) = D_k^{22} - D_k^{21}(J(k) + D_k^{11})^{-1} D_k^{12} \tag{13}$$

where [73]:

$$D_k^{11} = F_k' \Gamma_k^{-1} F_k \tag{14a}$$

$$D_k^{12} = -F_k' \Gamma^{-1} = [D_k^{21}]' \tag{14b}$$

$$D_k^{22} = \Gamma_k^{-1} + J_Z(k+1) \tag{14c}$$

Using the Matrix Inversion Lemma, it can be shown [73] that (13) and (14) are equivalent to the following recursion:

$$J^t(k+1) = [\Gamma_k^t + F_k^t J^t(k)^{-1}(F_k^t)']^{-1} + J_Z^t(k+1) = J_X^t(k) + J_Z^t(k+1) \tag{15}$$

where $J_X^t(k)$ expresses the prior information and $J_Z^t(k+1)$ the measurement contribution, respectively. Assuming measurement origin uncertainty, the j-th measurement at the i-th sensor at time step k is given by either

$$Z_k^i = [h_k^i]^t(x_k^t) + [\omega_k^i(j)]^t \tag{16}$$

if the measurement originated from target t, or by

$$Z_k^i = \upsilon_k^i(j) \tag{17}$$

if it is a false alarm. In the equations above, $[h_k^i]^t$ is a non-linear function, $[\omega_k^i(j)]^t$ is a zero mean Gaussian random variable with covariance R_k and $\upsilon_k^i(j)$ is distributed uniformly across the surveillance region. It can be shown [48, 47] that

$$J_z^t(k) = \sum_{i=1}^{n} J_{z_i}^t(k) = \sum_{i=1}^{n} \mathbb{E}\{([H_k^i]^t)' R_k^{-1} [H_k^i]^t\} \tag{18}$$

where R_k is the measurement noise covariance and (a, b)-th element of matrix $[H_k^i]^t$ is given by:

$$[H_k^i]^t(a, b) = \frac{\partial [h_k^i]^t(a)}{\partial x_k^t(b)} \tag{19}$$

We can see that (13) is equivalent to:

$$J^t(k+1) = J_X^t(k) + J_Z^t(k+1) = J_X^t(k) + \sum_{i=1}^{n} \mathbb{E}\{([H_{k+1}^i]^t)' R_{k+1}^{-1} [H_{k+1}^i]^t\} \tag{20}$$

which has the same form as the Extended Kalman Filter (EKF) covariance matrix propagation equation except for the expectation operator. In this chapter, we defined trace as a scalar performance measure of PCRLB which is proportional to the circumference of the rectangular region enclosing the minimum achievable covariance ellipsoid.

The objective function corresponding to M targets for a given sensor is thus given by:

$$I(k) = \sum_{j=1}^{M} \log\left(\text{trace}\{J_j(k|k)\}\right) = \sum_{j} \log\left(\text{trace}\{P_j(k|k)^{-1}\}\right) \tag{21}$$

where $P_j(k|k)$ is the posterior covariance matrix of the state vector corresponding to target j at time k.

The reward associated with the state of a sensor corresponding to a specific MDP sector is expressed as the expected information gain corresponding to this MDP sector. One MDP sector may contain a number of surveillance sectors – a set of areas that can be covered during a single sensor radar scan. Then, the reward of the MDP sector comprising W surveillance sectors is

$$\sum_{f=1}^{W} I_{f\ m,n}^{\text{scan}}(k, s) \tag{22}$$

where $I_{f\ m,n}^{\text{scan}}(k, s)$ is the expected information gain from the surveillance sectors to be covered by a sensor. If there are $N_{m,n}(k)$ targets in surveillance sector (m, n) (out of mn surveillance sectors) at time step k, then the expected information gain by sensor s from this sector is given by

$$\begin{aligned} I_{m,n}^{\text{scan}}(k, s) &= \sum_{j=1}^{N_{m,n}(k)} \log \text{trace}\{J_{s,j}(k|k)\} - \log \text{trace}\{J_{s,j}(k|k-1)\} \\ &= \sum_{j=1}^{N_{m,n}(k)} \log \text{trace}\{J_{s,j}(k|k-1) + \pi_D(k, s, j) J_{z_s}(k)\} - \log \text{trace}\{J_{s,j}(k|k-1)\} \end{aligned} \tag{23}$$

where $J_j(k|k-1)$ is the predicted information matrix and $J_{s,j}(k|k)$ is the updated posterior information matrix corresponding to target j, and $\pi_D(k, s, j)$ is the target detection probability expressing the reduction in the target originated measurement contribution to potential information gain [79]. For a GMTI radar, a target will not be detected if the magnitude of the target's measured range rate falls below a threshold $\dot{r}_{\min}$. Therefore, $\pi_D(k, s, j)$ is given by

$$\pi_D(k, s, j) = 1 - P\{|\dot{r}(k, s, j)| < \dot{r}_{\min}|\dot{r}(k, s, j|k-1), \sigma_{\dot{r}}(k, s, j|k-1)^2\} \quad (24)$$

where $\dot{r}(k, s, j)$, $\dot{r}(k, s, j|k-1)$ and $\sigma_{\dot{r}}(k, s, j|k-1)$ are the measured range rate, predicted range rate and the variance of the predicted range rate respectively. If there are no detected targets in the sector (m, n), $I^{\text{scan}}_{m,n}(k, s)$ is given by

$$I^{\text{scan}}_{m,n}(k, s) = \log \text{trace}\{J_{m,n}(k, s)\} - \log \text{trace}\{J^0_{m,n}\} \quad (25)$$

where $J^0_{m,n}$ is the prior information matrix for sector (m, n) and $J_{m,n}(k, s)$ is the expected updated information matrix corresponding to sector (m, n), when scanned by sensor s. $J_{m,n}(k, s)$ is expressed as follows:

$$J_{m,n}(k, s) = J^0_{m,n} + \tilde{\pi}_D(k, s, m, n)\tilde{\pi}_{\text{new}}(k, m, n)\tilde{H}(k, s, m, n)'\tilde{R}(k, s, m, n)^{-1} \tilde{H}(k, s, m, n) \quad (26)$$

where $\tilde{\pi}_{\text{new}}(k, m, n)$ is the probability of a new target appearing in sector m, n at time k and $\tilde{\pi}_D(k, s, m, n)$ is the probability of detection of that target. Also, $\tilde{H}(k, s, m, n)$ and $\tilde{R}(k, s, m, n)$ are measurement and measurement covariance matrices, respectively.

4 Multi-Level Hierarchy of MDPs for Sensor Management

In a single-level MDP approach, the surveillance area is divided into a number of MDP sectors. Each MDP sector may contain a number of targets. The informative value about the targets during one radar scan is represented collectively for that sector as follows:

$$J^{\text{scan}}_z(k, s) = \sum_{j=1}^{T(z)} \log \text{trace}\{I_{s,j}(k|k)\} - \log \text{trace}\{I_j(k|k-1)\} \quad (27)$$

where z is an MDP sector containing $T(z)$ targets at time k. Some MDP sectors may not contain any detected targets. For those sectors, undetected targets are the possible source of information. Both detected and expected targets in an MDP sector will define its reward function $R(s)$ where s a state of a sensor. Combination of the state of the populated MDP sectors as well as the current location of a sensor determines sensor's state. The solution of the Bellman equation describing the corresponding MDP process characterized by its set of states S, actions A, transition probabilities

Pr as well as the reward function on states $R(s)$ will provide the sensor with the optimal action $a(s)$ for each of its possible states s. When reaching a neighboring MDP sector after taking a previous action, according to the state s, a sensor chooses an action a which corresponds to one of the possible flight directions. The computational complexity of solving an MDP is P-complete, thus the proposed method is computationally attractive. In this section, we would like to present existing drawbacks to a single-level MDP approach. After dividing the area under surveillance into MDP sectors, all the targets located in the same MDP sector will be represented collectively. The approach above is certainly attractive with multiple targets as the optimization problem need not to be solved for each one of the targets individually. On the other hand, the decision regarding a sensor trajectory may be suboptimal because some, if not the majority of the targets, may be located away from the center of an MDP sector. An MDP sector is the lowest discrete level to which the optimization problem is divided. Therefore, the trajectory will connect centers of two adjacent MDP sectors ignoring the location of the targets within. One solution could be increasing the number of MDP sectors. That approach may increase the complexity of the problem as many of the MDP sectors may not be so densely populated to justify increasing the overall number of MDP sectors.

The proposed solution is multi-level hierarchy of MDPs controlling each of the sensors. Each level in the hierarchy solves a problem at a different level of abstraction. In this chapter, we present two levels of MDP-based sensor management. The first (highest) level is exactly the same as one in a single-level MDP approach. The second (lower) level comprises two adjacent MDP sectors from the higher level, providing an additional level of MDP control for sensors. Figure 4 shows the surveillance area for two MDP processes and Fig. 5 shows the pseudo code of the multi-level sensor control algorithm. As a result of a multi-level MDP hierarchy, a sensor will navigate more precisely in a populated MDP sectors of the higher level as they will be represented by a lower-level MDP process.

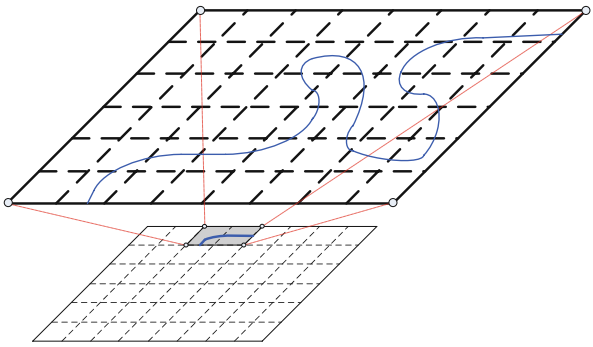


Fig. 4 Multi-level hierarchy of MDPs. Each MDP sector of the upper level is represented by a number of MDP sectors of the underlying MDP process. Each level in the hierarchy solves a problem at a different level of abstraction. A sensor will navigate more precisely in populated MDP sectors of the higher level as they will be represented by a lower-level MDP process

function MultiLevelSensorControlAlgorithm

1. **Initialization:** Designate surveillance area. Set initial parameters
2. Sensor movement control.

   a. *If* (NoPolicySetL1 *Or* ReachedPolicyUpdateL1)
   ChangePolicyL1:
   b. **ExecuteL1Maneuver**

   - **L1BoundaryCheck**
     Until(*ReachedNewL1Sector*)
     If(L2 enabled *And* L1SectorIsPopulated)
     sensorControlAlgorithm L2
     Else
     ContinueL1Maneuver according to $a(s)$
   - **NewL1SectorDecisionPoint**
     If(*ReachedPolicyUpdateL1*)
     ChangePolicyL1
     Else
     DetermineNewActionL1
     ExecuteManeuverL1

3. Scan decision control.
4. Process obtained measurements.

Fig. 5 Multi-level sensor control

5 Dynamic Element Matching-Based Modified Value Iteration Algorithm

5.1 Drawbacks of Finding the Optimal Policy of MDP

A policy π is a mapping from states to actions with a single action a_t corresponding to each state s_t. The policy π is optimal only if the value of each of the states $V_\pi(s)$ is maximized and is equal to the optimal value of state $V^*(s)$ for all the states $s \in S$. The optimal V^* is defined to be unique and is the solution of the Bellman equation. Therefore, it is implied that for any state s there could be just one optimal action $a(s)$.

This approach introduces a number of drawbacks to the practical problems solved using MDP framework. In practice, a number of actions may have the same or similar objective function. Also, noise may cause the wrong action to have the highest objective function. Choosing an action based merely on its yielding the highest value of state will thus eliminate another possible actions yielding the same or similar values of state. Figure 6 depicts a graphical representation of several actions corresponding to the highest values of state. In a classical approach only one action having a highest value will be chosen each time. In addition, choosing a single action for the given state may cause the following problems in practical applications:

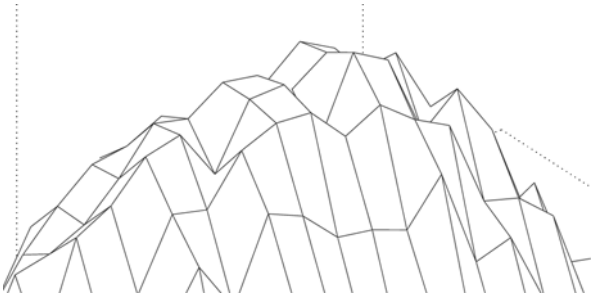


Fig. 6 Actions corresponding to highest values of different states. In a classical approach only one action having a highest value will be chosen each time

- Loop formation, especially taking into account loops mistakenly formed because of processing noisy observation data on the basis of which the policy is calculated.
- Decreasing the potential coverage thus reducing the likelihood of discovering new targets in the surveillance area. Obviously, by moving via different paths, a sensor can identify a larger number of potential targets.

5.2 Dynamic Element Matching

An early description of the Dynamic Element Matching principle was given in a patent by Van de Plassche [89] from 1976 as a mean of enhancing performance of current sources. His system is based on a circuit transferring to the output, according to a cyclic permutation, a number of currents, which differ in their values due to the mismatches in the manufacturing process. The resulting current has a value equal to the average value of the currents and a ripple that is formed by the differences between the currents and thus can be subsequently removed by a low pass filter. The proposed DEM algorithm reduced errors caused by mismatches in analog components of current sources by appropriate selection of different current sources every time it was required. Later, Van de Plassche described a high accuracy D/A converter, which employed the DEM principle [90]. Since then numerous patents and publications emerged [23, 58], describing use of the DEM algorithms in current generators design, as well as for achieving high integral linearity and low total harmonic distortion (THD) in D/A and A/D converters, without requiring precisely matched components. DEM techniques are most commonly used in multi-bit Delta-Sigma converters to achieve the required integral linearity without the use of precise component matching by dynamically rearranging the interconnections of mismatched components. In 2002 there was proposed an approach for incorporating a two-dimensional DEM technique for improving of a mixed signal WTA (Winner-Take-All) tracking [1]. There have been proposed many DEM techniques: Dynamic

Element randomization [24], Dynamic element Rotation [76], Individual level averaging [58], among many others. All of those techniques similar in dynamically rearranging the interconnections of mismatched components, converting harmonic distortions resulting from elements mismatch into a wide-bandwidth noise. In the case of Dynamic Element randomization, the purpose is to remove the correlation between the mismatch error at one time and the mismatch error at any other time thus converting it into white noise. The relationship between the in-band RMS noise divided by full scale and the percentage element mismatch is [65]

$$\sigma\left[\frac{n_{\text{inband}}}{V_0 M}\right] = \frac{1}{\sqrt{R}\sqrt{M}}\sigma\left[\frac{\Delta E_i}{E}\right] \tag{28}$$

where M is the number of elements (actions in our case), R is an oversampling ratio (number of occurrences of the same state in the policy) and $\frac{\Delta E_i}{E}$ is a fractional element mismatch (value of state mismatch in our case).

5.3 Modified Value Iteration Method

The proposed solution to the issues above is introducing a modified value (or policy) iteration algorithm utilizing DEM approach. The algorithm will identify several potential actions yielding highest value of state for a given state. After applying a certain threshold to the actions yielding highest values of state for each particular state, a group of several candidate actions are identified for each state. Figure 7 shows a set of actions having highest values of state that have been selected to form the set of potential candidate-actions. Consequently, one of DEM algorithms is applied in order to choose the winner-action for a given state. Each time this state is reached, another winner-action will be chosen based on the DEM method used. The pseudo-code of the modified VI algorithm is shown in Fig. 8.

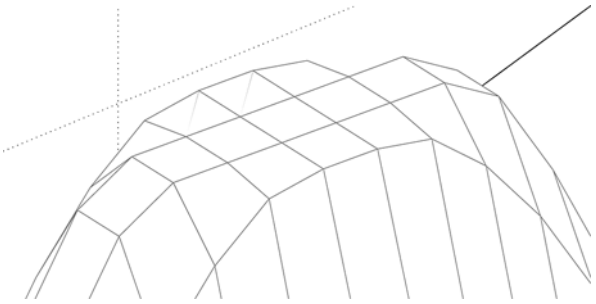


Fig. 7 A set of actions having highest values of state that have been selected to form the set of potential candidate-actions

function ModifiedValueIteration(MDP)
returns the optimal policy $\pi^*(s)$,
inputs: MDP=$< S, A, P, R >$
$V(s) = 0,\ s \in \mathscr{S}$
$\Delta_s \leftarrow 0,\ s \in \mathscr{S}$
loop
for $s \in \mathscr{S}$
$\upsilon \leftarrow V(s)$
$V_{s_t} \leftarrow P(s_{t+1}|s_t, a_t)\,[R(s_t, a_t) + \gamma V(s_{t+1})]$
$V_{1,2,\ldots,m}(s) \leftarrow \max_{a_{1,2,\ldots,m}} \sum_{s_t \in S} V_{s_t}$
$\Delta \leftarrow \max_a(\Delta, |\upsilon - V(s)|)$
return Δ_s
until $max(\Delta_s) < \Theta,\ s \in \mathscr{S}$
return $\pi^*(s) = DEM(arg\max_{a_{1,2,\ldots,m}}(\Delta, |\upsilon - V(s)|))$

Fig. 8 Modified value iteration algorithm. DEM algorithm is utilized to select dynamically the current action out of several equalized candidates

Figure 9 shows the queueing scheme for the modified Value Iteration VI algorithm. Dynamic Element Randomization or Dynamic Element Rotation can be incorporated to select action candidates winners in each of the DEM blocks.

To quantify the results and verify the advantage of the DEM-based MDP, we introduce the following metrics:

- *Mean Opportunity Index (MOI)*. MOI is defined as a mean candidate actions number per state in a policy. Higher MOI will increase a potential for a better performance while maintaining the same precision.
- *Maximal Opportunity Index (MAX OI)*. MAX OI is defined as a maximal candidate actions number per state in a policy.

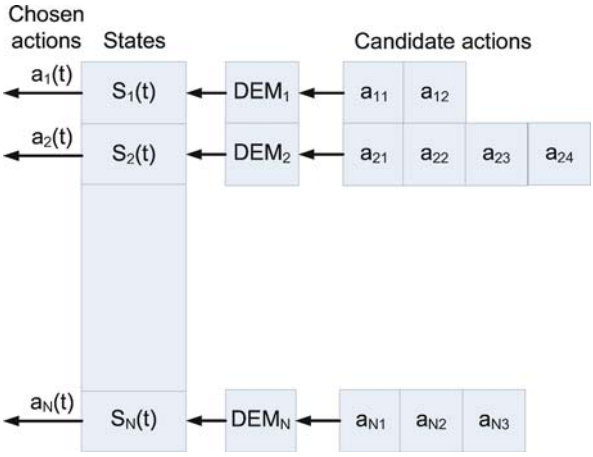


Fig. 9 Modified MDP solution scheme. Dynamic element randomization or dynamic element rotation can be incorporated to select action candidates winners in each of the DEM blocks

- *State Coverage Index (SCI).* SCI is defined as a mean ratio of a total number of different states reached during an execution of a policy to a total number of states.
- *Area Coverage Index (ACI).* ACI is defined as a mean ratio of a total number of different locations reached during an execution of a policy to a total number of location. Higher ACI means that the resulting area coverage during a policy execution is higher. Higher area coverage will result in better detection of new targets.

Table 1 demonstrates the numerical results in terms of the metrics above while executing policies solved using a conventional VI algorithm and a randomized DEM based VI algorithm respectively. Figure 10 shows the decision maker (sensor) path where regularly solved MDP is used for path selection. Squares represent sectors populated by targets, triangles represent sectors with undetected targets not taken into consideration in the current policy. Figure 11 shows the path in a randomized DEM based VI algorithm simulation. Area coverage is increased covering part of previously undetected targets.

Table 1 Opportunity index and state coverage index metrics for an original and randomized DEM-based VI algorithms

	Original VI algorithm	DEM-based VI algorithm	Improvement (%)
MOI	1	1.72	72
MAX OI	1	4	400
SCI	0.06	0.08	33.3
ACI	0.56	0.68	21.4

6 Distributed Data Fusion Architecture

The current section presents a solution for controlling the information flow in distributed data fusion architectures of various types – distributed track fusion, track fusion using tracklets, or distributed composite tracking [7]. In these aforementioned approaches, local and remote sensor tracks, tracklets, or AMRs are passed between platforms to be processed in a distributed data fusion process at each platform.

6.1 Issues in Distributed Data Fusion

We can distinguish between two cases of platforms interchanging data over communication network depicted in Figs. 12 and 13. In Fig. 12, the data from each platform should be passed to all the other platforms. In the first one, the same data coming from a certain platform are requested by other platforms as well, which causes redundancy and overloads the communication channels. Moreover, the task of providing tracking information from each node (platform) to every other node

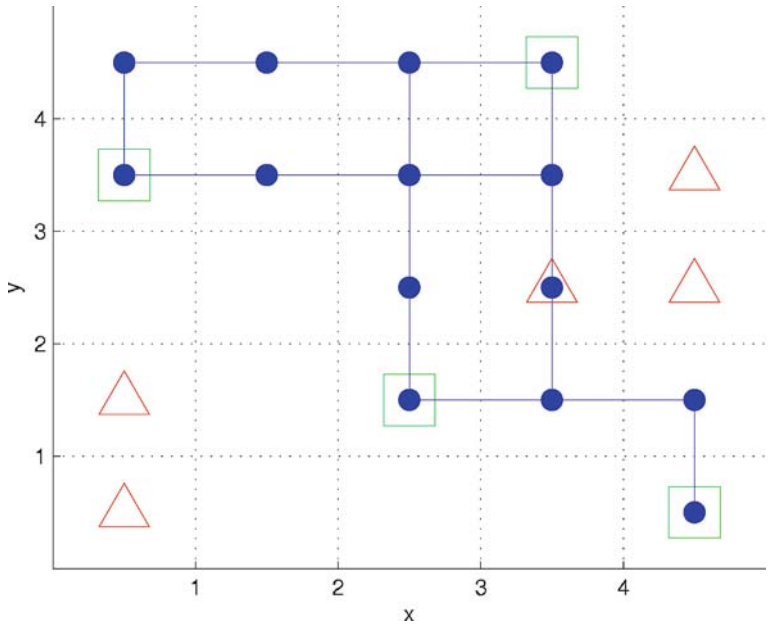


Fig. 10 Regular MDP is used for path selection. *Squares* represent sectors populated by targets. *Triangles* represent sectors with undetected targets not taken into consideration in the current policy. *Connected circles* show the decision maker path

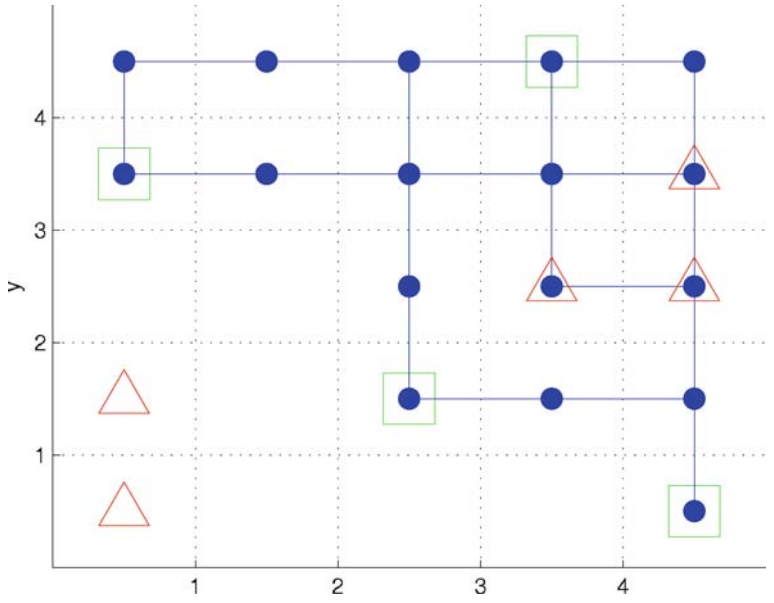


Fig. 11 Modified DEM based MDP is used for path selection. Area coverage is enhanced thus covering part of previously undetected targets

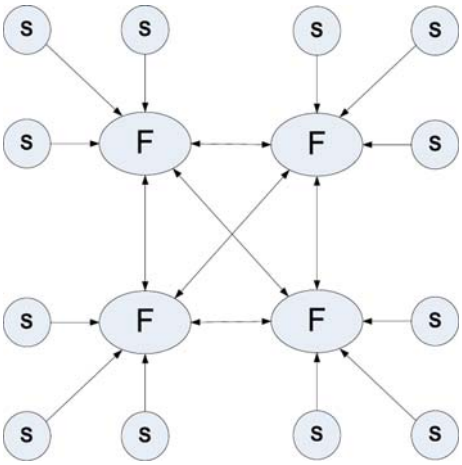


Fig. 12 Distributed data fusion example. The data from each platform should be passed to all the other platforms

may be just infeasible taking into account large number of the nodes in the network and significantly high data rates as opposed to the limitations of the communication channels capacity. The second approach is shown in Fig. 13. We are considering a node that requests specific information originating from other nodes. The figure depicts a special case in which $node_3$ requests information originating from $node_1$. The most straightforward (and, obviously, not the most optimal) solution would be requesting this information from the source, i.e., $node_1$. But as can be seen from the figure, the information originating from $node_1$ is also transmitted to nodes 2 and 4. It should be noted though that the information (e.g., tracks, tracklets or AMRs)

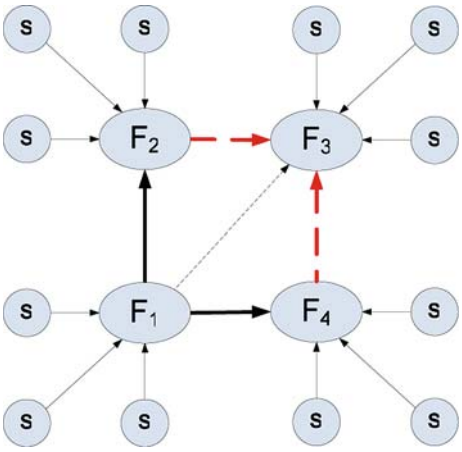


Fig. 13 Data flow in a distributed data fusion system. *Solid lines* represent existing data flows. *Dashed* and *dotted lines* represent candidate data flow channels to be decided

transmitted from a node to other nodes may differ in its characteristics and quality. Therefore the required information may be obtained from neighboring nodes as well, as depicted in Fig. 13, thus eliminating redundancy in the transmitted information, unnecessary load on the communication channels, time overhead in getting the information, higher refusal probability, etc. If we consider $node_3$'s request for data originating from $node_1$, the decision to be made in this case is which of the platforms – 1, 2 or 4 – should supply the requested information.

6.2 Data Fusion Control as a Decision-Based Approach

Choosing to get some specific information from one of the nearest nodes rather than from the information source itself introduces new decisions to be made, like, for example, which of the several available nodes the information should be requested from. For instance, $node_3$ may request $node_4$ to provide information originating from $node_1$, but this may not be the most optimal decision. The communication channel from $node_4$ to $node_3$ may not accommodate the required data rate. It is important to note that in such a case the optimal configuration is not achievable by a single decision or solution but is a multi-stage process.

We do not seek a solution at a particular time only before formulating a new optimization problem for the next step as such approach may be computationally inefficient. We are rather looking for an approach which, once having solved an optimization problem we have formulated, will provide the required action or decision for the next step taking into account the outcome of the previous one. It means that we are considering a situation where the decisions should be made sequentially or in a recursive manner. The result of each decision cannot be fully predicted, but can be accounted for using available statistical information before the next decision is made. The objective is to minimize the cost of our actions (or, alternatively, maximize a utility function) [18, 51]. Similarly to the approach applied for resource management in previous sections, the approach utilized for information control is based on MDP.

6.3 Data Lookup

When discussing the decision making process, we assumed that the information regarding the availability of the required information among the nodes is known. In fact, one of the fundamental problems that has to be addressed in order to apply the aforementioned decision process is in identifying all the nodes that possess the required information. The distributed data fusion architecture does not include any centralized control and consists of the nodes that are decentralized, potentially unreliable and heterogenous. Locating content in such system becomes therefore a complex task. Nodes may join, leave or become inaccessible, but this should not affect the operation of the total distributed system. One solution would be to use

one of the available distributed lookup protocols to efficiently identify the nodes that store the desired data. For this purpose we define a space of the keys K. To each node in the distributed system is assigned a key $n_i \in K$. We use a hash function h to map any particular source of information into the space K as well. For example, each sensor will be identified by a corresponding key $s_i \in K$. What is important to note is that the number of keys that a platform possesses equals the total number of sources of information it has, including the ones originating from the platform and the ones that a platform receives from others. The idea is then to assign each node to keep information about the identity of the information sources, values of the hash function of which lie in a certain proximity to the platform's key n_i. The resulting structure is called Distributed Hash Table (DHT). Several architectures of DHTs have been proposed: Chord [84], CAN [72], Kademlia [61], Tapestry [98] among others. Figure 14 demonstrates a node look-up process in the Chord DHT architecture [84]. The shortest available look-up time is $O(logN)$, where N is total number of the data sources in the distributed system.

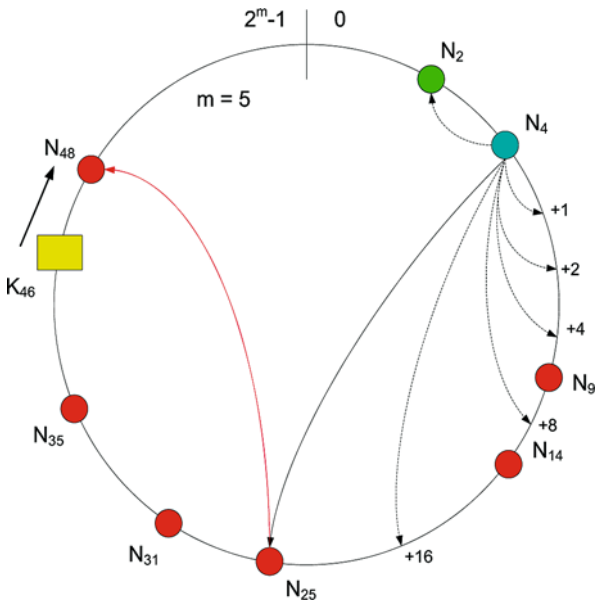


Fig. 14 Key lookup in chord

6.4 MDP-Based Multisensor Fusion for Multitarget Tracking

In this section, we will express the component elements of the Markov decision process in terms of the parameters of the optimization problem that we are facing. Here, the multisensor fusion problem for multitarget tracking is mapped into a collection of corresponding MDP problems, each one being solved by the corresponding node.

Each node will have a corresponding set of MDP parameters S,A,P,R reflecting the optimization problem this node needs to solve.The current section presents a solution for controlling the information flow between the platforms. The tracking data exchanged among those platforms can be of the following types: tracks, tracklets, or Associated Measurement Reports (AMRs). Subsequently, the tracking data that are passed between platforms are processed in a distributed data fusion process. We would like to engage an MDP-based decision mechanism to provide each platform with the required data for the distributed data fusion process while reducing redundancy in the information flow in the overall system. We will express below the components of the MDP for information flow control in terms of the parameters of the optimization problem that we are facing. Here, the multisensor fusion problem for multitarget tracking is mapped into a collection of corresponding MDP problems, each one being solved by the corresponding node. Each node will have a corresponding set of MDP parameters S,A,P,R reflecting the optimization problem this node needs to solve. Figure 15 shows the fusion control scheme in which a separate MDP is designated to each active track. A pseudo code for a fusion control algorithm is shown in Fig. 16.

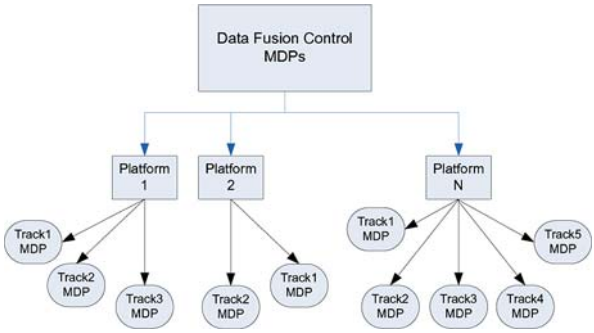


Fig. 15 Fusion control MDP scheme

function FusionControl

1. **Initialization:** Set initial parameters
2. **Policy Calculation:**
   a. Update list of initial and confirm tracks - T
   b. *For* $tr \subset T$
   If (NoPolicySet(tr) *Or* ReachedPolicyUpdate(tr))
   ChangePolicy:
      i. Calculate updated TransitionProbabilityMatrix P
      ii. Calculate updated RewardFunction R
      iii. Calculate MDP policy π_{tr}
3. **Data fusion control:**
   For each track tr request data from sources defined by $a_{tr}(s)$

Fig. 16 Fusion control algorithm

6.4.1 Set of States: *S*

The state of a node has the general structure depicted in Table 2. Each field takes one or more bits of digital information which are enumerated in the table. Below we describe the elements of the table:

- *Original node*. This field specifies the node from which the sought information originates.
- *Supplying node*. This field specifies all the nodes that currently receive the information originating from the *Original node*.
- *Data available*. This field indicates whether the requested data from one of the *Supplying nodes* is currently arriving and available.
- *Refusals*. This field contains the total number of refusals from the corresponding *Supplying node*.

Table 2 General structure of the node states

Original node	Supplying node	Status	Data bit	Bit #
$platform_{o1}$	$platform_{n1}$	Data avail.	0	0
		Refusals	1	1
			0	2
	$platform_{n2}$	Data avail.	1	3
		Refusals	1	4
			0	5
	$platform_{n3}$	Data avail.	1	6
		Refusals	0	7
			0	8
$platform_{o2}$	$platform_{n4}$	Data avail.	1	9
		Refusals	0	10
			1	11
	$platform_{n5}$	Data avail.	1	12
		Refusals	0	13
			0	14
	$platform_{n6}$	Data avail.	0	15
		Refusals	1	16
			0	17

6.4.2 Set of Actions: *A*

The set A contains all the possible actions that a node can take in order to specify the requested information sources in its next step of decision making. In our case, we have $n+1$ different actions in the set, expressing a request for information from any of the n nodes possessing the required information and an additional action of not requesting information at all. It should be noted that one of the existing trade-offs is requesting information from more than one source. That increases the probability of a positive outcome (the requested source providing the requested information) but at the same time increases the overall network load which may have a negative

impact on the requesting platform itself. Generally, it is acceptable in certain cases to request information from more than one source taking into account the relative unreliability of available sources. Obviously, when there is a large number of information sources, the policy of requesting information from multiple sources may be potentially harmful to both the whole distributed data fusion network and the requesting node itself.

6.4.3 Transition Probabilities: *P*

The transition probability matrix specifies the probability $P(s_{t+1}|s_t, a_t)$ of transition to a specific state s_{t+1}, provided the transition is done from another state s_t while performing a certain action a_t. The platform requesting data from other platforms has information regarding the holders of the required information as well as the knowledge of other circumstances that may influence successful reception of this information – for example, distributed network channels capacity, current load of the mentioned channels, the load of the nodes that have to supply information. Also, a specific node requesting information may have a priority rating index that may be different for various nodes.

For example, under the assumption that all other parameters such as network channels load, etc., are equal, $platform_i$ will have a higher chance to receive the required information from a certain node only because that node has higher priority rating index for $platform_i$. Other factors may also influence the probability P – for example, weather conditions at a certain node that may influence its chances to successfully transmit the requested information, hardware reliability, survival probability among many others. Also, a node may be able to learn empirically the characteristics of other nodes and, thereby, adjust transition probabilities [83]. All such information is eventually translated into the transition probability matrix. We can see this process as a mapping of all the relevant features of the external world into the transition probability matrix described above.

6.4.4 Real-Valued Reward Function on States: *R*

The vector R contains the values of the immediate rewards associated to being in a certain state. As mentioned above, we use information based objective function based on the Fisher information measure [4, 80]:

$$J = \sum_j \log |I_j(k|k)| = \sum_j \log |P_j(k|k)^{-1}| \tag{29}$$

where $P_j(k|k)$ is the posterior covariance matrix of the state vector corresponding to target j at time k. It is expressed as follows:

$$P_j(k|k)^{-1} = P_j(k|k-1)^{-1} + Y_j(k) \tag{30}$$

where $P_j(k|k-1)^{-1}$ is the predicted state information (inverse of the state prediction covariance matrix) and $Y_j(k)$ is the new information that is given by:

$$Y_j(k) = H(k, s, j)' R(k, s, j)^{-1} H(k, s, j) \tag{31}$$

where $H(k, s, j)$ is the measurement matrix and $R(k, s, j)$ is the measurement covariance matrix at the time step k corresponding to the sensor s from which the incoming measurement has originated and target j. Then, the expected updated information $I_j(k|k)$ can be expressed as follows:

$$I_j(k|k) = I_j(k-1|k) + H(k, s, j)' R(k, s, j)^{-1} H(k, s, j) \tag{32}$$

The reward associated with a measurement arriving from a remote sensor s after being successfully associated with one of the tracks of the platform is therefore expressed as the expected information gain corresponding to sensor from which the measurements originated. If there are N_s AMR's arriving from the same remote sensor s which are associated with the tracks of the receiving platform, then the expected information gain is given by

$$J_{N_s}(k, s) = \sum_{j=1}^{N_s(k)} \log |I_{s,j}(k|k)| - \log |I_j(k|k-1)| \tag{33}$$

where $I_j(k|k-1)$ is the predicted information matrix and $I_{s,j}(k|k)$ is the updated information matrix corresponding to target j.

7 Distributed Tracking Algorithms Implementing MDP-Based Data Fusion System

In this section, we present three distributed data fusion algorithms – associated measurement fusion, tracklet fusion and track-to-track fusion – to be applied with the MDP-based data fusion system.

7.1 Associated Measurements Fusion

In this architecture, the measurements, which are associated with the local tracks, are transmitted. Although the sensors may generate a large number of measurements, particularly in a dense clutter environment, only a few of them are associated with tracks and transmitted. Also, if the positions, measurement covariance and measurement matrix of all sensors connected to the distributed fusion network are available to each platform, then the corresponding quantities are not required to be communicated. Since the measurements are considered to be independent of the state dynamics, this architecture requires much less computation. When using

associated measurements fusion each platform performs the following steps: associating (either local or remote) measurements with the current tracks and updating the tracks using the associated measurements. If the associated measurements originated from the same platform they are transmitted to other platforms using the communication policy described above.

7.2 Track-to-Track Fusion

In this algorithm, we assume that the trackers on all the platforms start with the same information about tracks at time t_0. The tracks are updated by local trackers on each platform using the measurements received from the sensors on that platform up to time t_1. At this time the local tracks are broadcasted and each local tracker updates its tracks using the information received from the other local trackers. Since the tracks in the local trackers are started with the same initial information and the targets go through the same noisy transformation process the tracks obtained by different local trackers are correlated. The computation of the exact cross-covariance matrices in a track-to- track fusion system would hugely increase either computation or communication load [81]. This work uses the approximation proposed in [8, 27]. It assumes that at the time of computation of the cross-covariances between local tracks the covariances of all of the local tracks have already reached the steady state. The terms of the cross-covariance matrix $P^{\times}$ for a one dimensional tracking problem with state consisting of position and velocity $[x\ \dot{x}]$, are found as

$$p_{ij}^{\times} = \rho_{ij}\sqrt{p_{ii}^{l}p_{jj}^{k}}\ \ i,\ j \in \{1,2\} \tag{34}$$

where p_{ii}^{l} and p_{jj}^{k} denote elements i,j of covariance matrices P^{l} and P^{k} belonging to a certain track originating from platforms l and k; ρ_{ij} are unknown cross-correlation coefficients. There were proposed a number of ways to calculate cross-correlation coefficients [8, 27]. We used the following values of cross-correlation coefficients proposed in [8]: $\rho_{11} = 0.15, \rho_{12} = 0.25, \rho_{22} = 0.70$. For a 2-D tracking problem with the target state given by $[x\ \dot{x}\ y\ \dot{y}]$, the approximate cross-covariance matrix used in this work is given by

$$P^{\times} = \begin{bmatrix} p_{11,x}^{\times} & p_{12,x}^{\times} & 0 & 0 \\ p_{21,x}^{\times} & p_{22,x}^{\times} & 0 & 0 \\ 0 & 0 & p_{11,y}^{\times} & p_{12,y}^{\times} \\ 0 & 0 & p_{21,y}^{\times} & p_{22,y}^{\times} \end{bmatrix} \tag{35}$$

When using track-to-track fusion each platform performs the following steps: associating new measurements to the local tracks and updating them, periodically communicating the tracks to the other local trackers, computing the approximate cross-covariance matrices between tracks, associating the received tracks to local tracks, fusing the tracks received from the other trackers with the local tracks.

Denoting the mth track from platform i by T_m^i and the corresponding state and covariances by $\hat{x}_m^i$ and P_m^i, respectively, for a track set $\{T_{k1}^1, T_{k2}^2, \ldots, T_{k_n}^n\}$ the maximum likelihood estimate of the fused track states is given by

$$x_S^{\text{fused}} = (E' \bar{P}_S^{-1} E)^{-1} E' \bar{P}_S^{-1} \bar{x}_S \tag{36}$$

where S is the track set $\{T_{k1}^1, T_{k1}^2, \ldots, T_{k1}^n\}$, $E = [I_{n_x}\ I_{n_x}, \ldots, I_{n_x}]$ is $nn_x \times n_x$ matrix and n_x is the dimension of the state vector. Matrices $\bar{x}_S$ and $\bar{P}_S$ are given by (37) and (38), respectively,

$$\bar{x}_S = \begin{bmatrix} x_{k1}^1 \\ \vdots \\ x_{k_n}^n \end{bmatrix} \tag{37}$$

$$\bar{P}_S = \begin{bmatrix} P_{k_1}^1 & P_{k_1,k_2}^{1,2} & \ldots & P_{k_1,k_n}^{1,n} \\ P_{k_2,k_1}^{2,1} & P_{k_2}^2 & \ldots & P_{k_2,k_n}^{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{k_n,k_1}^{n,1} & P_{k_n,k_2}^{n,2} & \ldots & P_{k_n}^n \end{bmatrix} \tag{38}$$

The covariance matrix of the fused track is given by

$$P_S^{fused} = (E' \bar{P}_S^{-1} E)^{-1} \tag{39}$$

7.3 Tracklet Fusion

As tracklets are track data not cross-correlated with the common information among the platforms, for the data fusion purposes they can be treated like measurements and be associated to tracks of other platforms. That solves the problem of data synchronization typical to track-to-track fusion considered above. Since not all the correlation among the versions of the same track maintained by several platforms can be removed, such an approach is only reliable when dealing with targets having small maneuvering index. There were proposed a number of methods for calculating tracklets [40, 41]. In this chapter, we use an algorithm in which a tracklet is the state of a track decorrelated with the state of the same target at the time when the last tracklet was transmitted. For this decorrelation operation the older state requires to be predicted to the time of the more recent one. For tracklet computation, the state of the corresponding track must be observable from the measurements received after the last communication of a tracklet corresponding to the track. That is, at least two measurements are required if the target state vector contains the position and the velocity. Assuming that the last tracklet was transmitted at the time k, the track was last updated at time step $k + i$, and there are enough measurements between time

step k and time step $k+n$, the tracklet and the corresponding covariance matrix at time step $k+n$ are given according to [33] by (40) and (41), respectively. That is,

$$x_l(k+n) = \hat{x}(k+n|k) + P(k+n|k)[P(k+n|k) - P(k+n|k+i)]^{-1}(\hat{x}(k+n|k+i) - \hat{x}(k+n|k)) \quad (40)$$

$$P_l(k+n) = P(k+n|k)[P(k+n|k) - P(k+n|k+i)]^{-1}P(k+n|k) - P(k+n|k) \quad (41)$$

When using tracklet fusion, each platform performs the following steps: associating new measurements to the local tracks and updating latter, periodically computing the tracklets and communicating them to other platforms, associating the received tracklets to local tracks using 2-D association [17] and finally fusing the received tracklets with the local tracks.

8 Simulation Results

8.1 Resource Management

In this section, we present simulation results obtained from a 60 min simulation involving a number of sensors and targets under surveillance. The surveillance region dimensions were 42 by 42 km. The simulation included 30 targets and 5 sensors. The sensors in the simulations were of GMTI type. The state of the target j was of the following form:

$$\mathbf{x}^j = [x^j \;\; \dot{x}^j \;\; y^j \;\; \dot{y}^j]' \quad (42)$$

where x^j and y^j are 2-D Cartesian coordinates of the target j and $\dot{x}^j$ and $\dot{y}^j$ are its velocity components. We convert the original measurements obtained from the sensor s in the form $[r(k,s,j)\;\; \theta(k,s,j)\;\; \dot{r}(k,s,j)]'$, where $r(k,s,j)$, $\theta(k,s,j)$ and $\dot{r}(k,s,j)$ are the range, azimuth angle and range rate of the target j supplied by sensor s at time t_k, respectively, to the measurement vector of the following form

$$\mathbf{z}^j = [x^j \;\; y^j \;\; \dot{r}^j]' \quad (43)$$

where $\dot{r}^s$ is the speed of target j. The original measurement vector is assumed to contain independent additive Gaussian noise with variances denoted as σ_r^2, σ_θ^2 and $\sigma_{\dot{r}}^2$ respectively. The measurement covariance matrix $R(k,s,j)$ corresponding to the converted measurement is given by [9]

$$R(k,s,j) = \begin{bmatrix} R_{1,1} & R_{1,2} & 0 \\ R_{1,2} & R_{2,2} & 0 \\ 0 & 0 & \sigma_{\dot{r}}^2 \end{bmatrix} \quad (44)$$

The elements of R are

$$R_{1,1} = r^2\sigma_\theta^2 \sin^2\theta + \sigma_r^2 \cos^2\theta \tag{45}$$

$$R_{2,2} = r^2\sigma_\theta^2 \cos^2\theta + \sigma_r^2 \sin^2\theta \tag{46}$$

$$R_{1,2} = (\sigma_r^2 - r^2\sigma_\theta^2) \sin\theta \cos\theta \tag{47}$$

The overall surveillance region has been divided into a number of clusters equal to the number of the sensors deployed. The sensors move at a constant speed of 75 m/s. In the simulation one point track initialization [97] is applied. The tracker consists of a Kalman filter with auction-based [17] assignment for measurement-to-track association. A white noise acceleration model is assumed for the targets with process noise standard deviation of 1 m/s^2. The measurement noise standard deviations are $\sigma_r = 10$ m, $\sigma_\theta = 10^{-3}$ rad and $\sigma_{\dot{r}} = 1$ m/s. The number of false alarms in each sector follows a Poisson distribution with mean 0.1 and the false measurements are uniformly distributed in the sector. Figure 17 shows the trajectories of all targets. Shaded squares designate currently scanned sectors. Figure 18 shows the trajectories of the five sensors and the targets. Each sensor is responsible for a designated area and during the policy execution time tracks the targets in this area. For the demonstrated simulation scenario the total surveillance area was divided into five equal overlapping regions. During the current policy execution sensors record the information regarding the targets they track. Occasionally, sensors may spot targets located in the regions responsibility on which belongs to other sensors. In this

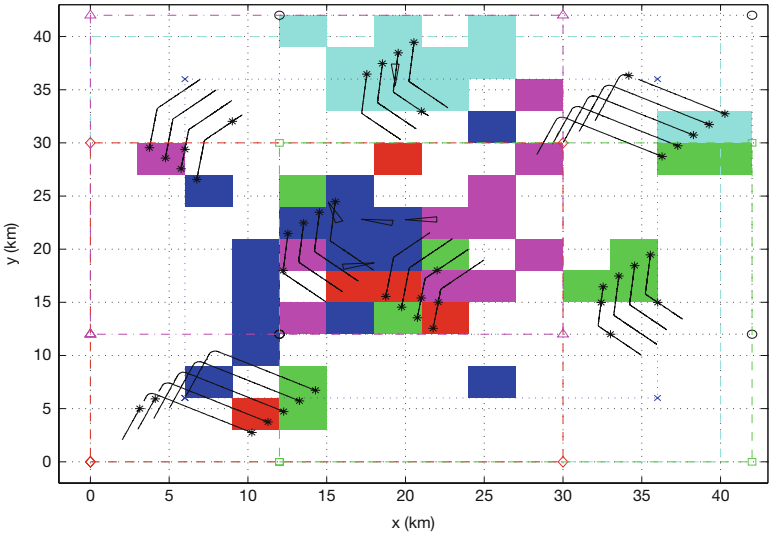


Fig. 17 The trajectories of the 30 targets. *Shaded* squares designate currently scanned sectors. Current positions of the sensors are designated as *triangles*

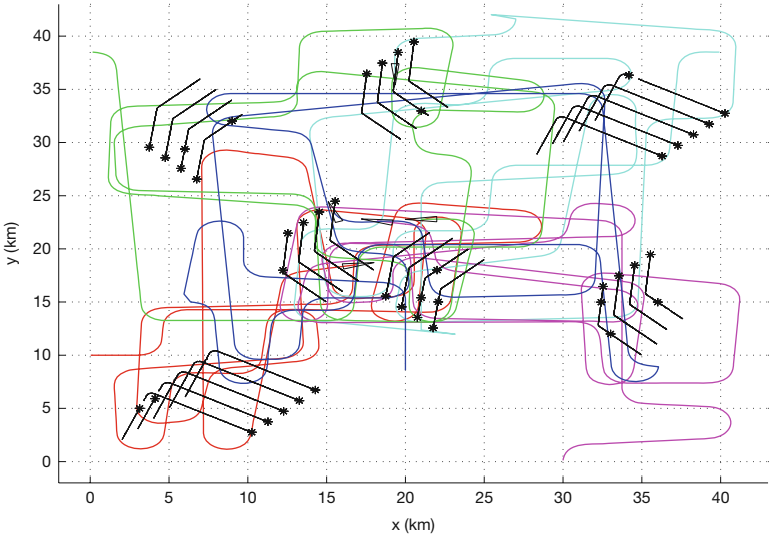


Fig. 18 The trajectories of the 5 sensors and the 30 targets between two consecutive policy updates

case during the policy execution this information is transferred to the corresponding sensor. When performing a policy update, each sensor takes into consideration the targets that have been spotted by it, by other sensors as well as the targets potentially residing in the areas that have not been scanned during the current policy execution. Then, each sensor forms a list of the sectors, either populated by confirmed targets or unscanned ones and using information-based objective function approach described above, reward of each MDP sector is calculated for the next policy. Figure shows 21 the scan decisions for one of the sensors during four consecutive cycles. Ten sectors were chosen for scan during each cycle. As can be seen from Fig. 18, sensors do not always succeed to move exactly as planned. One of the reasons of it in the current simulation is as follows. When a sensor needs to change its course, it starts a coordinated turn in corresponding direction. As a result, at the end of the coordinated turn, the direction of the sensor may deviate from the planned one. Thanks to the MDP approach any outcome will have the corresponding optimal action to take. Controlled by a corresponding MDP, a sensor can perform a coordinated turn with angular turn rate of up to 0.05 rad/s. The targets move at different speeds. They include maneuvering targets and move-stop-move ones. The scan time used in this simulation is 5 s. Figure 19 show four snapshots of the simulation at the times of 1, 5, 10 and 15 min respectively. The sensors are depicted as triangles with the sharp angles pointing the direction of their movement. The lines represent the tracks and the stars show the last updates position of the detected targets. The shaded sectors represent surveillance sectors that have been scanned during the last scan. All the targets were belonging to one of the seven different groups. Figure 20 shows 50 Monte Carlo run position and velocity RMSE results of the tracks corresponding to the targets from group one and two. Table 3 shows the summary of the performance

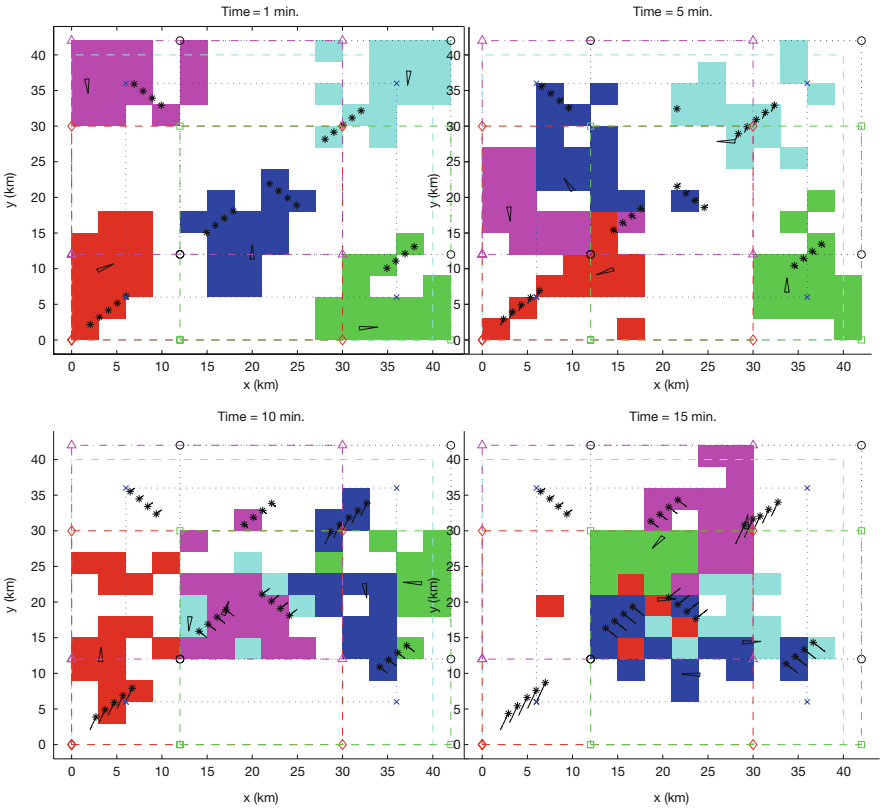


Fig. 19 Snapshots depicting sensor locations and target tracks. The sensors are depicted as triangles. The *lines* represent the tracks and the *stars* show the last updates position of the detected targets. The shaded sectors represent surveillance sectors that have been scanned during the last scan

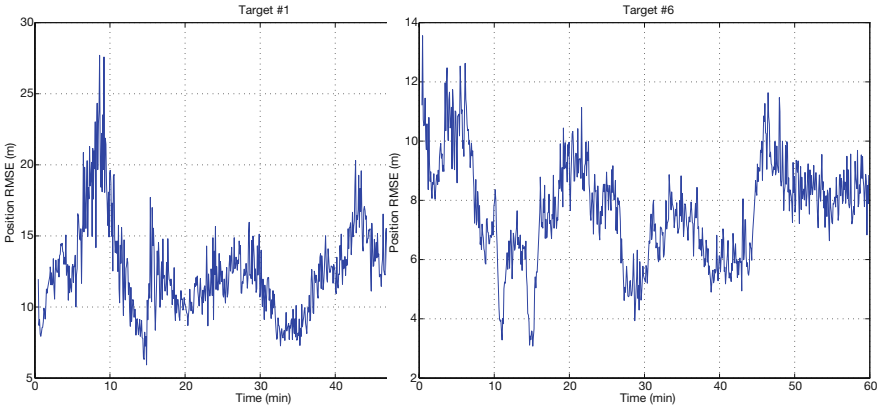


Fig. 20 The position and velocity RMSE of target 1 (group 1) and target 6 (group 2), respectively

Table 3 Performance metrics in single-layer MDP hierarchy

Group no.	Tar no.	Position RMSE (m)	Velocity RMSE (m/s)
1.	1	12.7	1.5
2.	6	7.6	1.2
3.	10	7.8	1.2
4.	14	9.5	1.3
5.	19	12.6	1.5
6.	23	9.8	1.3
7.	27	8.9	1.3

metrics for all the groups in the single-layer MDP hierarchy. The results demonstrate the ability of the proposed approach to maintain an acceptable level of accuracy for different targets.

Figures shows 21 scan decisions for one of the sensors during four consecutive cycles. Ten sectors were chosen for scan during each cycle.

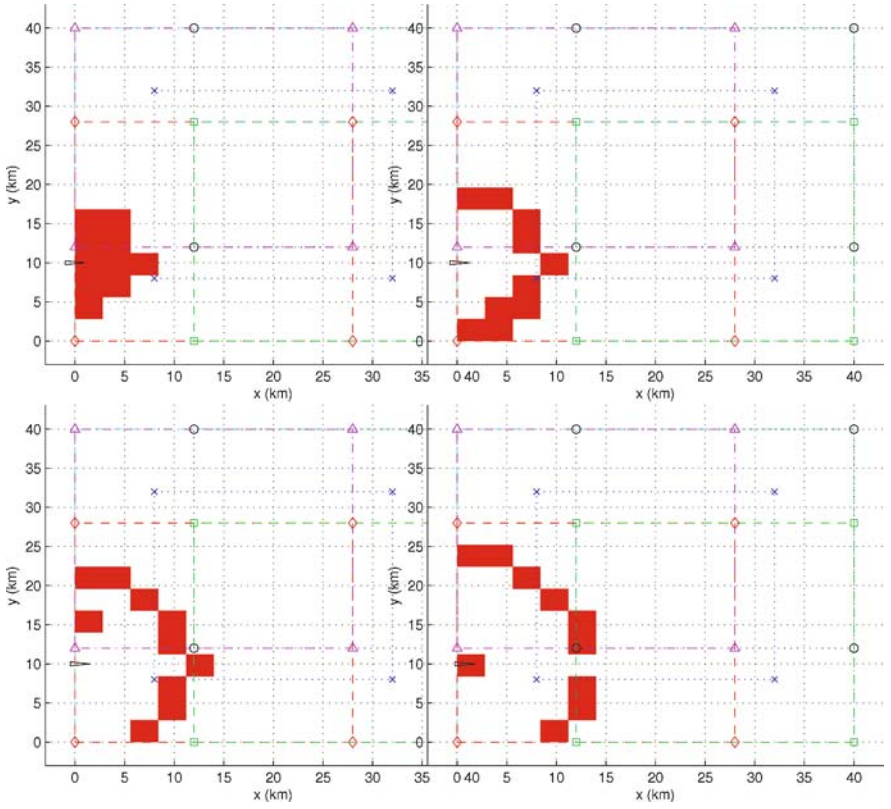


Fig. 21 Scan decisions for one of the sensors during four consecutive cycles. Ten sectors were chosen for scan during each cycle

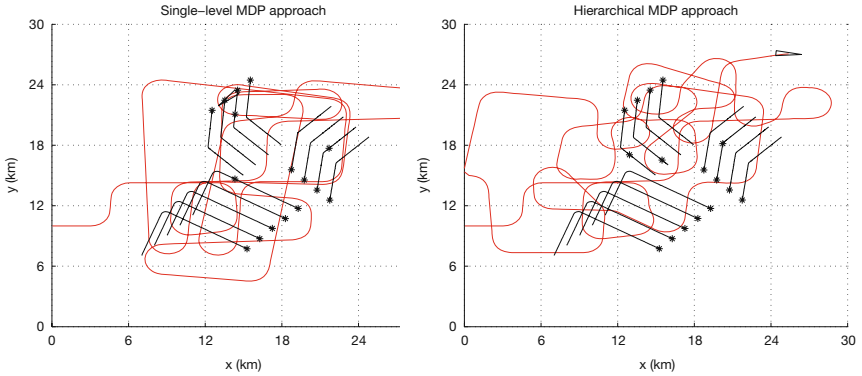


Fig. 22 The trajectories of the first sensor and the 13 targets. Single-level MDP approach (*left*) and two-level hierarchical MDP approach (*right*)

Figure 22 shows the path of the first of the five sensors as well as the 13 targets when a single-level and a two-level MDP approaches, respectively, were employed. As can be seen from Fig. 22(left), sensors do not always succeed to move exactly as planned, which is particularly visible in a single-level MDP approach. Thanks to the MDP approach, any outcome will have the corresponding optimal action to take. When using multi-level MDP approach, fine-tuning that takes place at lower levels of MDP processes corrects navigation errors at higher levels.

Table 4 shows 50 Monte Carlo run position RMSE results of the tracks corresponding to the targets 1 to 13 from the three groups for both methods. The results demonstrate the ability of the proposed approach to maintain an acceptable level of accuracy for different targets. The two-level MDP approach demonstrated higher level of accuracy than the single-level one. Further, we present simulation results obtained from a 60 min simulation involving four sensors and ten targets under surveillance. The overall surveillance region has been divided into four clusters, equal to the number of the sensors deployed. The sensors move at a constant speed of 40 m/s. Tables 5 and 6 show the summary for position and velocity RMSE respectively for all the targets. As can be seen from the results, the modified algorithm which included DEM approach demonstrated an average accuracy improvement in position and velocity RMSE of the targets under surveillance increasing State Coverage Index by more than 30% resulting in better detection of new targets.

8.2 Information Management

The following test-case will compare the performance of the approach among three different data fusion methods, namely, associated measurements fusion, track-to-track fusion and tracklets fusion.

The simulated cluster contains 5 platforms and 8 sensors. Each sensor belongs to a specific platform associated with a corresponding tracker. Sensors 1 and 2 are

Table 4 Performance metrics comparison between single-layer and multi-layer MDP hierarchy

	Position RMSE (m)	
Target no.	Single-layer	Multi-layer
1.	11.9	8.7
2.	10.6	7.9
3.	10.0	7.2
4.	9.4	7.1
5.	9.6	7.3
6.	10.6	7.6
7.	9.8	7.0
8.	10.2	7.1
9.	11.6	8.2
10.	10.1	7.2
11.	9.3	6.6
12.	9.6	6.9
13.	11.1	8.1
Average improvement		27.6%

Table 5 Performance metrics summary for position RMSE

	Position RMSE (m)		
Target no.	Original algorithm	DEM-based algorithm	Improvement (%)
1.	19.2	17.7	8.8
2.	18.2	14.6	25.0
3.	13.9	15.4	−9.7
4.	16.8	17.5	−4.0
5.	20.2	13.9	45.9
6.	21.3	14.4	47.2
7.	14.8	16.5	−10.7
8.	18.4	11.6	58.1
9.	17.4	13.0	34.4
10.	15.9	14.7	8.2
Average improvement			20.3

Table 6 Performance metrics summary for velocity RMSE

	Velocity RMSE (m)		
Target no.	Original algorithm	DEM-based algorithm	Improvement (%)
1.	1.6	1.6	−3.9
2.	1.7	1.7	0.6
3.	1.8	1.6	12.9
4.	1.8	1.5	14.8
5.	1.3	1.0	26.6
6.	1.6	1.2	26.2
7.	1.5	1.3	20.9
8.	1.8	1.5	18.3
9.	1.7	1.6	9.2
10.	1.5	1.7	−9.2
Average improvement			11.7

connected to $tracker_1$, sensors 3 and 4 to $tracker_2$, sensor 5 to $tracker_3$, sensors 6 and 7 to $tracker_4$ and sensor 8 is connected to $tracker_5$. The sensors are located at $x_s^i = [65, 155]', [80, 140]', [10, -65]', [30, -40]', [-80, 20]', [-70, 130]', [-40, 150]', [20, 150]'$ km, respectively. The original measurements are obtained from the sensor s in the form $[r(k, s, j) \quad \theta(k, s, j)]'$ where $r(k, s, j)$ and $\theta(k, s, j)$ are the range and the azimuth angle of the target j supplied by sensor s at time t_k, respectively. The standard deviations of range r and bearing Θ of the sensors are $[\sigma_r, \ \sigma_\Theta] = [40, 2.5], [35, 2], [52.5, 3.5], [30, 2.5], [45, 4.5], [52.5, 3.5], [52.5, 3.5], [45, 4.5]$ (m, mrad), respectively. The sampling intervals of the sensors are 2.5, 3.5, 2, 3, 1.5, 2, 2, 1.5 s, respectively. The false alarms are uniformly distributed in the coverage areas of the sensors with the number of false alarms Poisson distributed with means of 40, 40, 100, 50, 50, 40, 50, 50, respectively. The simulation included two closely spaced targets. The scenario of the two target movements includes several constant velocity stages interleaved with coordinated turn maneuvers performed at rates $|\omega| = 4\ ^\circ/\text{s}$. The initial positions of the targets are $[\xi_j, \eta_j] = [5, 68.6]$ km, $[5, 69.1]$ km, respectively, and the initial targets velocity is 300 m/s. Figure 23 shows the coverage areas of the sensors and the targets trajectory. A white noise acceleration model is assumed for the targets with process noise standard deviation σ_υ of 15 m/s^2. The state of the target j is of the form

$$\mathbf{x}^j = [x^j \quad \dot{x}^j \quad y^j \quad \dot{y}^j]' \tag{48}$$

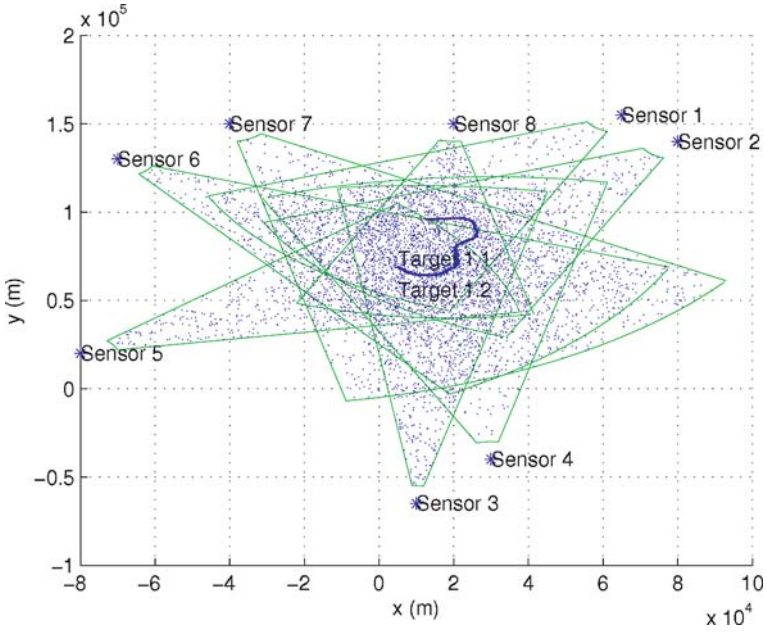


Fig. 23 Test-case C. Coverage area of the sensors and targets trajectory. Eight sensors belonging to five different platforms are present

where x^j and y^j are the x and y Cartesian coordinates of the target j and $\dot{x}^j$ and $\dot{y}^j$ its x and y velocity components, respectively. We convert the original measurements obtained from the sensor s in the form

$$z = [r(k, s, j) \;\; \theta(k, s, j)]' \tag{49}$$

where $r(k, s, j)$, $\theta(k, s, j)$ and $\dot{r}(k, s, j)$ are the range and the azimuth angle of the target j supplied by sensor s at time t_k, respectively, to the measurement vector of the form

$$\mathbf{z}^j = [x^j \;\; y^j]' \tag{50}$$

using the standard coordinate conversion [9]:

$$\mathbf{z}^j = [r^j \cos \Theta^j \quad r^j \sin \Theta^j]' \tag{51}$$

The measurement covariance matrix R corresponding to the converted measurement is given by [9]

$$R_L = \begin{bmatrix} R_L^{11} & R_L^{12} \\ R_L^{12} & R_L^{22} \end{bmatrix} \tag{52}$$

The elements of R_L are

$$R_L^{11} = r^2 \sigma_\Theta^2 \sin^2 \Theta + \sigma_r^2 \cos^2 \Theta \tag{53}$$

$$R_L^{12} = (\sigma_r^2 - r^2 \sigma_\theta^2) \sin \Theta \cos \Theta \tag{54}$$

$$R_L^{22} = r^2 \sigma_\Theta^2 \cos^2 \Theta + \sigma_r^2 \sin^2 \Theta \tag{55}$$

In the simulation two-point track initialization is applied [9]. Table 7 shows the time-averaged position and velocity RMSE after 100 Monte Carlo runs for distributed data fusion system based on associated measurements fusion. The performance of the mode with no information sharing is the worst. The performance of the mode in which all the AMRs are transmitted among all the sensors is the best. We can see that the performance of the MDP controlled mode is much better than that of the first mode but still worse than that of the second mode. In MDP controlled mode the maximal number of sensors that could transmit AMRs to $platform_3$ was first restricted to two sensors and then to four sensors. The information flow in the system was reduced, which in many cases may justify the reduction in performance. In many cases the situation in which all the platforms transmit AMRs to all the other platforms is infeasible. The performance may be increased though by increasing the maximal allowed number of sensors transmitting remote AMRs to any given platform. Table 7 also includes the results of an additional simulation in which there were always picked two sources having the best characteristics – sources number two and six. The results were inferior to those of an MDP-based approach.

Table 7 Performance metrics summary, associated measurements fusion

	Position RMSE (m)		Velocity RMSE (m/s)	
Communication mode	$Target_1$	$Target_2$	$Target_1$	$Target_2$
All AMRs shared	45.5	45.1	32.4	27.8
No AMRs shared	703.3	664.8	100.7	70.4
MDP controlled, 4 sources	65.5	62.1	35.1	34.9
MDP controlled, 2 sources	72.6	71.1	36.5	36.9
2 best sources	94.4	98.6	45.9	43.3

Table 8 Performance metrics summary, track-to-track fusion

	Position RMSE (m)		Velocity RMSE (m/s)	
Communication mode	$Target_1$	$Target_2$	$Target_1$	$Target_2$
All tracks shared	268.2	272.6	60.8	65.9
No tracks shared	993.2	814.5	121.7	103.0
MDP controlled	297.3	301.6	65.9	71.3

Table 9 Performance metrics summary, tracklet fusion

	Position RMSE (m)		Velocity RMSE (m/s)	
Communication mode	$Target_1$	$Target_2$	$Target_1$	$Target_2$
All tracks shared	215.1	204.7	58.4	59.8
No tracks shared	749.4	750.4	104.2	104.6
MDP controlled	268.8	250.8	58.8	65.4

Table 8 shows the time-averaged position and velocity RMSE after 100 Monte Carlo runs for the distributed data fusion system based on track-to-track fusion. Table 9 shows the time-averaged position and velocity RMSE after 100 Monte Carlo runs for the distributed data fusion system based on tracklet fusion.

9 Communication Data Rate and Computational Load in Distributed Tracking Algorithms

The focus of this section will be on the communication cost of the aforementioned data fusion types. One of the main advantages of network-centric approaches, where the sensors may be located at significant distances from one another, versus platform-centric one, where the sensors are located in close vicinity from one another, is in achieving a higher precision and robustness by the separation of sensors located on distinct platforms [7]. The sharing of tracking data results in tracking results that are in general more complete, more accurate, and more timely than those obtained in the case of a single sensor or a single platform. In many cases, sharing the data provides tracking information that is unavailable for many of the participating platforms or may be obtained at significantly lower quality. One of the main barriers in network-centric tracking is the transfer of tracking data from one platform to another via a communication link capable of transferring the required

volumes of information. This communication load should be taken into consideration in the design of the tracking system and cannot be ignored. As mentioned before, the task of providing tracking information from one platform to anther node may be infeasible because of the limitations of the communication channel capacity. The decision mechanism, presented in this chapter, provides each platform with the required data for the distributed data fusion process subject to the available channel capacities and reducing redundancy in the information flow in the overall system. Below we will address the communication requirements [7] for each of the distributed data fusion approaches we are using, namely, associated measurement fusion, tracklet fusion and track-to-track fusion, using notations and assumptions described originally in [7].

When associated measurement reports (AMR) arriving from other platforms as well as from the local sensors are used to form a common tracking picture, the data rate Ψ for a platform participating in the data exchange process is given by

$$\Psi_{AMR} = \frac{N_{\text{meas/scan}}}{\delta_{\text{scan}}} n_{\text{data}} \tag{56}$$

where δ_{scan} is the tracking interval, $N_{\text{meas/scan}}$ is the number of associated measurements per scan and n_{data} is the allocated number of bits used by a system to transmit the data between the platforms:

$$n_{\text{data}} = n_{\text{meas}} + n_{\text{meas\_acc}} + n_{\text{number}} + n_{\text{time}} \tag{57}$$

where n_{meas}, $n_{\text{meas\_acc}}$, n_{number},and n_{time} are the bit allocation numbers for a transmitted measurement, its related accuracy information (unique components of a covariance matrix), track number, and the measurement's time, respectively. The bit allocation for AMR-based data consisting of range, bearing and elevation is

$$n_{\text{meas}} = n_{\text{range}} + n_{\text{bearing}} + n_{\text{elevation}} \tag{58}$$

$$n_{\text{meas\_acc}} = n_{\text{r\_acc}} + n_{\text{b\_acc}} + n_{\text{e\_acc}} \tag{59}$$

The resulting data rate is therefore

$$\begin{aligned} \Psi_{\text{AMR}} = \frac{N_{\text{meas/scan}}}{\delta_{\text{scan}}} (& n_{\text{range}} + n_{\text{bearing}} + n_{\text{elevation}} + n_{\text{r\_acc}} \\ & + n_{\text{b\_acc}} + n_{\text{e\_acc}} + n_{\text{number}} + n_{\text{time}}) \end{aligned} \tag{60}$$

In the case of track-to-track fusion, the data rate is given by

$$\Psi_{\text{tracklet}} = \frac{N_{\text{meas/scan}}}{\delta_{\text{scan}}} n_{\text{data}} \tag{61}$$

For the state of the target of the form

$$\mathbf{x} = [x \;\; \dot{x} \;\; y \;\; \dot{y} \;\; z \;\; \dot{z}]' \tag{62}$$

where x, y and z are the Cartesian coordinates of the target and $\dot{x}$, $\dot{y}$ and $\dot{z}$ are its x, y and z velocity components, respectively, n_{data} is given by

$$n_{\text{data}} = 3n_{\text{pos}} + 3n_{\text{vel}} + n_{\text{number}} + n_{\text{time}} + 21n_{\text{acc}} \tag{63}$$

where n_{acc} is the number of bits allocated for the 21 unique elements of the state covariance matrix. The resulting data rate is given by

$$\Psi_{\text{track2track}} = \frac{N_{\text{meas/scan}}}{\delta_{\text{scan}}}(3\, n_{\text{pos}} + 3\, n_{\text{vel}} + n_{\text{number}} + n_{\text{time}} + 21\, n_{\text{acc}}) \tag{64}$$

The data rate in the case of track-to-track fusion is similar to the track-to-track one and is given by

$$\Psi_{\text{tracklet}} = \frac{N_{\text{meas/scan}}}{\delta_{\text{scan}} N_{\text{meas/tracklet}}} n_{\text{data}} = \frac{\Psi_{\text{track2track}}}{N_{\text{meas/tracklet}}} \tag{65}$$

where $N_{\text{meas/tracklet}}$ is the measurements number per one tracklet. The resulting data rate is given by

$$\Psi_{\text{tracklet}} = \frac{N_{\text{meas/scan}}}{\delta_{\text{scan}} N_{\text{meas/tracklet}}}(3\, n_{\text{pos}} + 3\, n_{\text{vel}} + n_{\text{number}} + n_{\text{time}} + 21\, n_{\text{acc}}) \tag{66}$$

9.0.1 Communication and Computational Load Results

Table 10 shows the resulting communication and computational load results for each of the distributed data fusion approaches.

The tables features different metrics for the data fusion methods used – data rate Ψ for a platform participating in the data exchange process, computation time t_{comp} for the overall simulated time of 200 s, and averaged position RMSE $\varepsilon_{\mathbf{x}\ \text{RMSE}}$. Also, the table contains the combinations of the metrics above, for example the product of required computation time and required data rate – Ψt_{comp}. For all of the metrics and their combinations the higher range means less favorable results, which makes the comparison easier. The lowest data rate was achieved when using MDP based associated measurements data fusion, followed by MDP-based track-to-track or tracklet data fusion simulations. The highest one was registered during track-to-track or tracklet based data fusion simulations when all the data were shared, both of which used the same time interval between data transmissions – 10 s. In terms of computation time, the best results were achieved when using tracklet based data fusion method when no data were shared among the platforms. The highest computation time resulted in the track-to-track MDP-based data fusion simulation. The best accuracy was achieved when using AMR data fusion sharing all the information, closely followed by MDP-based AMR data fusion results. The worst accuracy was achieved in track-to-track data fusion simulation when no data were shared

Table 10 Communication and computational load

Fusion method/metrics	Ψ (bit/s)	t_{comp} (s)	$\varepsilon_{\mathbf{x}\ \text{RMSE}}$ (m)	$\Psi\,\varepsilon_{\mathbf{x}\ \text{RMSE}}$	Ψt_{comp}	$t_{\text{comp}}\varepsilon_{\mathbf{x}\ RMSE}$	$\Psi\,\varepsilon_{\mathbf{x}\ \text{RMSE}} t_{\text{comp}}$
AMR fusion, all tracks shared	587.8	14.2	45.3	2.6×10^4	8.3×10^3	6.4×10^2	3.7×10^5
AMR fusion, no tracks shared	0	9.7	684.0	0	0	6.6×10^3	0
AMR fusion, MDP controlled	310.4	13.6	63.8	1.9×10^4	4.2×10^3	8.7×10^2	2.7×10^5
Track-to-track fusion, all tracks shared	645.6	10.9	270.4	1.7×10^5	7.0×10^3	2.9×10^3	1.9×10^6
Track-to-track fusion, no tracks shared	0	9.3	903.8	0	0	8.4×10^3	0
Track-to-track fusion, MDP controlled	338.9	13.8	299.4	1.0×10^5	4.6×10^3	4.1×10^3	1.4×10^6
Tracklet fusion, all tracks shared	645.6	9.2	209.9	1.3×10^5	5.9×10^3	1.9×10^3	1.2×10^6
Tracklet fusion, no tracks shared	0	8.5	749.9	0	0	6.4×10^3	0
Tracklet fusion, MDP controlled	338.9	13.1	259.8	8.8×10^4	4.4×10^3	3.4×10^3	1.1×10^6

among the platforms. The best $\Psi\varepsilon_{\mathbf{x}\,\mathrm{RMSE}}$ metrics was achieved when using MDP based associated measurements data fusion and the worst when using track-to-track data fusion when all the data were shared among the platforms. In summary, associated measurements with MDP-based data fusion yielded the best results in the majority of the metrics categories and track-to-track data fusion with all tracks shared gave the worst results in most of them. That could be explained in part by the need to transmit the elements of covariance matrix combined with the fact that as the computation of the exact cross-covariance matrices in a track-to-track fusion system would hugely increase either computation load, approximation techniques are used to compute them. Similar reasons can be stated for the tracklet data fusion as both covariance matrices need to be transmitted. Also, due to the common process noise between the tracks, not all correlations between the various versions of the same track, maintained by different local trackers, can be removed by using the tracklets. Therefore, this algorithm is only an approximation and it is reliable only for targets with small maneuvering index.

10 Conclusions

In this chapter, we presented a solution for one of the main problems in network-centric tracking – decentralized information sharing among the platforms participating in the distributed data fusion. We proposed a Markov Decision Process-based algorithm for controlling the flow of information in a network-centric data fusion architecture. The proposed decision process-based algorithm for controlling the information flow in a network-centric data fusion architecture utilizes the approach of using information based objective function as one of the components in the formulated optimization problem. We demonstrated that the approach led to a substantial reduction in data flow volumes, which also incurred a certain price in terms of reduction in performance. In order to minimize communication among the nodes, node lookup is performed using a decentralized lookup substrate. A proposed decision mechanism provides the platforms, which are decentralized, heterogenous and potentially unreliable, with the required data for the distributed data fusion process while reducing redundancy in the information flow in the overall system. The node lookup is performed in $O(\log(N))$ and the computational complexity of solving the MDP is P-complete, which shows that the proposed method is computationally attractive for distributed data fusion applications.

The work applied the suggested approach to three different data fusion methods, namely, associated measurements fusion, track-to-track fusion and tracklet fusion. As one of the main barriers in network-centric tracking is the transfer of tracking data from one platform to another, the analysis included communication load of the three data fusion methods. This communication load should be taken into consideration in the design of the tracking system and cannot be ignored. As mentioned, the task of providing tracking information from one platform to anther node may be infeasible because of the limitations of the communication channel capacity. The

lowest data rate was achieved when using MDP based associated measurements data fusion, followed by MDP-based track-to-track or tracklet data fusion simulations. The highest one was registered during track-to-track or tracklet based data fusion simulations when all the data were shared, both of which used the same time interval between data transmissions. In terms of computation time, the best results were achieved when using tracklet based data fusion method when no data were shared among the platforms. The highest computation time resulted in the track-to-track MDP-based data fusion simulation. The best accuracy was achieved when using AMR data fusion sharing all the information, closely followed by MDP-based AMR data fusion results. The worst accuracy was achieved in track-to-track data fusion simulation when no data were shared among the platforms. The best $\Psi\varepsilon_{\mathbf{x}\ \mathrm{RMSE}}$ metrics was achieved when using MDP-based associated measurements data fusion and the worst when using track-to-track data fusion when all the data were shared among the platforms. In summary, associated measurements with MDP-based data fusion yielded the best results in the majority of the metrics categories and track-to-track data fusion with all tracks shared gave the worst results in most of them. That could be explained in part by the need to transmit the elements of covariance matrix combined with the fact that as the computation of the exact cross-covariance matrices in a track-to-track fusion system would hugely increase either computation load, approximation techniques are used to compute them. Similar reasons can be stated for the tracklet data fusion as both covariance matrices need to be transmitted. Also, due to the common process noise between the tracks, not all correlations between the various versions of the same track, maintained by different local trackers, can be removed by using the tracklets. Therefore, this algorithm is only an approximation and it is reliable only for targets with small maneuvering index.

In this chapter we also provided an efficient solution the problem of collaborative sensor management using Markov Decision Processes. We presented an altered version of a classical Value Iteration algorithm to calculate the optimal policy for Markov Decision Processes used to address the problem of collaborative sensor management and data fusion for multitarget tracking. Dynamic Element Matching algorithms were successfully used as a core element in the modified algorithm. A number of new introduced performance metrics, such as Mean Opportunity Index, Maximal Opportunity Index and Area Coverage Index, verify the effectiveness of a policy, especially for quantifying the impact of the modified DEM-based Value Iteration VI algorithm on an MDP policy. The modified algorithm, applied to control a group of sensors carrying out surveillance over a region that included a number of moving targets, demonstrated an average accuracy improvement expressed in approximately 20% reduction in position RMSE of the targets under surveillance. State Coverage Index (SCI) was increased by more than 30%. The proposed method demonstrated robust performance while guaranteeing polynomial computational complexity resulting in better detection of new targets. The authors also present multi-level hierarchy of MDPs controlling each of the sensors. Each level in the hierarchy solves a problem at a different level of abstraction providing sensors with ability to navigate with a higher precision in a densely populated regions.

References

1. D. Akselrod, A. Fish, and O. Yadid-Pecht, "A mixed signal enhanced WTA tracking system via 2-D dynamic element matching", *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'2002)*, Phoenix, AZ, May 26–29, 2002.
2. D. Akselrod, C. V. Goldman, A. Sinha, and T. Kirubarajan, "Collaborative sensor management for multitarget tracking using decentralized Markov decision processes", *SPIE Defense and Security Symposium*, Orlando, FL, April 17–21, 2006.
3. D. Akselrod, A. Sinha, C. V. Goldman, and T. Kirubarajan, "Efficient control of information flow for distributed multisensor fusion using Markov decision processes", *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.
4. D. Akselrod, A. Sinha, and T. Kirubarajan, "Collaborative distributed data fusion architecture using multi-level Markov decision processes", *Proceedings of the 10th International Conference on Information Fusion*, Quebec, Canada, July 9–12, 2007.
5. D. Akselrod, A. Sinha, and T. Kirubarajan, "Collaborative hierarchical Markov decision processes based distributed data fusion and collaborative sensor management for multitarget multisensor tracking applications", *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC 2007)*, Montreal, Canada, October 7–10, 2007.
6. D. Akselrod, A. Sinha, and T. Kirubarajan, "Collaborative distributed sensor management for multitarget tracking using hierarchical Markov decision processes", *SPIE Optics & Photonics Symposium*, San Diego, CA, August 26–30, 2007.
7. Y. Bar-Shalom and W. D. Blair, *Multitarget-Multisensor Tracking: Applications and Advances*, Volume III, Artech House, Norwood, MA, 2000.
8. Y. Bar-Shalom and H. Chen, "Multisensor track-to-track association for tracks with dependent errors", *Proceedings of the IEEE CDC, Bahamas*, December 2004.
9. Y. Bar-Shalom and X. R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS Publishing, Storrs, CT, 1995.
10. Y. Bar-Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, Wiley, New York, 2001.
11. Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation, Tracking and Navigation: Theory, Algorithms and Software*, John Wiley & Sons, New York, June 2001.
12. R. W. Beard, T. W. McLain, M. A. Goodrich, and E. P. Anderson, "Coordinated target assignment and intercept for unmanned air vehicles", *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 6, pp. 911–922, December 2002.
13. R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.
14. K. Benameur, "Optimal receiver location for emitter tracking", *Proceedings of the American Control Conference*, Vol. 6, pp. 4276–4281, Arlington, VA, June 2001.
15. D. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes", *Mathematics of Operations Research*, Vol. 27, No. 4, pp. 819–840, 2002.
16. D. P. Bertsekas, *Dynamic Programming: Deterministic and Stochastic Models*, Prentice-Hall, Englewood Cliffs, NJ, 1987.
17. D. P. Bertsekas, *Linear Network Optimization: Algorithms and Codes*, MIT Press, Cambridge, MA, 1991.
18. D. P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, Nashua, NH, Vol. 1, 2, 1995.
19. Y. Bin and K. Sycara, "Learning the quality of sensor data in distributed decision fusion", *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.
20. S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*, Artech House, Dedham, MA, 1999.
21. W. D. Blair, G. A. Watson, T. Kirubarajan, and Y. Bar-Shalom, "Benchmark for radar resource allocation and tracking in the presence of ECM", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No. 3, pp. 1015–1022, October 1998.

22. A. Capponi, C. Pilotto, G. Golino, A. Farina, and L. Kaplan, "Algorithms for the selection of the active sensors in distributed tracking: comparison between Frisbee and GNS methods", *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.
23. L. R. Carley,"A noise-shaping coder topology for 15+ bit converters", *IEEE Journal of Solid-State Circuits*, Vol. SC-24, pp. 267–273, April 1989.
24. L. R. Carley and J. Kenney, "A 16-bit 4'th order noise-shaping D/A converter", *Proceedings of the IEEE 1988 Custom Integrated Circuits Conference*, pp. 21.7.1–21.7.4, Rochester, NY, May 1988.
25. K. C. Chang, C. Y. Chong, and Y. Bar-Shalom, "Joint probabilistic data association in distributed sensor networks", *IEEE Transactions on Automatic Control*, Vol. 33, No. 10, pp. 889–897, October 1986.
26. K. C. Chang, R. K. Saha, and Y. Bar-Shalom, "On optimal track-to-track fusion", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 33, No. 4, pp. 1271–1276, October 1997.
27. H. Chen and Y. Bar-Shalom, "Performance limits of track-to-track fusion versus centralized estimation: theory and application", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 39, No. 2, pp. 386–400, April 2003.
28. L. Chen, M. Cetin, and A. Willsky, "Distributed data association for multi-target tacking in sensor networks", *Proceedings of the 8th International Conference on Information Fusion*, PA, July 2005.
29. H. Chen, T. Kirubarajan, and Y. Bar-Shalom, "Centralized vs. distributed tracking algorithms for air to air scenarios", *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, Vol. 4048, April 2000.
30. C. Y. Chong and S. Mori, "Graphical models for nonlinear distributed estimation", *Proceedings of the 7th International Conference on Information Fusion*, Stockholm, Sweden, June 2004.
31. C. Y. Chong and S. Mori, "Distributed fusion and communication management for target identification", *Proceedings of the 8th International Conference on Information Fusion*, PA, July 2005.
32. C. Y. Chong, "Distributed architectures for data fusion", *Proceedings of the 1st International Conference on Information Fusion*, Las Vegas, NV, July 1998.
33. C. Y. Chong, S. Mori, W. H. Barker, and K. C. Chang, "Architectures and algorithms for track association and fusion", *IEEE Aerospace and Electronic Systems Magazine*, Vol. 15, No. 1, January 2000.
34. C. Y. Chong, S. Mori, and K. C. Chang, "Distributed multitarget multisensor tracking", in *Multitarget-Multisensor Tracking: Advanced Applications*, Chapter 8, pp. 247–295 (Y. Bar-Shalom, editor), Artech House, Norwood, MA, 1990.
35. C. Y. Chong, F. Zhao, S. Mori, and S. Kumar, "Distributed tracking in wireless ad hoc sensor networks", *Proceedings of the 6th International Conference on Information Fusion*, Queensland, Australia, July 2003.
36. F. Dambreville and J. P. LeCadre, "Detection with spatial and temporal optimization of search efforts involving multiple modes and multiple resources management", *Proceedings of the 4th Annual Conference on Information Fusion*, 2001.
37. D. Danu, A. Sinha, T. Kirubarajan, M. F. Farooq, and D. Peters "Performance evaluation of multi-platform distributed data fusion methods for multi-target tracking", *Proceedings of the IEEE Aerospace Conference*, Big Sky, MT, USA, March 2007.
38. S. Deb, M. Yeddanapudi, K. R. Pattipati, and Y. Bar-Shalom, "A generalized S-D algorithm for multisensor-multitarget state estimation", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 33, No. 2, pp. 523–538, April 1997.
39. Z. Ding and L. Hong, "Static/dynamic distributed interacting multiple model fusion algorithms for multiplatform multisensor tracking", *Optical Engineering*, Vol. 36, No. 3, pp. 708–715, 1997.
40. O. E. Drummond, "A hybrid sensor fusion algorithm architecture and tracklets", *Proceedings of the SPIE Signal and Data Processing of Small Targets*, Vol. 3163, 1997.

41. O. E. Drummond, "Tracklets and a hybrid fusion with process noise", *Proceedings of the SPIE Signal and Data Processing of Small Targets*, Vol. 3163, 1997.
42. H. Durrant-Whyte and B. Grocholsky, "Management and control in decentralised networks", *Proceedings of the International Conference on Information Fusion*, pp. 560–565, Cairns, Queensland, Australia, July 2003.
43. M. Flint, M. Polycarpou, and E. Fernandez-Gaucherand, "Cooperative control for multiple autonomous UAVs searching for targets", *Proceedings of the IEEE Conference on Decision and Control*, pp. 2823–2828, Las Vegas, Nevada, December 2002.
44. T. Furukawa, F. Bourgault, H. F. Durrant-Whyte, and G. Dissanayake, "Dynamic allocation and control of coordinated UAVs to engage multiple targets in a time-optimal manner", *Proceedings of the IEEE International Conference on Robotics & Automation*, pp. 2353–2358, New Orleans, LA, USA, April 2004.
45. C. V. Goldman and S. Zilberstein, "Goal-oriented Dec-MDPs with direct communication", University of Massachusetts Computer Science Technical Report #04–44, 2004.
46. C. V. Goldman and S. Zilberstein, "Decentralized control of cooperative systems: categorization and complexity analysis", *Journal of Artificial Intelligence Research*, Vol. 22 pp. 143–174, 2004.
47. M. L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom, "Multisensor resource deployment using posterior Cramér-Rao bounds", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 40, No. 2, April 2004.
48. M. L. Hernandez, A. D. Marrs, N. J. Gordon, S. Maskell, and C. M. Reed, "Cramér-Rao bounds for non-linear filtering with measurement origin uncertainty", *Proceedings of the 5th International Conference on Information Fusion*, Annapolis, MD, July 2002.
49. T. J. Ho and M. Farooq, "Centralized and decentralized IMM algorithms for multisensor track fusion", *Proceedings of the ONR/NPS Workshop on Estimation, Tracking and Fusion*, Monterey, CA, May 2001.
50. R. A. Howard, *Dynamic Programming and Markov Processes*. The MIT Press, Cambridge, MA, 1960.
51. L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: a survey", *Journal of Artificial Intelligence Research*, Vol. 4, pp. 237–285, 1996.
52. M. Kalandros and L. Y. Pao, "Sensor management for tracking interacting targets", *Proceedings of the ONR/NPS Workshop on Estimation, Tracking and Fusion*, Monterey, CA, May 2001.
53. T. Kirubarajan, Y. Bar-Shalom, W. D. Blair, and G. A. Watson, "IMMPDA solution to benchmark for radar resource allocation and tracking in the presence of ECM", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No. 3, pp. 1023–1036, October 1998.
54. T. Kirubarajan, Y. Bar-Shalom, and K. R. Pattipati, "Multiassignment for tracking a large number of overlapping objects", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-37, No. 1, pp. 2–21, January 2001.
55. T. Kirubarajan, Y. Bar-Shalom, K. R. Pattipati, and I. Kadar, "Ground target tracking with topography-based variable structure IMM estimator", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-36, No. 1, pp. 26–46, January 2000.
56. T. Kirubarajan, H. Wang, and Y. Bar-Shalom, "Efficient multisensor fusion using multidimensional data association", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-37, No. 2, pp. 386–400, April 2001.
57. A. Kumar, Y. Bar-Shalom, and E. Oron, "Image segmentation based on optimal layering for precision tracking", *Partitioning Data Sets, Series in Discrete Mathematics and Theoretical Computer Science*, Vol. 19, pp. 155–168, AMS, 1995.
58. B. H. Leung and S. Sutarja, "Multibit sigma – delta A/D converter incorporating a novel class of dynamic element matching techniques", IEEE Transactions on Circuits Systems II, Vol. 39, pp. 35–51, January 1992.
59. X. R. Li, "Optimal linear estimation fusion – Part VII: dynamic systems", *Proceedings of the Sixth International Conference of Information Fusion*, pp. 455–462, 2003.

60. M. L. Littman, T. L. Dean, and L. P. Kaelbling, "On the complexity of solving Markov decision problems", *Proceedings of the UAI Conference*, Montreal, QC, Canada, August 1995.
61. P. Maymounkov and D. Mazieres, "Kademlia: a peer-to-peer information system based on the XOR metric", *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS '02)*, Cambridge, MA, March 2002.
62. T. W. McLain, P. R. Chandler, and M. Pachter, "A decomposition strategy for optimal coordination of unmanned air vehicles", *Proceedings of the American Control Conference*, pp. 369–373, Chicago, IL, June 2000.
63. J. R. Moore and W. D. Blair, "Practical aspects of multisensor tracking", in *Multitarget-Multisensor Tracking: Applications and Advances III* (Y. Bar-Shalom and W. D. Blair, editors), Artech House, Norwood, MA, 2000.
64. R. Niu, K. Varshney, M. Moore, and D. Klamer, "Decision fusion in a wireless sensor network with a large number of sensors", *Proceedings of the 7th International Conference on Information Fusion*, Stockholm, Sweden, June 2004.
65. S. R. Norsworthy, R. Schreier, and G. C. Temes, *Delta-Sigma Data Converters, Theory, Design, and Simulation*, IEEE Press, New York, 1997.
66. P. Ogren, E. Fiorelli, and N. E. Leonard, "Cooperative control of mobile sensor networks: adaptive gradient climbing in a distributed environment", *IEEE Transactions on Automatic Control*, Vol. 49, No.8, pp. 1292–1302, August 2004.
67. C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov chain decision processes", *Mathematics of Operations Research*, Vol. 12, No. 3, 1987.
68. D. Perugini, D. Lambert, L. Sterling, and A. Pearce, "Distributed information fusion agents", *Proceedings of the 6th International Conference on Information Fusion*, Queensland, Australia, July 2003.
69. A. B. Poore, and A. J. Robertson III, "A new class of Lagrangian relaxation based algorithms for a class of multidimensional assignment problems", *Computational Optimization and Applications*, Vol. 8, No. 2, pp. 129–150, September 1997.
70. M. L. Puterman, *Markov Decision Processes – Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Inc., New York, 1994.
71. Z. Rabinovich, C. V. Goldman, and J. S. Rosenschein, "The complexity of multiagent systems: the price of silence", *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pp. 1102–1103, Melbourne, Australia, 2003.
72. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable contentaddressable network", *Proceedings of the ACM SIGCOMM 2001 Conference*, San Diego, CA, August 2001.
73. B. Ristic, S. Zollo, and S. Arulampalam, "Performance bounds for maneuvering target tracking using asynchronous multi-platform angle-only measurements", *Proceedings of the 4th International Conference on Information Fusion*, Montreal, Quebec, August 2001.
74. A. Rogers, R. K. Dash, N. R. Jennings, S. Reece, and S. Roberts, "Computational mechanism design for information fusion within sensor networks", *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.
75. R. L. Rothrock and O. E. Drummond, "Performance metrics for multiple-sensor, multiple-target tracking," *Signal and Data Processing of Small Targets 2000, Proceedings SPIE*, Vol. 4048, pp. 521–531, July 2000.
76. Y. Sakina, "Multi-bit $\Sigma\Delta$ analog-to-digital converters with nonlinearity correction using dynamic barrel shifting", Electronics Research Laboratory, College of Engineering, University of California, Berkley CA, Memorandum No. UCB/ ERL M93/63, 1993.
77. L. Sevgi, A. Ponsford, and H. C. Chan, "An integrated maritime surveillance system based on high-frequency surface-wave radars", *IEEE Antennas and Propagation Magazine*, Vol. 43, No. 4, pp. 28–43, August 2001.
78. A. Sinha, H. Chen, D. G. Danu, T. Kirubarajan, and M. Farooq, "Estimation and decision fusion: a survey", *Proceedings of the IEEE International Conference on Engineering of Intelligent Systems*, Islamabad, Pakistan, April 2006.

79. A. Sinha, T. Kirubarajan, and Y. Bar-Shalom, "Optimal cooperative placement of UAVs for ground target tracking with Doppler radar", *Proceedings of the SPIE Signal Processing, Sensor Fusion, and Target Recognition*, Orlando, FL USA, April 2004.
80. A. Sinha, T. Kirubarajan, and Y. Bar-Shalom, "Autonomous ground target tracking by multiple cooperative UAVs", *Proceedings of the IEEE Aerospace Conference*, Big Sky, MT USA, March 2005.
81. A. Sinha, T. Kirubarajan, and M. Farooq, "Performance test of a multi-platform distributed data fusion system", Technical Report 2006/51, Estimation, Tracking and Fusion Laboratory (ETFLab), Electrical and Computer Engineering Department, McMaster University, Hamilton, Ontario, Canada, June, 2006.
82. A. N. Steinberg, "Stimulative intelligence", *Proceedings of the National Symposium on Sensor and Data Fusion*, McLean, VA, June 2006.
83. A. N. Steinberg, "Open networks: generalized multi-sensor characterization", *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.
84. I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup service for internet applications", *Proceedings of the ACM SIGCOMM 2001 Conference*, San Diego, CA, August 2001.
85. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.
86. R. Tharmarasa, T. Kirubarajan, and M. L. Hernandez, "Large-scale optimal sensor array management for target tracking", *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, Vol. 37, No. 5, pp. 803–814, September 2007.
87. R. Tharmarasa, T. Kirubarajan, M. L. Hernandez, and A. Sinha, "PCRLB based multisensor array management for multitarget tracking", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 43, No. 2, pp. 539–555, April 2007.
88. P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramér-Rao bounds for discrete time nonlinear filtering", *IEEE Transactions on Signal Processing*, Vol. 46, No. 5, pp. 1386–1396, May 1998.
89. R. J. van de Plassche, "Precision current-source arrangement", U.S. Patent 3 982 172, September 21, 1976.
90. R. J. van de Plassche, "Dynamic element matching for high-accuracy monolithic D/A converters", *IEEE Journal of Solid-State Circuits*, Vol. SC-11, pp. 795–800, December 1976.
91. H. Van Trees, *Detection, Estimation and Modulation Theory*, Vol. I, Wiley, New York, 1968.
92. H. Wang, T. Kirubarajan, and Y. Bar-Shalom, "Large scale air traffic surveillance using IMM estimators with assignment", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 35, No. 1, pp. 255–266, January 1999.
93. G. A. Watson, W. D. Blair, and T. R. Rice, "Enhanced electronically scanned array resource management through multisensor integration", *Proceedings of the SPIE Conference on Signal Processing of Small Targets*, Orlando, FL, 3163, pp. 329–340, 1997.
94. G. A. Watson and G. H. McCabe, "Benchmark problem with a multisensor framework for radar resource allocation and tracking of highly maneuvering targets, closely-spaced targets, and targets in the presence of sea-surface-induced multipath", Technical Report NSWCDD, Dahlgren, VA, 1999.
95. R. J. Williams and L. C. Baird III, "Tight performance bounds on greedy policies based on imperfect value functions", Tech. Rep. NU-CCS-93–14, Northeastern University, College of Computer Science, Boston, MA, 1993.
96. E. J. Wright and K. B. Laskey, "Credibility models for multi-source fusion", *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.
97. S. W. Yeom, T. Kirubarajan, and Y. Bar-Shalom, Y., "Track segment association, finestep IMM and initialization with Doppler for improved track performance", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 40, No. 1, pp. 293–309, January 2004.
98. B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz, "Tapestry: a resilient global-scale overlay for service deployment", *IEEE Journal on Selected Areas in Communications*, Vol. 22, No. 1, pp. 41–53, January 2004.

Part III
Practice

Deployment Techniques for Sensor Networks

Jan Beutel, Kay Römer, Matthias Ringwald, and Matthias Woehrle

Abstract The prominent visions of wireless sensor networks that appeared about a decade ago have spurred enormous efforts in research and development of this new class of wireless networked embedded systems. Despite the significant effort made, successful deployments and real-world applications of sensor networks are still scarce, labor-intensive and often cumbersome to achieve. In this article, we survey prominent examples of sensor network deployments, in particular for environmental monitoring applications, their interaction with the real world and classify a number of potential causes for errors and common pitfalls. In the second half of this work, we present methods and tools to be used to detect failures, identify and understand root causes. These instrumentation techniques and analysis tools are specifically designed or adapted for the analysis of distributed networked embedded systems at the level of components, sensor nodes, and networks of nodes.

1 Introduction

Sensor networks offer the ability to monitor real-world phenomena in detail and at large scale by embedding wireless network of sensor nodes into the environment. Here, *deployment* is concerned with setting up an operational sensor network in a real-world environment. In many cases, deployment is a labor-intensive and cumbersome task as environmental influences trigger bugs or degrade performance in a way that has not been observed during pre-deployment testing in a lab. The reason for this is that the real world has a strong influence on the function of a sensor network by controlling the output of sensors, by influencing the existence and quality of wireless communication links, and by putting physical strain on sensor nodes. These influences can only be modeled to a very limited extent in simulators and lab testbeds.

J. Beutel (✉)
Institute for Computer Engineering and Networks Lab, ETH Zurich, 8092 Zurich, Switzerland
e-mail: beutel@tik.ee.ethz.ch

The work presented in this paper was supported by the Swiss National Science Foundation under grant number 5005-67322 (NCCR-MICS), and by the European Commission under contract number FP7-2007-2-224053 (CONET).

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_9,

Information on the typical problems encountered during deployment is rare. We can only speculate on the reason for this. On the one hand, a paper which only describes what happened during a deployment seldom constitutes novel research and might be hard to get published. On the other hand, people might tend to hide or ignore problems which are not directly related to their field of research. Additionally it is often hard to discriminate desired and non-desired functional effects at the different layers or levels of detail.

In this chapter we review prominent wireless sensor network installations and problems encountered during their deployment. The insight into a sufficiently large number of deployments allows to identify common deployment problems, especially in the light of system architecture and components used, in order to gain a broader and deeper understanding of the systems and their peculiarities. In a second part we survey approaches and methods to overcome deployment problems at the level of components, sensor nodes, and networks of nodes. A special focus is on techniques for instrumentation and analysis of large distributed networked embedded systems at run-time.

2 Wireless Sensor Network Deployments

To understand the problems encountered during deployment, 14 different projects are reviewed with different goals, requirements and success in deploying the sensor network. The key figures for the projects surveyed are given in Table 1. Main deployment characteristics are included such as the network size and the duration of a deployment. The column *yield* denotes the amount of data reported by the sensor network with respect to the expected optimum, e.g., based on the sample rate.

Table 1 Characteristics of selected deployments

Deployment	Year	#nodes	Hardware	Duration	Yield	Multi-hop
GDI I	2002	43	Mica2Dot	123 days	16%	No
GDI II – patch A	2003	49	Mica2Dot	115 days	70%	No
GDI II – patch B	2003	98	Mica2Dot	115 days	28%	Yes
Line in the sand	2003	90	Mica2	115 days	n/a	Yes
Oceanography	2004	6	Custom HW	14 days	Not reported	No
GlacsWeb	2004	8	Custom HW	365 days	Not reported	No
SHM	2004	10	Mica2	2 days	Up to 50%	Yes
Pipenet	2004/2005	3	Intel Mote	425–553 days	31% − 63%	No
Redwoods	2005	33	Mica2Dot	44 days	49%	Yes
Potatoes	2005	97	TNode	21 days	2%	Yes
Volcano	2005	16	TMote Sky	19 days	68%	Yes
Soil ecology	2005	10	MicaZ	42 days	Not reported	No
Luster	2006	10	MicaZ	42 days	Not reported	No
Sensorscope	2006–2008	6–97	TinyNode	4–180 days	Not reported	Yes

In the above projects, a variety of sensor node hardware has been used as summarized in Table 2. The majority of the early projects used the Mica2 mote [31] designed at the University of California at Berkeley and produced by Crossbow

Table 2 Sensor node characteristics

	Mica2(Dot)	T-Node	MicaZ	Tmote Sky
Microcontroller	ATmega128L	ATmega128L	ATmega128L	MSP 430
Architecture	8 bit	8 bit	8 bit	16 bit
Clock	7.328 MHz (4 MHz)	7.328 MHz	7.328 MHz	8 MHz
Program memory	128 kB	128 kB	128 kB	48 kB
Data memory	4 kB	4 kB	4 kB	10 kB
Storage memory	512 kB	512 kB	512 kB	1024 kB
Radio	Chipcon CC1000	Chipcon CC1000	Chipcon CC2420	Chipcon CC2420
Frequency	433 / 915 MHz	868 MHz	2.4 GHz	2.4 GHz
Data rate	19.2 kbps	19.2 kbps	250 kbps	250 kbps

	TinyNode	Intel Mote	Oceanography	GlacsWeb
Microcontroller	MSP 430	ARM7TDMI	PIC 18F452	PIC 16LF878
Architecture	16 bit	32-bit	8 bit	8 bit
Clock	8 MHz	12 MHz	<40 MHz	<20 MHz
Program memory	48 kB	64kB	32 kB	16 kB
Data memory	10kB	(*)	1536 B	368 B
Storage memory	512kB	512kB	0.25 kB	64 kB
Radio	Xemics XE1205	Zeevo BT Radio	not specified	Xemics
Frequency	868 MHz	2.4 GHz	173 MHz	433 MHz
Data rate	152.3 kbps	1 Mbps	10 kbps	9600 bps

Data for Mica2Dot in parentheses.

Technology Inc. or a variant of it (Mica2Dot, T-Node [19]). Its main components are an Atmel ATmega128L 8-bit microcontroller and a Chipcon CC1000 radio module for the 433/868/915 MHz ISM bands. More recent deployments often use the TI MSP 430 microcontroller due to its energy efficiency and more advanced radio modules such as the Chipcon CC2420 (implementing the 802.15.4 standard) on the Tmote Sky [30] or the Xemics XE1205 on the TinyNode [10] sensor nodes.

2.1 Great Duck Island

One of the earliest deployments of a larger sensor network was carried out in the summer of 2002 on Great Duck Island [31], located in the gulf of Maine, USA. The island is home to approximately 5000 pairs of Leach's Storm Petrels that nest in separate patches within three different habitat types. Seabird researchers are interested in questions regarding the usage pattern of nesting burrows with respect to the microclimate. As observation by humans would be both too costly and might disturb the birds, a sensor network of 43 nodes was deployed for 4 months just before the breeding season. The nodes had sensors for light, temperature, humidity, pressure, and infrared radiation and have been deployed in a single hop network. Each sensor node samples its sensors every 70 s and sends its readings to a solar-powered gateway. The gateway forwards the data to a central base station with a database and a two-way satellite connection to the Internet. During the 123 days of the experiment, 1.1 million readings have been recorded, which is about one sixth of the theoretical 6.6 million readings generated over this time.

In a book chapter [31], the authors analyze the network's behavior in detail. The most loss of data was caused by hardware-related issues. Several nodes stopped working due to water entering the sensor node casing. As all sensors were read out by a single analog-to-digital converter, a hardware failure of one of the sensors caused false readings of other sensors. Due to the transparent casing of the sensor nodes, direct sun light could heat the whole sensor node and thus lead to high temperature readings for nodes which are deployed above ground. Over time, various sensors report false readings such as humidity over 150% or below 0%, or too low or unreasonably high temperature. The temperature sensor of about half the nodes failed at the same time as the humidity sensor suggesting water inside the packaging. Although it did not directly cause packet loss as the gateway was always listening, several nodes did show a phase skew with respect to their 70 s sending interval. A crash of the database running on the base station resulted in the complete loss of data for two weeks.

After lessons learned from the first deployment, a second deployment was conducted in 2003 [41]. This time, two separate networks, a single-hop network of 49 nodes similar to the one in the first deployment and a multi-hop network with 98 nodes were deployed. The multi-hop network used the routing algorithm developed by Woo [52]. Again, the project suffered from several outages of the base station – this time caused by harsh weather. In the multi-hop network, early battery depletion was caused by overhearing in combination with low-power listening. In the pre-deployment calculation, the group did not account for an increased overhearing in the multi-hop network although it could have been predicted.

2.2 A Line in the Sand

An early project focussing on intrusion detection, classification and tracking of targets is "A line in the sand" [2]. The system consists of densely deployed sensor nodes with integrated binary detection sensors that collaboratively detect multiple objects.

The system implementation was evaluated by repeated deployments of ninety sensor nodes on three different sites in spring, summer, and fall of 2003 and smaller, focussed test deployments. The authors describe key problems encountered during the project. One of the main issues is the unreliability of the network. This is exacerbated by using a dense network of sensor nodes, which leads to network saturation and congestion and thus a considerable amount of collisions. Unexpected system interaction at scale diminished services for reliable communication. Additional unanticipated problems occurred due to hardware failures (e.g., debonding or desensitizing of sensors over time) and environmental conditions, exhausted batteries as well as problems due to incorrectly downloaded software. Unanticipated problems caused protocols to fail. The authors identify many faults that can be addressed by better packaging, hardware optimization and allowing for more redundancy, but indicate that many faults still require proper support for identification and resolution.

2.3 Oceanography

A small sensor network of 6 nodes was deployed in 2004 on a sandbank off the coast of Great Yarmouth, Great Britain [43] to study sedimentation and wave processes. A node did consist of a radio buoy for communication above the sea and a sensor box on the seabed connected by a wired serial connection. The sensor box had sensors for temperature, water pressure (which allows to derive wave height), water turbidity, and salinity. The authors reported problems with the sensor box due to last-minute software changes which led to cutting and re-fixing of the cable between buoy and sensor, and later, to the failure of one of the sensors caused by water leakage.

2.4 GlacsWeb

The GlacsWeb project [28] deployed a single-hop sensor network of 8 glacier probes in Norway. The aim of this system is to understand glacier dynamics in response to climate change. Each probe samples every four hours the following sensors: temperature, strain (due to stress of the ice), pressure (if immersed in water), orientation, resistivity (to detect whether the probe would be in sediment till, water or ice), and battery voltage. The probes were installed in up to 70 m deep holes located around a central hole which did hold the receiver of the base station.

In this deployment, initially, the base station only received data from seven probes and during the course of the experiment, communication with four probes failed over time. In the end, three probes were able to report their sensor readings. The base station experienced an outage. The authors speculate that the other probes might have failed for three reasons: Firstly, nodes might have been moved out of transmission range because of sub-glacial movement. Secondly, the node casing might have broken due to stress by the moving ice. And thirdly, clock drift and sleeping policy might have led to unsynchronized nodes which hinders communication.

2.5 Structural Health Monitoring

To assess the structural health of buildings, the Wisden [29] data acquisition system was conceived. Each node measures seismic motion by means of a three-axis accelerometer and forwards its data to a central station over a multi-hop network. The data samples are time stamped and aggregated in the network to compensate for the limited bandwidth. In the case of a seismic event, the complete data is buffered on a node for reliable but delayed end-to-end data transmission.

The authors report a software defect in their system, where an 8-bit counter was used for the number of locally buffered packets and an overrun would cause packets

to not be delivered at all. Also, the accelerometer readings showed increased noise when the battery voltage did fall below a certain threshold.

2.6 Pipenet

Pipenet [39] is a sensor network to monitor pipeline infrastructures allowing for increased spatial and temporal resolution of operational data. It collects hydraulic and acoustic/vibrational data at high sampling rates and analyzes the data to identify and locate leaks in a pipeline. Pipenet uses a tiered architecture with the sensor network tier consisting of a cluster of battery-operated Intel Motes equipped with a data acquisition board. Sensor nodes are directly connected to the pipe, sense and process the collected information and directly send it via Bluetooth to the upper tier. First results of the initial trial from December 2004 to July 2005 are presented. One of the main problems encountered was battery exhaustion leading to long-term missing data. Short-term missing data is attributed to a watch-dog rebooting the gateway at midnight each day, leading to a loss of all messages in transit. Other sources for packet loss are environmental conditions such as rainfall and snow, where especially snow had a significant effect on packet loss. Hardware problems due to bad antennas and insufficient waterproofness attributed to additional problems in one of the clusters.

2.7 Redwood Trees

To monitor the microclimate of a 70-meter tall redwood tree, 33 sensor nodes have been deployed along a redwood tree roughly every 2 m in height for 44 days in 2005 [46]. Each node measured and reported every 5 minutes air temperature, relative humidity, and solar radiation. The overall yield of this deployment has been 49%. In addition to the Great Duck Island hardware and software, the "Tiny Application Sensor Kit" (TASK) was used on the multi-hop network. The TinyDB [25] component included in TASK provides an SQL-like database interface to specify continuous queries over sensor data. In addition to forwarding the data over the network, each sensor node was instructed to record all sensor readings into an internal 512 kB flash chip.

Some nodes recorded abnormally high temperature readings above 40°C when other nodes reported temperatures between 5 and 25°C. This allowed to single out nodes with incorrect readings. Wrong sensor readings have been highly correlated to low battery voltage similar to the report for the Structural Health Monitoring. This should have not been surprising as the used sensor nodes, Mica2Dot motes, did not employ a voltage converter and the battery voltage fell below the threshold for reliable operation over time. Also in this project, two weeks of data were lost due to a gateway outage. The data stored in the internal flash chip was complete but did not cover the whole deployment. Although it was estimated that it would suffice,

initial tests, calibration, and a longer deployment than initially envisioned led to a full storage after about four weeks. In the end it turned out that the overall data yield of 49% was only possible by manually collecting all nodes and extracting the data from the flash memory on each node.

2.8 LOFAR-agro

A detailed report on deployment problems was aptly called "LOFAR-agro – Murphy Loves Potatoes" [19]. The LOFAR-agro project is aimed at precision agriculture. In summer 2005, after two field trials, 110 sensor nodes with sensors for temperature and relative humidity were deployed in a potato field just after potatoes have been planted. The field trials and the final deployment suffered from a long list of problems.

Similar to the oceanography project, an accidental commit to the source code revision control system led to a partially working MAC protocol being installed on the sensor nodes just before the second field trial. Later, update code stored in the nodes' external flash memory caused a continuous network code distribution which led to high network congestion, a low data rate and thus the depletion of all nodes' batteries within 4 days. The routing and the MAC component used different fixed size neighbor tables. In the dense deployment, where a node might have up to 30 neighbors, not all neighborhood information could be stored, which caused two types of faulty behavior. Firstly, the routing component of most nodes did not send packets to the gateway although the link would have been optimal. Secondly, as both components used different neighbor tables, packets got dropped by the MAC-layer when the next hop destination was not in its neighbor table. To allow nodes to recover from software crashes, a watchdog timer was used. Either due to actual program crashes or due to a malfunction of the watchdog handling, most nodes rebooted every 2–6 h. This did not only cause data loss for the affected node but also led to network instability as the entries for rebooting nodes are removed by their neighbors from their neighborhood tables. As in other projects, the LOFAR-agro project suffered from gateway outages. In this case, a miscalculation of the power requirements for the solar-powered gateway caused a regular outage in the morning when the backup battery was depleted before the sun rises and the solar cells provided enough power again. The sensor nodes were programmed to also store their readings in the external flash memory, but due to a small bug, even this fallback failed and no data was recovered after the deployment. In total, the 97 nodes which ran for 3 weeks did deliver 2% of the measured data.

2.9 Volcanoes

In August 2005, a sensor network of 16 sensor nodes has been deployed on the volcano Reventador in Ecuador [47]. Each node samples seismic and acoustic data

at 100 Hz. If a node detects a local seismic event, it notifies a base stations. If 30% of the nodes report an event in parallel, the complete data set of the last minute is fetched from all nodes in a reliable manner. Instead of immediately reporting all data, which would lead to massive network congestion and packet collisions with current low-power MAC protocols, the nodes are polled by the base station sequentially.

The first problem encountered was a software defect in the clock component which would occasionally report a bogus time. This led to a failure of the time synchronization mechanism. The team tried to reboot the network but this trigged another bug, which led to nodes continuously rebooting. After manual reprogramming of the nodes, the network was working quite reliably. A median *event yield* of 68% was reported, which means that for detected events 68% of the data was received. As with other deployments, data was lost due to power outage at the base station. During the deployment, only a single node stopped reporting data and this was later confirmed to be due to a broken antenna.

2.10 Soil Ecology

To monitor the soil ecology in an urban forest environment, ten sensor nodes have been deployed near John Hopkins University in the autumn of 2005. The nodes have been equipped with manually calibrated temperature and soil moisture sensors and packed into a plastic waterproof casing. The sensor application was designed to store all sensor readings in the local flash memory which had to be read out every two weeks to guarantee 100% sensor data yield in combination with a reliable data transfer protocol.

However, due to an unexpected hardware behavior, a write to the flash memory could fail and an affected node would then stop recording data. Further parts of the data have been lost due to human errors while downloading the data to a laptop computer. Similar to previous deployments, the software on the nodes had to be updated and for this the waterproof cases had to be re-opened several times which led to water leakage in some cases.

2.11 Luster

A recent project for environmental monitoring is Luster [37], a system to monitor environmental phenomena such as temperature, humidity, or CO_2. LUSTER is an environmental sensor network specifically designed for high reliability. Its main goal is to measure sunlight in thickets in order to study the relation between light conditions and the phenomenon of shrub thickets overwhelming grasslands. Luster is a multi-layer, single-hop architecture replicated into multiple clusters that cover the entire deployment area: The lowest layer is a cluster of sensor nodes collecting and transmitting data at a configurable rate to a clusterhead gateway. The gateway

nodes (Stargate microservers) form a delay tolerant network layer. For redundancy purposes an additional layer is introduced, the reliable distributed storage layer. This layer is made up of dedicated storage nodes overhearing data reported by sensor nodes at the lowest layer and passively storing it on SD cards, for physically retrieval at a later point in time. Overlapping sensor coverage increases reliability of the overall system.

The system was deployed at Hog Island. While two problems were caused by hardware failures, i.e., bad contacts to the sensor, and a non-responding storage node, the problems could be identified in the field using their inspection tool SeeDTV [22].

2.12 SensorScope

Sensorscope [4, 3] is a system for environmental monitoring with many deployments across Switzerland. Instead of few accurate and expensive sensors, many densely deployed inexpensive sensing stations are used to create accurate environment models. Instead of accuracy of individual sensors, the system design strives for generating models by high spatial density of inexpensive sensing stations. Each station is powered by a solar cell, which is mounted onto a flagstaff alongside the sensors and a TinyNode. Different environmental parameters are captured and gathered for later analysis. Most deployments are in the range of days to a couple of months. The environments are typically harsh, especially in the high mountain terrain deployments, e.g., on the Grand St. Bernard pass. Similar to the report of the LOFAR-agro project, the authors provide a detailed overview of issues and lessons learned in a comprehensive guide [3]. They again stress the importance of adequate packaging of the sensor nodes, and especially the connectors. With respect to these particularly harsh environments, substantial temperature variation showed considerable impact on the clock drift. However, while the clock drift typically affects the time synchronization of the network, in this case, it induced a loss of synchronization of the serial interface between the sink node and the GPRS modem. In an indoor test deployment, the authors report on packet loss due to interference where the interfering source could not be determined. Finally, a change of the query interval of the wind speed sensor, when moving from the lab to the field, caused counter overruns, which rendered a lot of sensor readings useless.

3 Deployment Problems

Based on the described problems typically found during deployments as surveyed in the previous section, a classification of problems is presented. Here, a “problem” is defined as a behavior of a set of nodes that is not compliant with a (informal) specification.

We classify problems according to the number of nodes involved into four classes: *node problems* that involve only a single node, *link problems* that involve

two neighboring nodes and the wireless link between them, *path problems* that involve three or more nodes and a multi-hop path formed by them, and *global problems* that are properties of the network as a whole.

3.1 Node Problems

A common node problem is *node death* due to energy depletion either caused by "normal" battery discharge [2, 39], short circuits or excessive leakage due to inadequate or broken packaging [41].

Low batteries often do not result in a fail-stop behavior of a sensor node. Rather, nodes show random behavior below a certain low battery level. In the Redwood Trees deployment [46], for example, *wrong sensor readings* have been observed at low battery voltage. Even worse, a slowly degrading node can also impact the performance of other nodes in a network ensemble, possibly still having enough energy.

An increased amount of network traffic, compared to initial calculations, led to an early battery depletion due to unexpected overhearing (e.g., Great Duck Island deployment [40], Sect. 2.1) or repeated network floods (e.g., LOFAR-agro deployment [19], Sect. 2.8). In the Great Duck Island deployment [41], a low-resistance path between the power supply terminals was created by water permeating a capacitive humidity sensor, resulting in early battery depletion and abnormally small or large sensor readings. Poor packaging [2, 3], bad contacts [37] to sensors or deteriorating sensors [2] are typical problems encountered for sensor nodes, especially in harsh environmental conditions.

Software bugs often result in *node reboots*, for example, due to failure to restart the watchdog timer of the micro controller (e.g., LOFAR-agro deployment [19]). Also observed have been software bugs resulting in hanging or killed threads, such that only part of the sensor node software continued to operate. Overflows in counters may also deviate the sensor readings by spoiling a reference interval [3]. In [2] problems are also attributed to incorrectly downloaded software.

Sink nodes act as gateways between a sensor network and the Internet. In many applications they store and forward data collected by the sensor network to a background infrastructure. Hence, problems affecting sink nodes or the gateway must be promptly detected to limit the impact of data loss (e.g., GlacsWeb deployment [28], Great Duck Island deployment [40], Redwood Trees deployment [46]). This can also manifest in temporary errors as reported for the serial link between the sink and its secondary interface [3] or periodic problems due to watchdog reboots [39].

3.2 Link Problems

Field experiments (e.g., [52, 14]) demonstrated a very high variability of link quality both across time and space resulting in temporary link failures and variable amounts of *message loss*. Interference in office buildings can considerably affect the packet loss; the source often cannot be determined [3].

Network congestion due to *traffic bursts* is another source of message loss. In the Great Duck Island deployment [40], for example, a median message loss of 30% is reported for a single-hop network. Excessive levels of traffic bursts have been caused by accidental synchronization of transmissions by multiple senders, for example, due to inappropriate design of the MAC layer [33] or by repeated network floods as in the LOFAR-agro deployment [19]. If message loss is compensated for by retransmissions, a *high latency* may be observed until a message eventually arrives at the destination. Congestion is especially critical in dense networks, e.g., when many nodes detect an event simultaneously and subsequently compete with the neighbors for medium access to send off an event notification [2].

Most protocols require each node in the sensor network to discover and maintain a set of network neighbors (often implemented by broadcasting HELLO messages containing the sender address). A node with *no neighbors* presents a problem as it is isolated and cannot communicate. Also, *neighbor oscillation* is problematic [33], where a node experiences frequent changes of its set of neighbors.

A common issue in wireless communication are *asymmetric links*, where communication between a pair of nodes is only possible in one direction. In a field experiment [14] between 5 and 15% of all links have been observed to be asymmetric, with lower transmission power and longer node distance resulting in more asymmetric links. If not properly considered, asymmetric links may result in fake neighbors (received HELLO from a neighbor but cannot send any data to it) and broken data communication (can send data to neighbor, but cannot receive acknowledgments).

Another issue is the physical length of a link. Even though if two nodes are very close together, they may not be able to establish a link (*missing short links*). On the other hand, two nodes that are very far away from each other (well beyond the nominal communication range of a node), may be able to communicate (*unexpected long links*). Experiments in [14] show that at low transmit power about 1% of all links are twice as long as the nominal communication range. These link characteristics make node placement highly non-trivial as the signal propagation characteristics of the real-world setting have to be considered [7] to obtain a well-connected network.

Most sensor network MAC protocols achieve energy efficiency by scheduling communication times and turning the radio module off in-between. Clock drift or repeated failures to re-synchronize the communication phase may result in failures to deliver data as nodes are not ready to receive when others are sending. In [26], for example, excessive *phase skew* has been observed (about two orders of magnitude larger than the drift of the oscillator).

3.3 Path Problems

Many sensor network applications rely on the ability to relay information across multiple nodes along a multi-hop path. In particular, most sensor applications include one or more sink nodes that disseminate queries or other tasking information

to sensor nodes and sensor nodes deliver results back to the sink. Here, it is important that a path exists from a sink to each sensor node, and from each sensor node to a sink. Note that information may be changed as it is traversing such a path, for example due to data aggregation. Two common problems in such applications are hence *bad path to sink* and *bad path to node*. In [19], for example, *selfish nodes* have been observed that did not forward received traffic, but succeeded in sending locally generated messages.

Since a path consists of a sequence of links, the former inherits many of the possible problems from the latter such as *asymmetric paths, high latency, path oscillations*, and *high message loss*. In the Great Duck Island deployment [40], for example, a total message loss of about 58% was observed across a multi-hop network.

Finally, *routing loops* are a common problem, since frequent node and communication failures can easily lead to inconsistent paths if the software is not properly prepared to deal with these cases. Directed Diffusion [17], for example, uses a data cache to suppress previously open data packets to prevent routing loops. If a node reboots, the data cache is deleted and loops may be created [38].

3.4 Global Problems

In addition to the above problems which can be attributed to a certain subset of nodes, there are also some problems which are global properties of a network. Several of these are failures to meet certain application-defined quality-of-service properties. These include *low data yield, high reporting latency*, and *short network lifetime* [45].

Low data yield means that the network delivers an insufficient amount of information (e.g., incomplete sensor time series). In the Redwood Trees deployment [46], for example, a total data yield of only about 20–30% is reported. This problem is related to message loss as discussed above, but may be caused by other problems such as a node crashing before buffered information could be forwarded [39], buffer overflows, etc. One specific reason for a low data yield is a *partitioned network*, where a set of nodes is not connected to the sink.

Reporting latency refers to the amount of time that elapses between the occurrence of a physical event and that event being reported by the sensor network to the observer. This is obviously related to the path latency, but as a report often involves the output of many sensor nodes, the reporting latency results from a complex interaction within a large portion of the network.

The lifetime of a sensor network typically ends when the network fails to cover a given physical space sufficiently with live nodes that are able to report observations. The network lifetime is obviously related to the lifetime of individual nodes, but includes also other aspects. For example, the death of certain nodes may partition the network such that even though sensing coverage is still provided, nodes can no longer report data to the observer.

3.5 Discussion

Orthogonal to the classification in the previous section, the deployment problems in the surveyed literature fall into two categories: implementation and design defects. A majority of the problems reported have been caused by problems and defects in the hardware and software implementation, and can be fixed after they have been detected, analyzed, and understood. Here, dedicated inspection tools allow to find the defects quicker.

The two most underestimated problems in the surveyed sensor network deployments have been the water-proof packaging of the sensor nodes required for an outside deployment and the provision of a reliable base station which records sensor data and has to run for months and years. This suggests that sensor nodes should be sold together with appropriate packaging. However, due to the variety of sensors used for different applications, a common casing is often not practicable or possible. The provision of a reliable base station is not specific to wireless sensor networks and mostly depends on a reliable power supply and software implementation.

4 Understanding the System

To detect the problems discussed in the previous section and to identify their causes, one needs to analyze the state of the system as it changes over time. Here, the term *system* does not only refer to the state of the sensor network, but also to the state of its enclosing environment as the latter is closely coupled to the state of the sensor network. The state of the sensor network itself is comprised of the state of the individual nodes plus the states of the communication channels between the nodes. The state of a sensor node can be further refined into the hardware state and the state of the software executed by the microcontroller. Isolating the contribution of a single system component from the overall context is a complicated task and care should be taken to actually design a sensor network in a way that this task is facilitated.

As many sensor networks are real-time systems or have real-time aspects (e.g., sensor readout at regular intervals), it is not only important to understand the progression of the system state over time, but also its exact timing, i.e., *when* a certain state is assumed with respect to real time. Especially in the context of a larger distributed system, in the case of a sensor network even exacerbated by stringent resource constraints, this is a feat requiring novel tools and methods for analysis.

4.1 Hardware

The sensor node hardware typically consists of three main subsystems: the microcontroller, a low-power radio and a sensing subsystem, often implemented as an auxiliary sensor board. Components are connected through buses, e.g., UART or SPI, and interrupt lines. Each of these components executes concurrently with synchronization between contexts initiated through interrupts on the microcontroller.

In the following, we discuss the individual component state of these three main building blocks and discuss how physical characteristics such as power consumption allow for inferring the overall system state.

Component State. System state is typically referred to as the state of the microcontroller and is inferred using an instrumentation technique (cf. Sect. 5). However, the microcontroller is not the sole functional component attributing to the system state; the node hardware state machine is the cross-product of each of the individual component state machines. As an example consider the process of sending a packet: the microcontroller prepares the packet and sends the packet to the radio buffer, e.g., via the SPI bus. While the radio schedules the transmission based on its internal state machine, the microcontroller can either perform further work or go to a low-power sleep state. In this interval, the microcontroller has no knowledge of the ongoing state changes of the radio driver. Finally, when a packet is transmitted, the radio notifies the processor via an interrupt.

Power Consumption. Each component of a sensor node can be characterized by its power consumption. Power consumption is dependent chiefly on current state but also features dynamic, non-linear effects due to capacitances and inductances. Sensor node components have typically very distinctive power consumption characteristics for their individual state. As an example, Table 3 shows power consumption for different hardware component states of the microcontroller and the radio. A detailed overview of current draw of different hardware states is provided in [13].

Table 3 Current consumption for different component states of a Tmote Sky node. Max values derived from datasheet

Microcontroller	Radio	Current (mA)
On	RX	23
On	TX	21
On	Off	2.4
Idle	Off	1.2
Standby	Off	0.021

In the context of a detailed powertrace is is possible to infer system behavior and system state in a most expressive way. Shown in Fig. 1 is an exemplary powertrace of an enhanced synchronized low-power listening scheme implemented in TinyOS-2.x for the Tmote Sky platform. By timing and coordinating packet transmissions

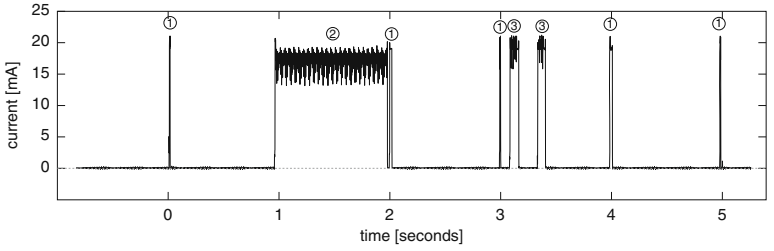


Fig. 1 A powertrace yields detailed insight into system behavior and anomalies

between periodic listening/acknowledgement periods (1) a data packet must not be sent repetitively (2) but only once when the receiving side is ready listening (3) [21].

4.2 Software

The state of the software executed by a microcontroller is defined by memory contents and by the contents of the registers of the microcontroller, in particular the program counter and the stack pointer. As many microcontrollers feature a Harvard architecture with separate program memory (flash) and data memory (RAM), the focus is often on the data memory as the contents of the program memory typically do not change over time. Many sensor nodes offer additional flash memories. While earlier systems used these primarily as data sinks for logging (and hence flash memory contents do not affect program behavior directly), there is a trend to use these memories in a more interactive way, for example, for file systems or virtual memories. In this case, flash memory contents may directly affect program behavior and hence are as important as RAM in understanding the behavior of the system.

Instead of working with a low-level view of the software state directly, one is often more interested in a higher-level view, e.g., the values of certain program variables (instead of raw memory contents) or a function call trace (instead of the content of the program counter). These high-level views also depend on the functionality of the operating system used. In many operating systems such as TinyOS, there is no strict separation between a kernel and the application. Here, the kernel is simply a library of functions that is linked to the application code. As there is no means of isolation and protection, bugs in the application code can modify kernel state and vice versa, such that the complete software state has to be taken into account to understand the behavior of the system.

Other systems offer a more strict separation and sometimes also isolation mechanisms between kernel and application. If this is the case, then one may trust that the operating system is well-behaved and focus on the application state. Some operating systems provide a thread abstraction with an individual program counter per thread. Although threads offer an additional level of abstraction, they also introduce additional complexity due to interactions between threads (e.g., shared variables, synchronization, timing aspects of thread scheduling).

4.3 Communication

The contents and timing of messages exchanged between sensor nodes does not only disclose the nature of interactions between sensor nodes (e.g., neighborhood relationships, routing topologies), but does also provide hints on the state of individual nodes. For example, a lack of messages from a certain node may indicate that the node has died, or the values of a sequence number contained in a message may indicate certain problems such as node reboots (after which the sequence number

counter in the node memory is reset to an initial value). Often dedicated messages carry system health information that is vital to the deployment, commissioning and operation of a sensor network application. As a number of deployment reports have shown, e.g., [42], expressiveness and simple, cleartext access to message payload is preferential over elaborate coding or even segmentation of data. In the case of segmentation of context over multiple packets, state cannot be reconstructed if one of the packets gets lost.

4.4 Environment

Sensor networks are deeply embedded into the physical world and hence the behavior of a sensor network is strongly dependent on the state of the environment. Not only does the state of the environment control the output of sensors attached to sensor nodes, but it also influences the wireless communication channel and hence communication as well as hardware performance (e.g., frequency of oscillators). Hence, additional infrastructure can be used (e.g., video cameras as in [41, 6]) to measure the state of the environment in a given deployment which is not only important to obtain ground truth for calibration of the system and individual sensors, but also to understand the reasons of certain failures, system behavior in general or degraded performance.

Some sensor networks apply techniques to process or aggregate sensor data in the network, such that the data collected from the sensor network does not disclose the output of individual sensors. In such settings it may be helpful to provide access to the raw output of the sensors in order to understand the system behavior.

5 Node Instrumentation

Node instrumentation is concerned with modifications of sensor nodes (both at the hardware and software levels) to allow access to the system state. One fundamental challenge here is to minimize interference with the application in the sense that introducing instrumentation should avoid so-called *probe effects*, where instrumentation (i.e., inserting a probe) results in a changed behavior of the system.

5.1 Software Instrumentation

Software instrumentation is required to retrieve the state of the software executing on the microcontroller. In practice, it is impossible (since there is not enough bandwidth to extract this information from the sensor node) and often also not necessary to extract the complete software state continuously. Instead, only a slice of the state (e.g., the values of a certain set of program variables) at certain points in time (e.g.,

when a certain program variable is modified, when a certain function is entered) suffices in many cases.

Source vs. Binary. Software instrumentation can be achieved at two levels: at the source level, where the source code of the software is modified, or at the binary level, where the binary program image (i.e., output of the assembler) is modified. At the source level, calls to specific functions are inserted, for example to log the value of a certain variable when a new value is assigned to this variable [35]. At the binary level, the binary CPU instructions are modified. A particular difficulty with this approach is that it is not easily possible to insert additional instructions as the code contains references (e.g., jump instructions) to certain addresses in the code image. By inserting instructions, these references would be invalidated. A common technique to avoid these problems are so-called *trampolines* [8, 44, 53], where an instruction X at address A in the code image is overwritten with a jump instruction to an address after the end of the binary image. At that address, a copy of the overwritten instruction X is placed, followed by instructions to retrieve the software state, followed by a jump instruction back to address A+1. This way, the layout of the original binary code image is not changed as the code to retrieve the software state is appended to the code image.

In some cases, instrumentation of the binary can be performed without actually modifying the binary code, by linking in additional code that makes references to the symbols (e.g., of variables or functions) defined in the binary. Marionette [49], for example, uses this approach to link RPC stubs with the binary to allow remote access to variables and functions in the code executing on the sensor node. The stubs are created by parsing an annotated version of the source code of the application.

Instrumentation at the source level is often easier to implement, but may significantly change the resulting binary code. For example, due to the inserted function calls the compiler may decide for a different optimization strategy, resulting in a significantly different binary code image being created. This may lead to pronounced *probe effects*, where the instrumented software behaves very different from the unmodified software due to changed memory layout, register allocations, or timing issues. In contrast, binary instrumentation is more complex to implement because one has to operate at the level of machine instructions, but probe effects are often less significant because the basic memory layout is not changed.

Operating System vs. Application. Another aspect of software instrumentation is whether the actual application code and/or the operating system is instrumented. In some cases it may be sufficient to only instrument the operating system or runtime environment to trace timing and parameters of kernel invocations, memory allocations, or context switches. In particular when using a virtual machine to interpret a byte code representation of the application program, an instrumentation of the byte code interpreter provides a detailed view of the applications state without actually instrumenting the application code. Some virtual machines have been developed for sensor nodes (e.g., [20]), but constrained resources often preclude the use of virtual machines low-end sensor nodes due to the resulting performance degradation.

Dynamic Instrumentation. Sometimes it is necessary to change the instrumentation during runtime, for example to implement interactive debugging, where one

wants to place watchpoints that are triggered when the value of a certain variable changes. While this is typically impossible with source code instrumentation as code would have to be recompiled and uploaded to the microcontroller, virtual machines and binary instrumentation do support this. A difficulty with binary instrumentation is that the program image resides in flash memory rather than in RAM. Changing the contents of flash memory is not only slow and consumes a lot of energy, but flash only supports a limited number of write cycles (in the order of thousands) before it fails [53]. Hence, changes of the binary instrumentation during runtime should be rather infrequent. This problem does not occur with virtual machines, as the interpreted program typically resides in data memory (i.e., RAM) – only the virtual machine itself resides in flash memory.

Specifications. One further aspect of software instrumentation is the way the user specifies how the software should be instrumented, i.e., which slice of the software state should be retrieved at what point in time. The most basic way is to specify the exact locations in the program code (either at source or binary level) where an instrumentation should be placed and which slice of the program state this instrumentation should retrieve. However, this is often a tedious and error-prone process. For example, when monitoring the value of a certain program variable over time, one may forget to place an instrumentation after one of the many places in the program where the value of this variable may be changed.

A more advanced approach inspired by Aspect-Oriented Programming (AOP) is to let the user specify a certain pattern such that instrumentation is inserted wherever the program code matches this pattern [8]. Example patterns may be of the form "whenever a function named *set** is entered, insert a given instrumentation" or "whenever the value of a variable named *state** is modified, insert a given instrumentation". An interpreter would read these patterns, match them with the program code (either source or binary) and place the appropriate instrumentation.

A related approach is based on the notion of events [50]. Here, an event is said to occur when the software assumes a certain predefined state. Instrumentation has to be added at all points where the software enters a state that matches the event definition. One basic way to implement this is to let the user identify all these points in the program. But in principle, this can also be automated. Upon occurrence of an event, a notification is generated that identifies the event, often also containing the time of occurrence and a certain slice of the software state at the time of occurrence of the event. One example of such events are watchpoints in a debugger [53].

Systems that support pattern-based or event-based instrumentation during runtime typically include a small runtime system to evaluate the pattern or event specification in order to identify the points in the code where instrumentation has to be placed, and to perform the actual instrumentation [8, 44, 53].

5.2 Hardware Instrumentation

Hardware instrumentation enables the extraction of the hardware state but also of the software state from a sensor node. Note that a hardware instrumentation is not

strictly necessary. For example, one may send off the software state (extracted using the techniques described in the previous section) using the radio or by blinking LEDs. Here, one can place an additional radio receiver or a video camera next to the sensor nodes to capture the state without physical access to the sensor nodes. In some cases it is even possible to retrieve parts of the hardware state in this way. For example, one may use the built-in analog-to-digital converter of the microcontroller to measure the battery voltage and send it off via radio or LEDs. However, all these techniques may result in significant changes of the system behavior and thus may cause probe effects. Hardware instrumentation tries to minimize these effects, but requires physical access to the sensors nodes, which may be difficult in the field.

Hardware instrumentation of sensor nodes uses the approaches and mechanisms known from embedded systems [15]. However, for WSN there is a considerable distinction: a single embedded system is only part of an interacting network. Thus in the following we describe only approaches from debugging embedded systems, which are focussed on a system aspect. Traditional embedded debugging methods for a single node such as flash monitors are not considered. We focus on hardware instrumentation utilizing a physical connection to the sensor node to extract logical monitoring information. On the sensor nodes, microcontroller interfaces are used to drive signals across a wired connection to a monitoring observer. The connection to an observer may include a converter translating the serial protocol to a standard interface such as USB or Ethernet.

The simplest approach for hardware instrumentation is to use General-Purpose IO's to generate binary flags, representing internal state of the node. While information content is limited, such wired pins are very helpful in debugging and have a minimal overhead in software. Further information can be extracted by using a serial protocol over the wire. Many microcontrollers provide support for serial protocols in their interfaces; in particular UARTs (universal asynchronous receiver/transmitters) are typically included. These interfaces can be used to send monitoring messages from the node to an observer. The additional information content is payed by increased, non-deterministic latency of an information message due to buffering and serialization. Flow control and error handling may be incorporated depending on monitoring message frequency and instrumentation requirements.

Additionally, microcontrollers typically include JTAG (IEEE 1149.1) functionality. JTAG stops the microcontroller and subsequently allows for non-intrusive extraction of node state to the external observer.[1] While, this method is very helpful in debugging individual nodes, the stopping of the execution induces considerable probing effects due to the halting of interaction with the system.

A separate HW instrumentation technique for extracting physical information is observation of power consumption through voltage and current measurements. While this is traditionally performed with expensive lab equipment such as voltage/current meters or oscilloscopes, in [16] a low-cost and distributed hardware

[1] Note that our discussion only focusses on information extraction and not on the debugging capabilities of JTAG such as stepping through a program execution.

instrumentation for power measurements on a testbed is presented. As sensor nodes have significant differences in power consumption for the microcontroller and the radio, the power consumption trace allows for extracting information on system state. A different approach is to focus on energy consumption of different operations. Using specialized hardware on each sensor node [11], energy attribution to individual tasks [13] may be derived allowing for additional insights.

6 Network Instrumentation Methods

A number of methods have been developed in recent years to aid development, testing and deployments of whole sensor network systems. The sheer number of nodes requires a careful and orchestrated instrumentation in order to allow to interact with a whole network of nodes. In this sense, *network instrumentation* refers to the instrumentation of sensor nodes to extract a comprehensive view of a sensor network. There are four different mechanisms for network instrumentation (i) in-band collection, (ii) local logging, (iii) sniffing and (iv) out-of-band collection. These approaches will be discussed in the following.

Table 4 Characteristics of network instrumentation methods based on distinct parameters: online capabilities, supported data volume and physical attachment

Approach	Online	Data volume	Attachment
In-Band	Yes	Low	No
Storage	No	Low	No
On-line Sniffing	Yes	Low	No
Off-line Sniffing	No	Low	No
Out-of-Band	Yes	High	Yes

With *in-band collection* as shown in Fig. 2 monitoring messages are routed through the sensor network to a sink, where they are merged and fed to the evaluation backend. This is similar to a data gathering approach to collect sensor values from the network. This approach has the lowest overhead as no additional hardware is required, but results in high interference with the application as all traffic is transmitted in-band with application traffic through the whole network.

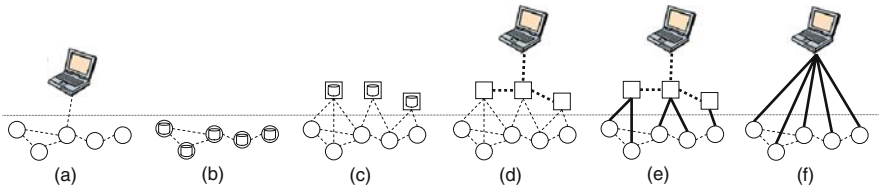


Fig. 2 Different network instrumentation methods have implications on resources used and the expressiveness of the results: (**a**) in-band collection (**b**) logging (**c**) offline sniffing (**d**) online sniffing (**e**) wireless testbed (**f**) wired testbed

With the *logging* approach in Fig. 2(b), monitoring messages are stored to a local node memory (flash) indicated by the small database symbols in the nodes. While this approach causes no interference with the application traffic, the nodes need to be collected to download the traces from their memories, resulting in substantial overhead. As local storage is typically limited by sensor nodes, the amount of monitoring messages is considerably limited.

With the *offline sniffer* approach in Fig. 2(c), an additional set of sniffer nodes (depicted as squares) is installed alongside the sensor network to overhear messages messages sent by sensor nodes. Sniffer nodes store these messages in their (flash) memories. As in the logging approach, sniffer nodes need to be collected to download messages from their memories, while the sensor network remains operational. This approach may be used in a completely passive manner without modifying sensor network software or protocols as in [9]. However, one may also modify the sensor network protocols to include additional information about node states into messages or to broadcast additional messages in-band with the application traffic to be overheard by the sniffer nodes. However, other sensor nodes would ignore these additional messages and would not forward them, resulting in significantly less in-band traffic compared to the in-band collection.

With the *online sniffer* approach in Fig. 2(d), sniffer nodes have a second, powerful radio that is free of interference with the sensor network radio (e.g., Bluetooth, WLAN). Using this second radio, sniffer nodes forward overheard monitoring messages to a dedicated sink where they are merged and fed to the backend for evaluation. In contrast to the *offline sniffer* approach, traces are processed online, but a reliable second radio channel is required. This approach has been used in [34].

Out-of-band logging avoides completely avoids the use of the sensor node radio for monitoring purposes to as to minimize interference with the application. Instead, a second, "out-of-band" communication channel is created by wiring the sensor nodes. However the implementation may differ concerning the transport mechanism for monitoring messages. A *wireless testbed* approach in Fig. 2(e) is similar to the online sniffer, but instead of sending monitoring messages in-band with application traffic, each sensor node is connected by wire (e.g., serial cable) to an additional node, a so-called observer that forwards the messages over the second radio channel to the sink. This approach results in even less interference and message loss than *online sniffing*, but requires substantial overhead for wiring. This approach has been used in [12]. Finally, with the *wired testbed* approach in Fig. 2(f), each node is wired to a sink. Monitoring messages are transmitted over the wire to a sink, where they are merged and fed to the evaluation backend in an online fashion. Many testbeds exist that support such a wired channel to each node, e.g., [48]. In fact, wired testbeds are the most common instrumentation method used to date. However, this approach is typically only feasible in the lab, while the other approaches could be applied both in the lab and in the field.

A basic concept in all instrumentation methods is that of an *observer*, be it integrated as in the in-band case, or external as in the out-of-band case. Any state that wants to be observed needs to be preserved and made available for extraction and

further analysis. By doing so, any method used ultimately influences the system it is applied on. Even in the case of a passive sniffer, care must be taken that the relevant data necessary for successful observation and analysis is actually exported. Here, a protocol designer thus must foresee provisions in the protocol design that allow to reconstruct the necessary context. Additionally, all network instrumentation methods rely on successful node instrumentation, especially w.r.t the preservation of timing properties. As stated earlier, wired testbeds are the most commonly used approach today. However, most testbeds are limited to distinct areas only, e.g., an office building. As testing has to take into account the influence of the actual environment to be encountered at the deployment site, special equipment is required to simulate environmental impact, for example by incorporating a temperature cycling chamber into a testbed setup.

7 Analyzing the System

Once access to the system state is provided through node and network instrumentation, system state can be analyzed, e.g. to detect failures. This analysis can be performed within the network on the sensor nodes themselves, outside of the sensor network, or with a mixture of both approaches (e.g., some preprocessing is performed in the sensor network to reduce the amount of information that has to be transmitted).

A fundamental problem that has to be addressed in analyzing the system state is incomplete information. Not only do limited node and network resources preclude to extract the complete system state, but often wireless communication with inherent message loss is used to extract the system state.

7.1 Monitoring and Visualization

Visualization of system parameters and state is an invaluable tool for immediate feedback. A number of traditional network analysis tools such as Ganglia, Cacti or Nagios can be readily applied [5], but especially at the deployment site handheld visual inspection devices [22, 3] provide insight into the operation of the network and individual nodes. As an example, SeeDTV [22] provides a handheld device, SeeMote, providing visual support, while being small, light-weight, and providing long battery lifetime. The SeeDTV application provides different views on the information provided by the sensor network such as statistics on the number of available nodes, status and health (battery level) of an individual selected node, and a detailed view onto the node's sampled ADC values. SeeDTV was used in the Luster deployment and helped to pinpoint 5 malfunctioning nodes, which could be immediately replaced.

With a focus on the development stage, Octopus [36] is an interactive tool allowing for visualizing information of system characteristics of a sensor network.

It additionally provides an interface for interactive reconfiguration of application parameters. As an example, a user may change the duty-cycle and observe the effects on the network.

7.2 Inferring Network State from Node States

Often a user is overwhelmed by monitoring the low-level system state extracted from the sensor nodes. Here, the low-level traces from multiple nodes can be combined to interfere a higher-level state of the network. For example, instead of looking at the individual routing tables of nodes, one may reconstruct the routing topology of the network and display it to the user.

SNIF [34] allows to infer the network state from message traces collected with a sniffer network. Inference is implemented with a data stream framework. The basic element of this framework is a data stream operator which accepts a data stream (e.g., a stream of overheard messages) as input, processes the stream (e.g., by removing elements from the stream or modifying their contents), and outputs another data stream. These operators can be chained together to form a directed acyclic graph. There are general-purpose operators that can be configured with parameters (e.g., a union operator that merges two data streams) and custom operators which are implemented by a user for a specific inference task. Ideally, one should be able to implement a given inference task just by configuring and combining existing general-purpose operators. However, in practice it is often necessary to implement some custom operators. LiveNet [9] provides a similar functionality, however the inference functionality is implemented in an ad-hoc manner using a general-purpose programming language.

EvAnT and the successive Rupeas DSL [50, 51] provide an event analysis framework for WSNs. Input to the system is a trace of log events collected from the execution. Each event is a tuple of key-value pairs, minimally including a node identifier and a type. Traces are represented as sets of events with no particular ordering on events implied. Rather the user may specify a strict ordering using timestamps or a partial order based on message ordering on sent and received packets. EvAnT allows for formulating queries and assertions on the trace. Queries are based on the EvAnT operators, which use simple predicates and associating functions for combining events. As an example, a routing path is composed by iteratively associating individual send/receive and forward links across the nodes. A case study in [50] presents how concise formulations in EvAnT extract information about message flow. In particular, through subsequent queries on the trace, the underlying problem of a low-power data gathering application is traced back to time synchronization of the MAC protocol layer.

7.3 Failure Detection

Failure detection is concerned with automatically identifying system states that represent failures (i.e., deviations from the system specification). Hence, instead of

manually scanning traces of node or network states, problematic system states are automatically identified.

There are two orthogonal approaches to failure detection. Firstly, one can specify the correct behavior of the system and deviations from this correct behavior are automatically detected and assumed to be failures. Secondly, one can directly specify faulty behavior and the actual system state is then matched with the failure specifications.

The first approach has the advantage that potentially any failure can be detected, even ones that are unknown a priori. However, it is often non-trivial to specify the correct behavior of the system as a wide range of different systems states may represent correct behavior. This is especially true for failures that result from interactions of multiple nodes, because here one has to specify the correct behavior of the whole network. In contrast, specifying the correct behavior of a single node is typically much easier.

One approach to deal with this problem is to focus on certain aspects of the system behavior instead of trying to specify the complete system behavior. One example for this approach are assertions, where the user can specify a predicate over a slice of the system state, formulating the hypothesis that this predicate should always be true. If the predicate is violated, a failure is detected. Note, however, that the user may also specify an incorrect hypothesis, so a failed assertion does not necessarily indicate a failure. Assertions can be non-local both in time (refer to system state obtained at different points in time) and space (refer to the state of multiple nodes). For example, passive distributed assertions [35] are local in time but non-local in space. Other examples for the use of assertions are [35, 44, 50, 23].

Another approach to address the problem of specifying the correct behavior of a system is to use machine learning techniques to automatically build a model of the system state during periods when the system is known to behave correctly. In Dustminer [18], for example, the frequency of occurrence of certain sequences of events is computed while the system is known to work correctly. If at a later point in time the frequency of a certain event sequence deviates significantly from the "correct" frequency, a failure is detected.

It is often easier to directly specify the system behavior that represents a very specific failure. However, with this approach only failures can be detected that are known a priori. Many systems build on this approach. For example, SNIF [34] and Sympathy [32] both offer failure detectors for node crash, node reboot, isolated nodes with no neighbors, no path to/from sink, and others.

7.4 Root Cause Analysis

It is often the case that a failure may cause a secondary failure. For example, if a node crashes (primary failure), then this may lead to a situation where other nodes fail to route data to the sink (secondary failure) because the crashed node was the only connection to the sink. Here, a user would be interested in knowing the primary

failure (i.e., the root cause) instead of being overwhelmed with all sorts of secondary failures.

One technique that has been applied to sensor networks for root cause analysis are so-called decision trees [32, 34]. As illustrated in Fig. 3 taken from [32], a decision tree is a binary tree where each internal node is a detector for a specific failure that may either output *Yes* (failure detected) or *No* (failure not detected). The leaf nodes indicate possible failures that can be detected. To process the decision tree, we start at the root and execute the failure detector. Depending on the output of the failure detector, we proceed to one of the child nodes. If the child node is a leaf, the failure noted in the leaf node is output and we are done. Otherwise, if the node is an inner node, we evaluate the respective failure detector and so on until we arrive at a leaf node.

Decisions trees in [32, 34] are empirically constructed, according to the rule that if failure A can result in a secondary failure B, then the failure detector node for A should be located on the path from the root of the tree to the node representing the failure detector for B.

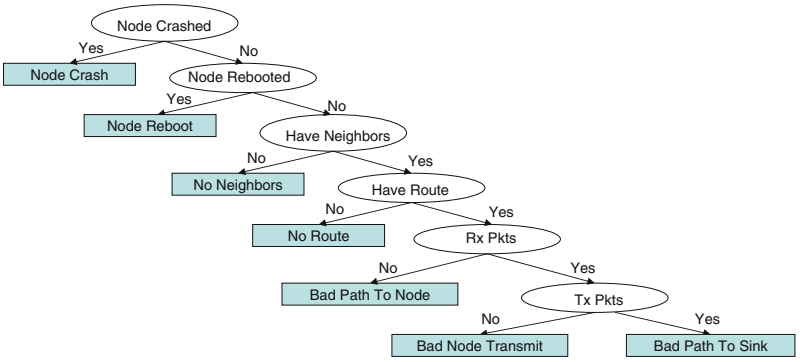


Fig. 3 The decision tree for root cause analysis used by Sympathy [32]

7.5 Node-Level Debugging

Traditional source-level debugging techniques such as placing breakpoints, watchpoints, or single-stepping can also be applied to sensor nodes. For example, in [53], the GNU debugger gdb is ported to TinyOS. However, this approach is limited as a sensor node is merely part of a larger network of nodes that often operate under real-time constraints. Since the timing of program execution is significantly affected by the debugger, significant probe effects are caused. In the extreme case, the slowed-down execution of the debugged node may itself lead to failures. Another problem with this approach is that dynamic modifications of the binary code executing on the microcontroller are needed to implement the debugger. As the program

image is stored in flash memory, this leads to excessive wear of the flash memory and high latency.

7.6 Replay and Checkpointing

Another analysis technique is to reproduce a faulty execution either directly in the deployed network or in a controlled environment (e.g., lab testbed or simulator). By varying some of the system parameters during such a replay, one can examine potential dependencies between these parameters and the faulty behavior, verify that a certain modification of the system has removed the cause of a problem, or simply tune the performance of the system.

Envirolog [24] supports repeatability of system executions albeit asynchronous events. This is grounded on the assumption of scoped event readings, i.e., data is only transferred via dedicated functions and associated parameters. A log module is responsible for logging events of interest, i.e., the according function calls with their parameters into flash memory, thus creating a timestamped event record. In a subsequent replayed execution, consumers of these events are re-wired to connect to the replay module, providing the recorded events from flash at the replayed instant in time. While this allows for repeated tests of different parameter settings, e.g., for protocol tuning, the results are only valid as long as the re-executions do not change the occurrence and timing of events. This fundamental problem of timing dependence is avoided with a model-based approach such as Emuli. Emuli [1] focusses on the capability to emulate network-wide sensing data to sensor nodes. This allows for analyzing the system execution on a model-based ground truth, whether it is used for repeatability or exploratory analysis for novel sensing data. Emuli provides time synchronization and coordination among nodes in order to generate a consistent network wide model of sensing data.

Checkpointing [27] extracts system state allowing for analysis, transferral between different execution (test) platforms, e.g., a simulator and a testbed, and for repeatability. However, in contrast to replay repeatability discussed above, checkpointing only concerns the initial state of an execution, not the subsequent execution itself. Nevertheless, a consistent state across the system, e.g., concerning the topology and neighbor tables in network-centric tests, is invaluable for comparative tests and analysis of parameter changes. Checkpointing of a single sensor node concerns the state of the individual components. As program code in program flash is typically not modified, only RAM state needs to be extracted, which is typically already provided by the bootloader. Other components such as External Flash, LEDs, and especially the radio require special considerations dependent on the requirements of application and the checkpoint. Dunkels et al. [27] describe the support in Contiki for storing and loading node state and its current scope. However, the difficulty of checkpointing lies in the synchronized checkpointing across all nodes in the network for a consistent snapshot. This is handled by intermittent Linux routers controlling a subset of nodes for freezing and unfreezing.

8 Concluding Remarks

In this contribution we have surveyed the most prominent sensor network deployments and been able to identify selected underlying problems in system design, during installation and deployments. A special focus has been made towards problems arising and relevant in practice with sensor network applications and deployments in a real environment. We have presented a number of techniques for the instrumentation and analysis, predominantly to be applied at run-time of a sensor network. With the techniques presented here, a designer of networked embedded systems should be able to define a suitable and successful design strategy suitable to meet the requirements specified. With regard to the distributed nature of sensor networks the typical problems encountered have been classified into node, link, network path as well as global problems. Likewise, the methods presented in this chapter help to increase the understanding at various levels, namely at the node, network, and system levels. Due to the diversity of applications, requirements and design goals, there is no single, distinctive approach to the design and deployment of sensor networks available today.

References

1. Alam, N., Clouser, T., Thomas, R., Nesterenko, M.: Emuli: model driven sensor stimuli for experimentation. In: Proc. 6th ACM Conf. Embedded Networked Sensor Systems (SenSys 2008), ACM Press, New York (2008) 423–424
2. Arora, A., Dutta, P., Bapat, S., Kulathumani, V., Zhang, H., Naik, V., Mittal, V., Cao, H., Demirbas, M., Gouda, M., Choi, Y., Herman, T., Kulkarni, S., Arumugam, U., Nesterenko, M., Vora, A., Miyashita, M.: A line in the sand: a wireless sensor network for target detection, classification, and tracking. Computer Networks 46(5) (2004) 605–634
3. Barrenetxea, G., Ingelrest, F., Schaefer, G., Vetterli, M.: The hitchhiker's guide to successful wireless sensor network deployments. In: Proc. 6th ACM Conf. Embedded Networked Sensor Systems (SenSys 2008), ACM Press, New York (2008) 43–56
4. Barrenetxea, G., Ingelrest, F., Schaefer, G., Vetterli, M., Couach, O., Parlange, M.: Sensorscope: Out-of-the-box environmental monitoring. In: Proc. 7th Int'l Conf. Information Processing Sensor Networks (IPSN '08), IEEE CS Press, Los Alamitos, CA (2008) 332–343
5. Beutel, J., Dyer, M., Martin, K.: Sensor network maintenance toolkit. In: Proc. 3rd European Workshop on Sensor Networks (EWSN 2006). (February 2006) 58–59
6. Bonnet, P., Leopold, M., Madsen, K.: Hogthrob: Towards a sensor network infrastructure for sow monitoring (wireless sensor network special day). In: Proc. Conf. Design, Automation and Test in Europe (DATE 2006), IEEE, Piscataway, NJ (March 2006)
7. Burns, R., Terzis, A., Franklin, M.: Design tools for sensor-based science. In: Proc. 3rd IEEE Workshop on Embedded Networked Sensors (EmNetS-III), Cambridge, Massachusetts, USA (May 2006)
8. Cao, Q., Abdelzaher, T.F., Stankovic, J.A., Whitehouse, K., Luo, L.: Declarative tracepoints: A programmable and application independent debugging system for wireless sensor networks. In: Proc. 6th ACM Conf. Embedded Networked Sensor Systems (SenSys 2008), ACM Press, New York (November 2008) 85–98
9. Chen, B., Peterson, G., Mainland, G., Welsh, M.: Livenet: Using passive monitoring to reconstruct sensor network dynamics. In: Proceedings of the 4th IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS 2008), Santorini Island, Greece (June 2008) 79–98

10. Dubois-Ferrière, H., Fabre, L., Meier, R., Metrailler, P.: Tinynode: A comprehensive platform for wireless sensor network applications. In: Proc. 5th Int'l Conf. Information Processing Sensor Networks (IPSN '06), ACM Press, New York (2006) 358–365
11. Dutta, P., Feldmeier, M., Paradiso, J., Culler, D.: Energy metering for free: Augmenting switching regulators for real-time monitoring. In: Proc. 7th Int'l Conf. Information Processing Sensor Networks (IPSN '08), ACM Press, New York (April 2008) 283–294
12. Dyer, M., Beutel, J., Kalt, T., Oehen, P., Thiele, L., Martin, K., Blum, P.: Deployment Support Network – A toolkit for the development of WSNs. In: Proceedings of the 4th European Conference on Wireless Sensor Networks (EWSN 2007), Delft, The Netherlands (2007) 195–211
13. Fonseca, R., Dutta, P., Levis, P., Stoica, I.: Quanto: Tracking energy in networked embedded systems. In: Proc. 9th Symp. Operating Systems Design and Implementation (OSDI '08), ACM Press, New York (2008) 323–338
14. Ganesan, D., Krishnamachari, B., Woo, A., Culler, D., Estrin, D., Wicker, S.: Complex behavior at scale: An experimental study of low-power wireless sensor networks. Technical Report CSD-TR 02–0013, UCLA, Los Angeles, California, USA (2002)
15. Ganssle, J.G.: The Art of Programming Embedded Systems. Academic Press, Inc., Orlando, FL, USA (1992)
16. Haratcherev, I., Halkes, G., Parker, T., Visser, O., Langendoen, K.: PowerBench: A Scalable Testbed Infrastructure for Benchmarking Power Consumption. In: Int. Workshop on Sensor Network Engineering (IWSNE) (2008)
17. Intanagonwiwat, C., Govindan, R., Estrin, D., Heidemann, J., Silva, F.: Directed diffusion for wireless sensor networking. IEEE/ACM Transactions on Networking 11(1) (2003) 2–16
18. Khan, M.M.H., Le, H.K., Ahmadi, H., Abdelzaher, T.F., Han, J.: Dustminer: troubleshooting interactive complexity bugs in sensor networks. In: Proc. 6th ACM Conf. Embedded Networked Sensor Systems (SenSys 2008), ACM Press, New York (November 2008) 99–112
19. Langendoen, K., Baggio, A., Visser, O.: Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In: Proc. 20th Int'l Parallel and Distributed Processing Symposium (IPDPS 2006), IEEE, Piscataway, NJ (April 2006) 8–15, 14th Int'l Workshop Parallel and Distributed Real-Time Systems (WPDRTS 2006).
20. Levis, P., Culler, D.E.: Maté: A tiny virtual machine for sensor networks. In: Proc. 10th Int'l Conf. Architectural Support Programming Languages and Operating Systems (ASPLOS-X), ACM Press, New York (2002) 85–95
21. Lim, R., Woehrle, M., Meier, A., Beutel, J.: Harvester: Energy savings through synchronized low-power listening. In: Proc. EWSN 2009 – Demos/Posters Session, Department of Computer Science, University College Cork, Ireland (February 2009) 29–30
22. Liu, H., Selavo, L., Stankovic, J.: SeeDTV: Deployment-time validation for wireless sensor networks. In: Proc. 4th IEEE Workshop on Embedded Networked Sensors (EmNetS-IV), ACM Press, New York (2007) 23–27
23. Lodder, M., Halkes, G.P., Langendoen, K.G.: A global-state perspective on sensor network debugging. In: Proc. 5th ACM Workshop on Embedded Networked Sensors (HotEmNets 2008). (June 2008)
24. Luo, L., He, T., Zhou, G., Gu, L., Abdelzaher, T.F., Stankovic, J.A.: Achieving repeatability of asynchronous events in wireless sensor networks with envirolog. In: INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings (April 2006) 1–14
25. Madden, S., Franklin, M., Hellerstein, J., Hong, W.: TinyDB: An acqusitional query processing system for sensor networks. ACM Transactions on Database Systems 30(1) (2005) 122–173
26. Mainwaring, A., Culler, D., Polastre, J., Szewczyk, R., Anderson, J.: Wireless sensor networks for habitat monitoring. In: Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA), Atlanta, Georgia, USA (September 2002) 88–97
27. Oesterlind, F., Dunkels, A., Voigt, T., Tsiftes, N., Eriksson, J., Finne, N.: Sensornet checkpointing: Enabling repeatability in testbeds and realism in simulators. In: Proc. 6th European Conference on Wireless Sensor Networks (EWSN 2009). Volume 5432 of Lecture Notes in Computer Science, Springer, Berlin (February 2009) 343–357

28. Padhy, P., Martinez, K., Riddoch, A., Ong, H.L.R., Hart, J.K.: Glacial environment monitoring using sensor networks. In: Proceedings of the Workshop on Real-World Wireless Sensor Networks (REALWSN) (2005)
29. Paek, J., Chintalapudi, K., Govindan, R., Caffrey, J., Masri, S.: A wireless sensor network for structural health monitoring: Performance and experience. In: Proc. 2nd IEEE Workshop on Embedded Networked Sensors (EmNets 2005) (May 2005) 1–9
30. Polastre, J., Szewczyk, R., Culler, D.: Telos: Enabling ultra-low power wireless research. In: Proc. 4th Int'l Conf. Information Processing Sensor Networks (IPSN '05), IEEE, Piscataway, NJ (April 2005) 364–369
31. Polastre, J.P., Szewczyk, R., Mainwaring, A., Culler, D., Anderson, J.: Analysis of wireless sensor networks for habitat monitoring. In Raghavendra, C.S., Sivalingam, K.M., Znati, T., eds.: Wireless Sensor Networks. Kluwer Academic Publishers, Dordrecht (2004)
32. Ramanathan, N., Chang, K., Kapur, R., Girod, L., Kohler, E., Estrin, D.: Sympathy for the sensor network debugger. In: Proceedings of the 3rd ACM Conference on Embedded Networked Sensor Systems (SenSys), Los Angeles, California, USA (November 2005) 255–267
33. Ramanathan, N., Kohler, E., Estrin, D.: Towards a debugging systems for sensor networks. International Journal of Network Management 15(4) (July 2005) 223–234
34. Ringwald, M., Römer, K., Vitaletti, A.: Passive inspection of wireless sensor networks. In: Proceedings of the 3rd International Conference on Distributed Computing in Sensor Systems (DCOSS 2007) (2007)
35. Römer, K.: PDA: Passive distributed assertions for sensor networks. In: Proc. 8th Int'l Conf. Information Processing Sensor Networks (IPSN '09), ACM Press, New York (April 2009)
36. Ruzzelli, A., Jurdak, R., Dragone, M., Barbirato, A., O'Hare, G., Boivineau, S., Roy, V.: Octopus: A dashboard for sensor networks visual control. In: MobiCom 2008. (2008)
37. Selavo, L., Wood, A., Cao, Q., Sookoor, T., Liu, H., Srinivasan, A., Wu, Y., Kang, W., Stankovic, J., Young, D., Porter, J.: Luster: wireless sensor network for environmental research. In: Proc. 5th ACM Conf. Embedded Networked Sensor Systems (SenSys 2007), ACM Press, New York (2007) 103–116
38. Sobeih, A., Viswanathan, M., Marinov, D., Hou, J.C.: Finding bugs in network protocols using simulation code and protocol-specific heuristics. In: Proceedings of the 7th International Conference on Formal Engineering Methods (ICFEM), Manchester, United Kingdom, Springer (2005) 235–250
39. Stoianov, I., Nachman, L., Madden, S., Tokmouline, T.: Pipenet – a wireless sensor network for pipeline monitoring. In: Proc. 6th Int'l Conf. Information Processing Sensor Networks (IPSN '07), ACM Press, New York (2007) 264–273
40. Szewcyk, R., Mainwaring, A., Polastre, J., Anderson, J., Culler, D.: An analysis of a large scale habitat monitoring application. In: Proc. 2nd ACM Conf. Embedded Networked Sensor Systems (SenSys 2004), Baltimore, Maryland, USA, ACM Press, New York (November 2004) 214–226
41. Szewczyk, R., Polastre, J., Mainwaring, A., Culler, D.: Lessons from a sensor network expedition. In: Proc. 1st European Workshop on Sensor Networks (EWSN 2004). Volume 2920 of Lecture Notes in Computer Science, Springer, Berlin (January 2004) 307–322
42. Talzi, I., Hasler, A., Gruber, S., Tschudin, C.: PermaSense: Investigating permafrost with a WSN in the Swiss Alps. In: Proc. 4th IEEE Workshop on Embedded Networked Sensors (EmNetS-IV), ACM Press, New York (2007) 8–12
43. Tateson, J., Roadknight, C., Gonzalez, A., Fitz, S., Boyd, N., Vincent, C., Marshall, I.: Real world issues in deploying a wireless sensor network for oceanography. In: Proc. Workshop on Real-World Wireless Sensor Networks (REALWSN 2005) (June 2005)
44. Tavakoli, A., Culler, D., Shenker, S.: The case for predicate-oriented debugging of sensornets. In: Proc. 5th ACM Workshop on Embedded Networked Sensors (HotEmNets 2008) (June 2008)
45. Tilak, S., Abu-Ghazaleh, N.B., Heinzelman, W.R.: A taxonomy of wireless micro-sensor network models. ACM SIGMOBILE Mobile Computing and Communications Review (MC2R) 6(2) (April 2002) 28–36

46. Tolle, G., Polastre, J., Szewczyk, R., Culler, D., Turner, N., Tu, K., Burgess, S., Dawson, T., Buonadonna, P., Gay, D., Hong, W.: A macroscope in the redwoods. In: Proc. 3rd ACM Conf. Embedded Networked Sensor Systems (SenSys 2005) (November 2005) 51–63
47. Werner-Allen, G., Lorincz, K., Johnson, J., Lees, J., Welsh, M.: Fidelity and yield in a volcano monitoring sensor network. In: Proc. 7th Symp. Operating Systems Design and Implementation (OSDI '06), ACM Press, New York (2006) 381–396
48. Werner-Allen, G., Swieskowski, P., Welsh, M.: Motelab: a wireless sensor network testbed. In: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN), Los Angeles, California, USA (2005) 483–488
49. Whitehouse, K., Tolle, G., Taneja, J., Sharp, C., Kim, S., Jeong, J., Hui, J., Dutta, P., Culler, D.: Marionette: using RPC for interactive development and debugging of wireless embedded networks. In: Proc. 5th Int'l Conf. Information Processing Sensor Networks (IPSN '06), ACM Press, New York (April 2006) 416–23
50. Woehrle, M., Plessl, C., Lim, R., Beutel, J., Thiele, L.: EvAnT: Analysis and checking of event traces for wireless sensor networks. In: Proc. Int'l Conf. Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2008). (June 2008) 201–208
51. Woehrle, M., Plessl, C., Thiele, L.: Poster abstract: Rupeas – an event analysis language for wireless sensor network traces. In: Adjunct Proc. 6th European Workshop on Sensor Networks (EWSN 2009), Cork, Ireland (February 2009) 19–20
52. Woo, A., Tong, T., Culler, D.: Taming the underlying challenges of reliable multihop routing in sensor networks. In: Proc. 1st ACM Conf. Embedded Networked Sensor Systems (SenSys 2003), ACM Press, New York (2003) 14–27
53. Yang, J., Soffa, M.L., Selavo, L., Whitehouse, K.: Clairvoyant: A comprehensive sourcelevel debugger for wireless sensor networks. In: Proc. 5th ACM Conf. Embedded Networked Sensor Systems (SenSys 2007), ACM Press, New York (November 2007) 189–203

Static and Dynamic Localization Techniques for Wireless Sensor Networks

Jean-Michel Dricot, Gianluca Bontempi, and Philippe De Doncker

Abstract In this chapter, we present several user localization techniques used in the context of wireless sensors networks. These techniques are introduced with respect to the three different stages followed in any localization process: (i) the signal acquisition and the derivation of the corresponding distance model, (ii) the terminal positioning algorithm, and (iii) the filtering of the estimates over time in order to provide a dynamic tracking. Moreover, it is shown how an additional sensor (i.e., an accelerometer here) can be included in the filter model. The emphasizes is put on the fusion of the informations derived from the sensors in order to significantly refine the localization accuracy.

1 Introduction

The term *sensor network* refers to a set of numerous sensors that are equipped with processor, memory, and wireless communication capabilities. These nodes are referred to as *motes*. A fundamental problem in sensor networks is localization, i.e., determining the location of the sensors. The location information is then used to geographically tag the data, perform user tracking, secure informations based on the location [32], enhance a routing protocol or support the resource discovery process [22]. In most scenario, the user carries a single sensor, referred to as *mobile beacon* (or *target*) and its location is inferred with respect to the known position of the fixed sensors of the network also called *anchors nodes*.

A Generic framework for the localization of target nodes is presented in Fig. 1. More precisely, localization algorithms can be divided into three main categories: *range-free*, *range-based*, and *anchor-free* algorithms. In *range-free* algorithms, the nodes estimate their locations by using (i) the known location information of their neighbours and (ii) geometrical considerations. It is assumed that there is no

J.-M. Dricot (✉)
Wireless Communications Group, Université Libre de Bruxelles– OPERA Dpt., Bruxelles, Belgium
e-mail: jdricot@ulb.ac.be

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_10,

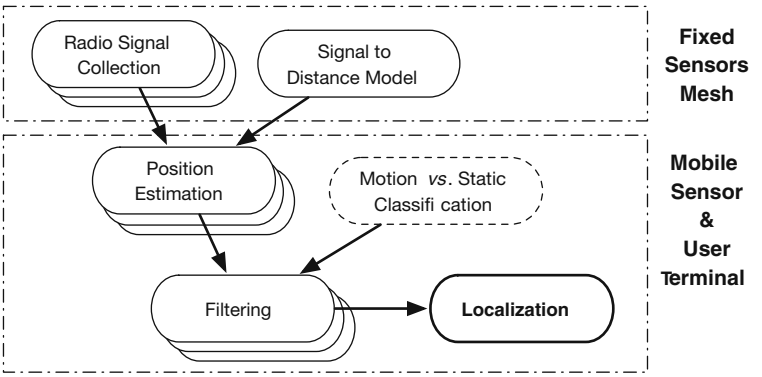


Fig. 1 The conceptual architecture of a localization framework

distance, angle, or any metric information available at each node [14, 23, 16, 27, 9]. For instance, the position of the target node is estimated as the centroid of the positions of the surrounding anchor nodes or by means of more complex geometrical relations. Three range-free localization techniques are presented in Sect. 2: the centroid, the bounding box, and the point-in-triangle techniques. Though these approaches offer the lowest level of precision, they are simple to implement, generate a low overhead in the network communications, and are accurate in most situations.

Range-based (also referred to as *anchor-based*) algorithms make use of the distance estimations between the mobile and the static nodes. The position of some nodes in the network (called anchors) are known a priori by means of an on-site measurement or by embedding a GPS receiver in these nodes. The target nodes try to estimate their position relative to these anchors. For instance, Bahl et al. [2] and Bischo et al. [3] used Received Signal Strength (RSS) to estimate distances, Ward et al. [38] used Time of Arrival of signals. The use of the Angle of Arrival of the signals was presented in [27]. Most of these methods require a high percentage of pre-deployed anchor nodes in the network. In Sect. 3, we present how a distance estimator can be computed by means of the radio signal strength, the angle of arrival, or the time of flight. Then, in Sect. 4 we present the circular and hyperbolic lateration techniques that are the most common approaches used in range-based localization. In a third class of localization techniques, the target nodes try to compute their position without the help of the anchors. Since it is not possible to derive the absolute positions of the target nodes, a relative coordinate system is established instead and the positions of the nodes are computed with respect to this marker. Relative location is of crucial importance in Location-Aided Routing (LAR) [24] and can be mapped to an absolute coordinate system by means of a limited amount of anchors equipped with GPS receivers.

Besides the *static* localization problem, the movement of the target can be taken into account in *dynamic* filtering and tracking schemes. In Sect. 5, a Kalman-class filter is presented and its expressions are derived in the scenario of a human-worn localization and tracking system.

Table 1 Notations used in this chapter, their corresponding interpretation and units

Symbol	Interpretation	Units
N	Total number of anchor nodes	–
$\mathbf{x}$	Exact position of the target node	[m]
$\tilde{\mathbf{x}}$	Position estimate of the target node	[m]
$\tilde{\mathbf{x}}_\mathbf{0}$	A first position estimate in the case of an iterative localization algorithm	[m]
$\mathbf{x}_i$	Known position of the i-th anchor node	[m]
$\tilde{\mathbf{x}}_i$	Position estimate of the i-th anchor node	[m]
d	Exact distance	[m]
d_i	Exact distance between the target and the i-th anchor node	[m]
$\tilde{d}$	Estimated (measured) distance	[m]
$\tilde{d}_i$	Estimated (measured) distance between the target and the i-th anchor node	[m]
α_i	Angle of arrival as measured at the i-th anchor node	[rad]
ε	Imprecision on the angle measurement	[rad]
n	Decay factor in wireless propagation (also referred to as *path loss exponent*)	–
P	Power of the wireless signal	[dB]
$\mathcal{N}(\mu, \sigma)$	Gaussian distribution with mean value μ and variance σ	–
ψ	Variance of the acceleration measured on a sensor worn by a human user	ms^{-2}
$\boldsymbol{\omega}_i$	Scalar value used as a weighting parameter	–
t_k	Time of the k-th sample	[s]
Δt	Period of time between two samples k and $k-1$	[s]
$\mathbf{x}_k$	Position of the target during the k-th sample	[m]
$\mathbf{s}_k$	Speed of the target during the k-th sample	[m]
$\mathbf{a}_k$	Acceleration of the target during the k-th sample	[m]

In Sect. 6, we discuss the notions of accuracy and precision. Indeed, the localization process is a noisy estimation and an absolute precision can never be achieved. Furthermore, some estimators may prove wrong and shall be detected and rejected automatically.

Section 7 is dedicated to more advanced localization techniques based on Machine Learning. The discussion starts with some insights on how fusion of sensors can be performed to further refine the tracking filter. A suitable non supervised linear classification technique for the data issued from the sensors is presented. This information is then used to parameterize a modified Kalman filter. The section ends with a presentation of data fusion techniques and a Monte Carlo-based method (referred to as *subsampling technique*) is presented. It aims at combining the multiple estimates and further improve the inferred position by limiting the impact of erroneous location estimators. Section 8 outlines some open problems and concludes the chapter. Finally, the notations used in this chapter are synthesized in Table 1.

2 Static Localization Techniques – Range-free

The algorithms presented in this is section are described as range-free since that they use Radio Signal Strength Indicator[1] (RSSI) value measurements but do not rely on an *accurate* RSSI-distance conversion model. Relative comparisons between the

[1] The RSSI is a measurement of the power present in a received radio signal.

RSSI values are used instead. The comparison functions have to be monotonic and calibrated to produce stable estimators or the relative distance between two nodes.

2.1 Weighted Centroid

The centroid method is a simple, efficient method that can be used in dense sensor networks. This method assumes that there exists a set of anchors having (i) a known position and (ii) and overlapping transmission zones. The idea of the centroid method is to estimate the position of a target as the average value (i.e., the geometrical centroid) of the positions of the anchor nodes in the transmission zone of the terminal. Furthermore, the positions of the anchors can be weighted. Figure 2 shows a pictorial illustration of this method. The unknown position is computed as

$$\tilde{\mathbf{x}} = \frac{\sum_{i=1}^{N} \omega_i \mathbf{x}_i}{\sum_{i=1}^{N} \omega_i}$$

where N is the amount of anchors in the transmission zone of the terminal and $\mathbf{x}_i$ is the position (known a priori) of the i-th anchor. Several strategies can be implemented to assign values to the weights ω_i. In [9, 10], the authors propose to compute the weights with respect to the ratio of successfully received beacons. More precisely, the mobile node broadcasts several HELLO messages. The anchors $\mathbf{x}_i$, $1 \leq i \leq N$ listen for beacons and compute a connectivity metric. Each weight is computed as follows:

$$\omega_i(t) = \frac{n^{(i)}_{\text{received}}(t)}{n^{(i)}_{\text{sent}}(t)}$$

where t is the time of reception of the broadcast messages, $n_{\text{received}}(t)$ and $n_{\text{sent}}(t)$ are the numbers of beacons received and sent up to the time t, respectively. A threshold

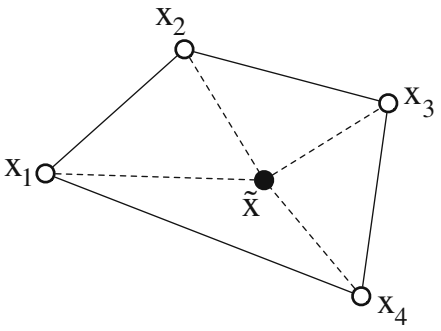


Fig. 2 Illustration of the weighted centroid method

value γ (usually $\gamma = 90\%$) is then used so that only the anchors with $\omega_i(t) \geq \gamma$ are considered.

Another possible approach is to rely on the average value of the RSSI.[2] Indeed, if we consider two anchors i and j, if $\mathbb{E}\left[\text{RSSI}_i\right] \geq \mathbb{E}\left[\text{RSSI}_j\right] \Rightarrow ||\mathbf{x}-\mathbf{x_i}|| \leq ||\mathbf{x}-\mathbf{x_j}||$, i.e., if the received signal received from anchor i is, on average, stronger than the signal at anchor j then the node is *most probably* closer from the anchor i than from the anchor j. Note that, as shown in the relation (1), this approach only holds on an *average* sense since the large scale fading creates fluctuations of the instantaneous RSSI around its average value. Using this approach, each weight is computed as

$$\omega_i(t) = \frac{\mathbb{E}\left[\text{RSSI}_i(t)\right] - \text{RSSI}_{\min}}{\text{RSSI}_{\max} - \text{RSSI}_{\min}}$$

where $\text{RSSI}_{\min}$ and $\text{RSSI}_{\max}$ are the minimum and the maximum values of the RSSI that can be acquired by the sensor node, respectively. These minimum and maximum values depend on the wireless transceiver considered.

2.2 Bounding Box

The bounding box method is a simple and computationally-efficient localization technique [33]. The main idea is to construct a bounding box for each anchor and

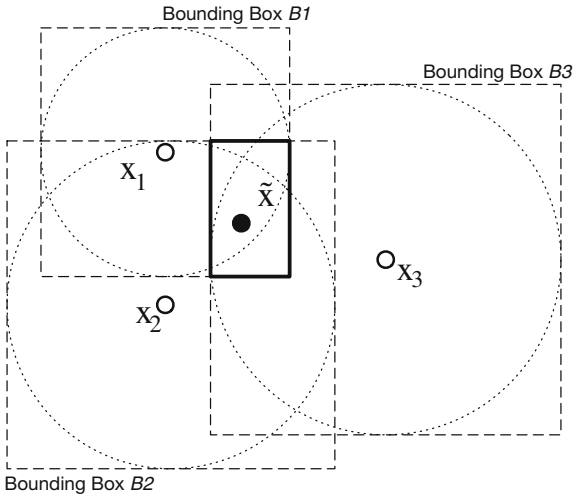


Fig. 3 Illustration of the bounding box algorithm

[2] The value of the RSSI is expressed in terms of received power and is usually between −91 and 0 dBm.

to compute the intersection of these bounding boxes. Figure 3 illustrates a typical situation in the case of three anchors. The length of side of the i-th bounding box is the (approximate) distance estimate to the target, i.e., the value of $\tilde{d}_i$ as computed using the techniques such as the ones that will be presented in Sect. 3. The position estimate is the bounding box (or the center of the bounding box) that is built as the intersection of all bounding boxes considered, i.e.,

$$\tilde{\mathbf{x}} \in \left\{ \bigcap_i \mathcal{B}_i \right\}$$

where $\mathcal{B}_i$ is a square box centered on $\mathbf{x}_i$ and with side length equal to $\tilde{d}_i$.

The geometry of the bounding box $\mathcal{B} = \left\{ \bigcap_i \mathcal{B}_i \right\}$ (and the corresponding interval of confidence for the localization) depends on the topology of the nodes. In dense sensor networks, the surface of $\mathcal{B}$ is limited and the position estimates typically converges to the true position. Although the accuracy of this technique is inherently limited, it is simple and fast to implement and to run on sensor nodes. Also, it can be used as an initial position in localization techniques based on a recursive estimation of the position [6, 29, 1]. Finally, though the bounding box technique makes use of distances estimates, it can be considered as range-free since it does not require an accurate estimate to work properly.

2.3 Point-in-Triangle

The Point-in-Triangle (PiT) algorithm [20] is an area-based, range-free localization scheme. The approach of PiT is presented in Fig. 4 and works as follows. In a first phase, the target broadcasts a beacon message. The set of all anchors receiving this message is built and all possible subsets of three anchors are built. The subset are

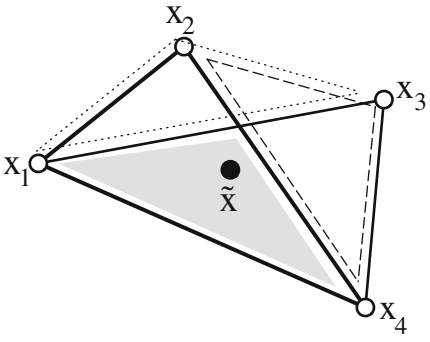


Fig. 4 Illustration of the PiT algorithm: the estimate belongs to the intersection of all possible triangles built on the anchors and containing the target

used to draw the corresponding triangles. For each triangle, a Point in Triangle (PiT) test, which is described below, is performed to determine if the target belongs to the area of this triangle. Finally, the target is located at the intersection of all triangles containing the target.

The geometrical interpretation of this approach is illustrated in Fig. 4 for a scenario with 4 anchors. These anchors define the triangles

$$\begin{aligned}\mathcal{T}_1 &= \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\} \ni \tilde{\mathbf{x}},\\ \mathcal{T}_2 &= \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4\} \ni \tilde{\mathbf{x}},\\ \mathcal{T}_3 &= \{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_4\} \ni \tilde{\mathbf{x}},\\ \mathcal{T}_4 &= \{\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\} \ni \tilde{\mathbf{x}},\end{aligned}$$

and the position estimate belongs to the triangle

$$\mathcal{T}_{\text{target}} = \bigcap_{i=1}^{4} \mathcal{T}_i.$$

The unknown location of the target is defined at the centroid of the intersection triangle $\mathcal{T}_{\text{target}}$. The main problem of the PiT scheme is to determine whether a given node belongs or not to a considered triangle. The PiT test is based on geometry: for a given triangle $\mathcal{T} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, a given mobile point $\tilde{\mathbf{x}}$ is *outside* the triangle if the gradient of the distance estimate to each vertex of the triangle is positive, i.e., the point $\tilde{\mathbf{x}}$ moves farther away from the points $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ *simultaneously*. Figure 5 provides a geometrical interpretation of the Point-in-Triangle test. On the figure to the right, since the point moves away *simultaneously* from all three vertices, it is likely to be outside of the triangle of interest. On the opposite, on the left part of the figure, $\tilde{\mathbf{x}}$ moves closer to $\mathbf{x}_1$ and $\mathbf{x}_2$ while moving away from $\mathbf{x}_3$. This point is therefore supposed to lie inside the triangle.

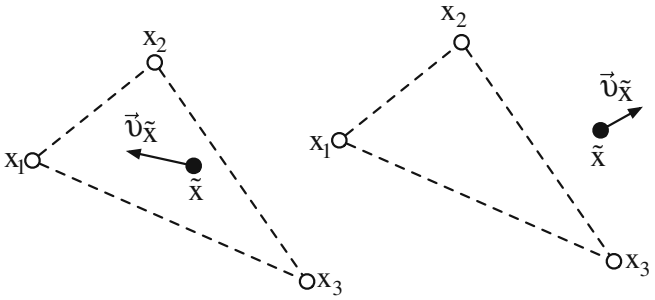


Fig. 5 Geometrical interpretation of the Point-in-Triangle test. On the figure to the right, the points away simultaneously from all summits is therefore outside of the triangle of interest. The vector $\bar{v}$ indicates the movement direction of the mobile target

3 Distance Estimation Techniques

3.1 Estimation of the Distance Based on the Received Power

In narrowband wireless communications systems, the distance to the emitter can be inferred at the receiver using the *shadowing model* [11]. This model makes the assumption of a rapid decrease of the signal strength with respect to the distance and adds a random variation in order to take into account the time- and distance-varying aspects of the signal, also called *large scale fading* [30]. The mathematical law, in logarithmic scale, relating the received power and the transmitted power is

$$P_{\mathrm{r}}(d)[\mathrm{dB}] = P_{\mathrm{r}}(d_0)[\mathrm{dB}] - 10\, n \log_{10}\left(\frac{d}{d_0}\right) + F \tag{1}$$

where $P_{\mathrm{r}}(d_0)$ is the received power at a reference distance d_0 and F is the logarithmic version (in dB) of a random variable X which takes into account the influence of the environment on signal propagation, i.e., $F = \log_{10} X$. Extensive experimental analysis [31] shows that X has a lognormal distribution, i.e., $F \sim \mathcal{N}(0, \sigma)$ (units: [dB]). The path-loss exponent n (also referred to as *decay factor*) can be fixed by means of experimentation: typical values for the path-loss exponent range from 2 (outdoor and or indoor with strong line-of-sight) to 4 (indoor environment with strong signal attenuation). The reference distance d_0 of the measurements is usually set at $d_0 = 1$ m.

Most wireless sensor networks do present a limited computation power and some of them do not implement the logarithmic and exponential functions. When the distance between two sensors is limited (i.e., a few meters away), a Taylor series development of the logarithm in (1) can be used. Since $\ln(x) \approx (x-1) + \Theta(x^2)$ for a small value of x, the expression (1) can be approximated as

$$\begin{aligned} P_{\mathrm{r}}(d)[\mathrm{dB}] &= P_{\mathrm{r}}(d_0)[\mathrm{dB}] - 10\, n \log_{10}(d) + F \\ &= P_{\mathrm{r}}(d_0)[\mathrm{dB}] - 10\, n\, \frac{\ln(d)}{\ln(10)} + F \\ &\approx P_{\mathrm{r}}(d_0)[\mathrm{dB}] - 10\, n\, \frac{d-1}{\ln(10)} + F \end{aligned} \tag{2}$$

Note that a linear approximation can also be used when the distance between the nodes is larger but the values of the observed distances are of the same order of magnitude (local linearity). More generally, the relation (2) shows that $\mathbb{E}[P_{\mathrm{r}}] = K_1 - K_2\, d$, where K_1 and K_2 are constants that can be obtained by means of proper experiment and depends on the specific propagation environment. This relation shows that the distance between two nodes in a short range of transmission can be obtained as a linear function of the average power of the signal observed at the receiver node. In practice, this relation is used to estimate the distance of the target

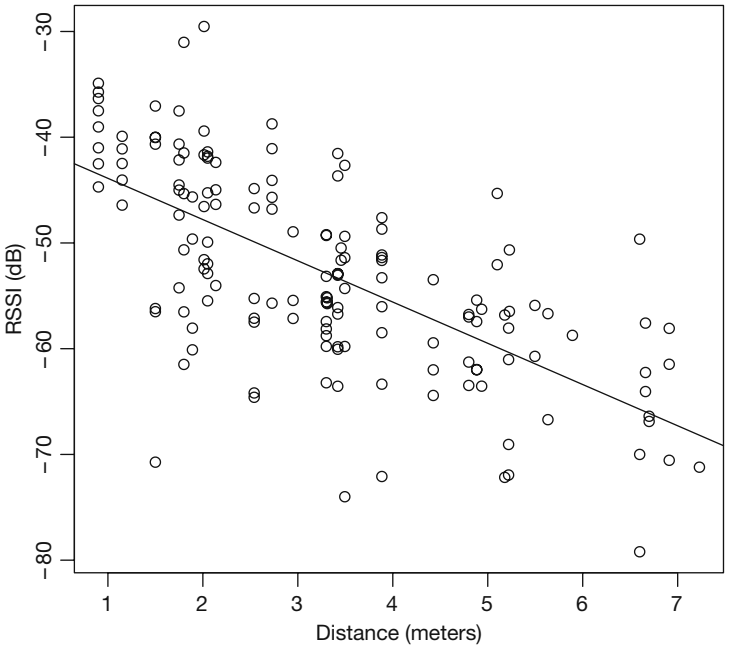


Fig. 6 Power at the receiver as a function of the distance

on the basis of power-related measures, like the Radio Signal Strength Indicator (RSSI).

The RSSI is an indicator of the Signal strength at the instant of the reception of a message and can be linked to the average reception power of the packet. Figure 6 illustrates the linearity of the relation between the distance and the RSSI for a set of experimental measures collected in a factory hall and over small distances. The received power was estimated by using the Reception Signal Strength Indicator (RSSI). For instance, a CC2420 radio was used in Fig. 6 and the RSSI value was obtained as a 8-bit numerical value stored in a register of the radio hardware. The RSSI value is averaged on 8 transmission symbols ($\sim 128\,\mu$s) and the approximate power of a received packet can be calculated from the calibration formula:

$$P = \mathrm{RSSI}_{\mathrm{value}} + \mathrm{RSSI}_{\mathrm{offset}}\ [\mathrm{dBm}]$$

where $\mathrm{RSSI}_{\mathrm{offset}}$ is an correction that is empirically estimated and is around -45 dBm for the CC2420 radio interface.

3.2 Angle-of-Arrival

The angle-of-arrival (AoA) (sometimes referred to as Direction-of-Arrival – DoA) technique performs an angulation at several anchor nodes in order to estimate the position of the target. The determination of angles can be done by means of an

antenna array capable of adaptative beamforming [36, 12]. More precisely, only the nodes in charge of the sensing of the AoA must be equipped of beamforming antenna arrays. The basic principle behind the angle-of-arrival technique is presented in Fig. 7 in a scenario with two anchor nodes. A pilot signal is sent by the target and received by at least two anchors. The angle of this pilot signal is computed at each anchor and defines a region of space where the pilot signal was emitted. If we denote by α_i the angle estimated by the i-th anchor, one has:

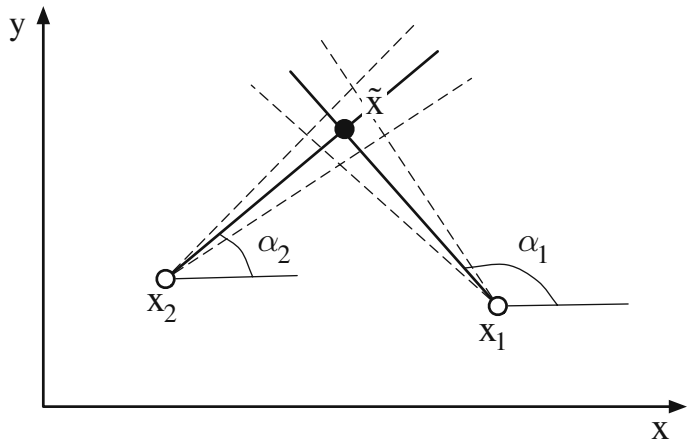


Fig. 7 Illustration of the angulation (angle-of-arrival) technique

$$\tan \alpha_i = \frac{y_i - \tilde{y}}{x_i - \tilde{x}}$$

where $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y})$ is the position estimate and $\mathbf{x}_i = (x_i, y_i)$ is the (known) position of the i-th anchor. If more than two anchors are used, a least square fit can be applied to derive the target's position estimate.

The error potential of this technique is influenced by the resolution of the antenna arrays and the order of magnitude of the distance separating the target and the anchors. Indeed, if the target is found to be at an angle α, the performed measurement implicitly contains an error estimate. If we denote this value by ε, the target will in fact lie in a region between the angles $\alpha - \varepsilon$ and $\alpha + \varepsilon$. Moreover, if the target is found to be at a distance $\tilde{d}$ of the anchor node, the error on the position (in meters) is proportional to $\varepsilon.\tilde{d} = \Theta(\tilde{d})$. Therefore, the localization will be more accurate if the target is close to the anchor nodes and will degrade linearly with the distance.

Another issue for the AoA technique is the existence of multipath propagation, which is typically found in indoor locations. These multiple reflections on the walls and material present in the environment create a multitude of subsequent echos that are captured by the receiver at the anchor node. These signals come from several, different angles and cause an indetermination on the actual angle of the target node. A possible solution to balance this indetermination is to use multiple anchor nodes, depending on the severity of the multipath propagation.

3.3 Time-of-Flight

In the time-of-flight (ToF) technique, the range measurement is performed on the basis of the amount of time needed between the emitting of a given signal and its reception by a receiver. The ToF can involve fast wireless signals or slower forms signals such as a sound signal (audible or not). For instance, in Fig. 8, a radio signal is sent to start a counting clock at the receiver. The traveling time of the wireless signal is supposed to be non-significant and can be neglected. After a fixed interval of time, the emitter sends an audio signal. This audio signal travels much more slowly than the wireless signal and reaches the receivers after a measurable travel time. At that moment, the counting clock is stopped and the distance between the emitter and the receiver is derived.

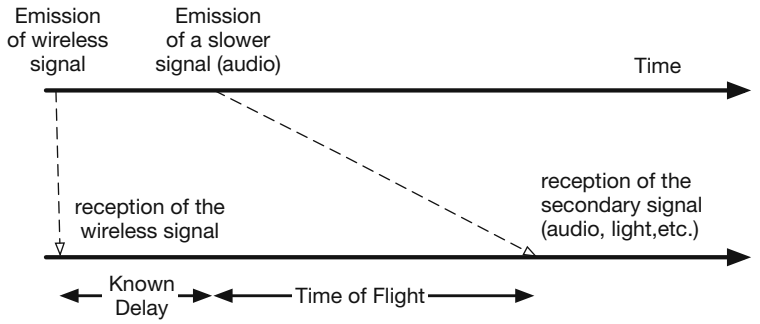


Fig. 8 Computation of the time-of-flight by means of two different signals having very different propagation times

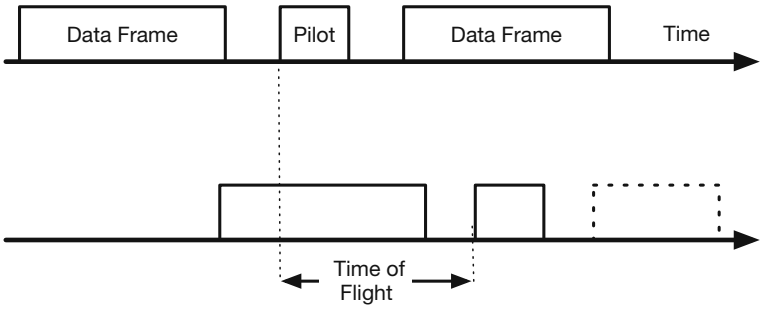


Fig. 9 Illustration of the code phase ranging

Another common technique, used in GSM systems, is to send a periodical pilot packet. Since the GSM system relies on a Time division multiple access[3] (TDMA)

[3] A wireless channel multiplexing technique in which each terminal is assigned a different time slot to communicate. Each terminal transmits in a chronological succession, one after the other, by using his assigned time slot.

scheme, the clocks of the base stations and the terminals are synchronized and the data slots are of a defined and well-known length. This technique is referred to as *pulse ranging* and is illustrated in Fig. 9. All pilots are sent at the same time and the time differences (and, therefore, the ranges) are computed. For instance, in Sect. 4.2, the hyperbolic lateration technique is presented and it exploits the time differences estimated from the ToF to infer the position of the user terminals.

4 Static Localization Techniques – Range-based

4.1 Circular Lateration and Multilateration

The circular alteration technique make use of the distance estimation between the anchors and the target. Figure 10 presents a typical situation for 3 anchors and a single target node. Let N be the number of fixed nodes (called *anchors*) whose positions are known a priori. The goal of the localization procedure is to find the position of the mobile nodes (referred to as *targets*) and labeled : $i \in \{1 \ldots N'\}$.

We note the real position of the node i in a p-dimensional Euclidean space by a vector in $\mathbb{R}^p$: $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$. Let $\mathbf{X}_{\text{anchors}} = [\mathbf{x}_i]\,,\ 1 \leq i \leq N$, be a matrix of size $p \times N$. The aim of the method is to find $\mathbf{X}_{\text{targets}} = \left[\mathbf{x}_j\right],\ 1 \leq j \leq N'$. Let $\tilde{\mathbf{x}}_j$ be our estimation of $\mathbf{x}_j$. If we denote by

$$\tilde{d}_{ij} = ||\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j|| = \sqrt{\sum_{k=1}^{p} (\tilde{x}_{ik} - \tilde{x}_{jk})^2}$$

the estimation Euclidean distance between two nodes i and j, we can now formulate the problem as the minimization of

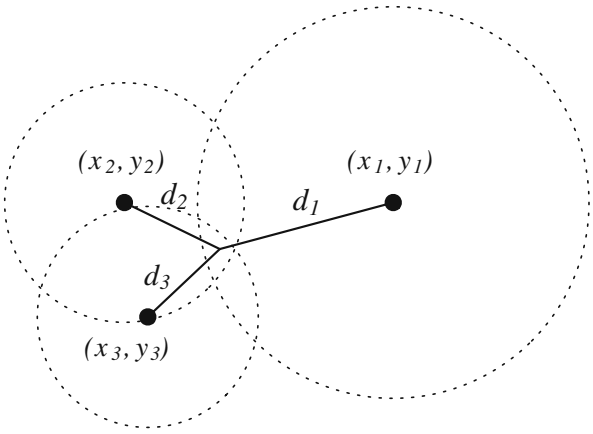


Fig. 10 The circular lateration technique

$$J(\tilde{\mathbf{x}}) = \sum_{\substack{1 \le i \le N \\ 1 \le j \le N'}} \left(\tilde{d}_{ij} - ||\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j||\right)^2$$

where $\tilde{d}_{ij}$ is an estimation of the distance d_{ij}. It can be shown that this minimization problem can be expressed under the form of a linear system. In order to derive this form, the estimation of the distance has to be expressed under the form of a Taylor development.

Let $\mathcal{D}$ denote an interval, $a \in \mathcal{D}$ a point of this interval, and $f : \mathcal{D} \to \mathbb{R}$ a function with derivatives up to $f^{(n+1)}$. The Taylor series expansion for $x \in \mathcal{D}$ of the function f is the following series around the point (x_0, y_0) is

$$\begin{aligned} f(x_0 + \Delta x, y_0 + \Delta y) = & f(x_0, y_0) + \frac{\partial f(x_0, y_0)}{\partial x}\Delta x + \frac{\partial f(x_0, y_0)}{\partial y}\Delta y \\ & + \Theta_2(x_0, y_0, \Delta x, \Delta y) \end{aligned}$$

where $\Theta_2(x, y, \Delta x, \Delta y)$ is the order 2 remainder term. If we denote a first estimate of the 2D-position $\tilde{\mathbf{x}}_\mathbf{0} = (\tilde{x}_0, \tilde{y}_0)$, the goal of the multilateration technique [33] is to compute a correction of the estimated position that will be applied to $\tilde{\mathbf{x}}_\mathbf{0}$. This correction value is noted $\Delta\mathbf{x} = [\Delta x, \Delta y]$. Note that the multilateration technique is an iterative process and the value of $\tilde{\mathbf{x}}_\mathbf{0}$ can be initialized by means of a range-free technique presented in Sect. 2.

Let us denote by d_i the range between the target and the i-th anchor and compute a first-order decomposition of the distance d_i

$$\begin{aligned} d_i(x, y) &= \sqrt{(x_i - x)^2 + (y_i - y)^2} \\ &= d_i(\tilde{x} + \Delta x, \tilde{y} + \Delta y) \\ &\approx d_i(\tilde{x}, \tilde{y}) + \frac{\partial d_i}{\partial \tilde{x}}\Delta x + \frac{\partial d_i}{\partial \tilde{y}}\Delta y \end{aligned} \tag{3}$$

where

$$\begin{aligned} \frac{\partial d_i}{\partial \tilde{x}} &= \frac{\tilde{x} - x_i}{\tilde{d}_i} = a_{i1} \\ \frac{\partial d_i}{\partial \tilde{y}} &= \frac{\tilde{y} - y_i}{\tilde{d}_i} = a_{i2} \\ \tilde{d}_i &= \sqrt{(x_i - \tilde{x})^2 + (y_i - \tilde{y})^2} \end{aligned}$$

The expression (3) can therefore be equivalently rewritten as

$$a_{i1}\Delta x + a_{i2}\Delta y = d_i(\tilde{x} + \Delta x, \tilde{y} + \Delta y) - d_i(\tilde{x}, \tilde{y}) \tag{4}$$

which yields to express the problem as the following matrix equation

$$\mathbf{A}\ \Delta\mathbf{x} = \mathbf{b}$$

with

$$\mathbf{b} = \begin{bmatrix} d_1(\tilde{x}+\Delta x, \tilde{y}+\Delta y) - d_1(\tilde{x}, \tilde{y}) \\ d_2(\tilde{x}+\Delta x, \tilde{y}+\Delta y) - d_2(\tilde{x}, \tilde{y}) \\ \vdots \\ d_N(\tilde{x}+\Delta x, \tilde{y}+\Delta y) - d_N(\tilde{x}, \tilde{y}) \end{bmatrix},\ \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \vdots & \\ a_{N1} & a_{N2} \end{bmatrix},\ \Delta\mathbf{x} = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{5}$$

Note that the quantity $d_i(\tilde{x}, \tilde{y})$ is the *known* distance between the estimated position and the position of the i-th anchor node. Also, the quantity $d_i(\tilde{x} + \Delta x, \tilde{y} + \Delta y)$ contains the *observed* distance between the i-th base anchor and the target. It is therefore also numerically known.

A close look at (5) shows that the system is overdetermined, i.e., there are more equations expressed through $\mathbf{b}$ and $\mathbf{A}$ than unknown values in $\mathbf{x}$. A close solution (in a least square sense) can be obtained by minimizing the residual quantity

$$R = (\mathbf{A}\ \Delta\tilde{\mathbf{x}} - \mathbf{b})^T(\mathbf{A}\ \Delta\tilde{\mathbf{x}} - \mathbf{b})$$

which can be achieved by computing the value of R where $\nabla R = 0$. This gives

$$-2\mathbf{A}^T\mathbf{b} + \mathbf{2A}^T\mathbf{A}\ \Delta\tilde{\mathbf{x}} = 0 \tag{6}$$

and

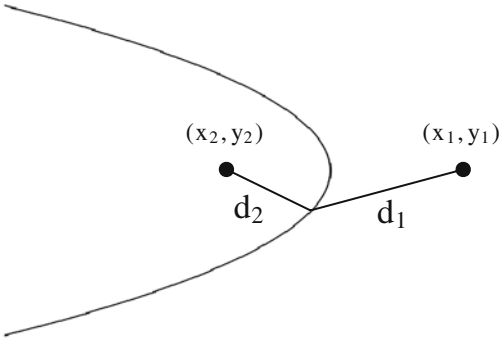
$$\Delta\tilde{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}(\mathbf{A}\ \Delta\mathbf{x} - \mathbf{b})^T(\mathbf{A}\ \Delta\mathbf{x} - \mathbf{b}) = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$$

It follows that the updated position estimate is

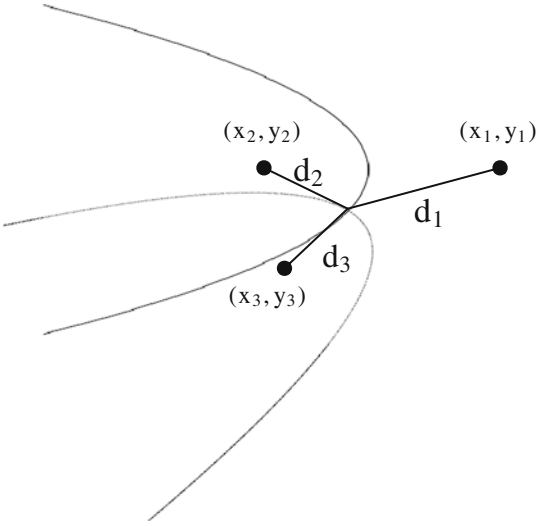
$$\begin{aligned} \tilde{\mathbf{x}} &= \tilde{\mathbf{x}}_\mathbf{0} + \Delta\tilde{\mathbf{x}} \\ &= \tilde{\mathbf{x}}_\mathbf{0} + (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} \end{aligned}$$

Note that the accuracy of the estimate can be assessed by computing the residual value between the estimated and the real locations. This value is expressed as

$$\begin{aligned} \mathbf{W} &= ||\mathbf{x} - \tilde{\mathbf{x}}||_2^2 - \mathbf{d} \\ &= \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i - \tilde{x})^2 + (y_i - \tilde{y})^2} - d_i \end{aligned}$$



(a) The locus of the positions that correspond to a given difference of distances is a hyperbola



(b) The hyperbolic lateration technique: the target lies at the intersection of all possible hyperbolae created from the range differences.

Fig. 11 Geometrical interpretation of the hyperlateration positioning technique

where $\mathbf{d} = [d_1, d_2, \cdots, d_N]^T$ is the vector containing the observed distance, i.e., the distance estimated by the anchors nodes by means of the radio signal strength and a proper signal-to-distance model.

4.2 Hyperbolic Lateration

The circular lateration technique relies on the estimation of the absolute distance between the target and a set of anchors. The hyperbolic lateration algorithm has

the same rationale though it relies on the range *differences*. Figure 11(a) presents a typical situation where the difference of the range estimates of two anchors $d_{ij} = d_i - d_j$ is used to localize a target. The *locus* of the all positions P whose difference of distances from the two fixed anchors (x_i, y_i) and (x_j, y_j) is the constant value $d_{ij} = d_i - d_j$ describes a *hyperbola*. The same locus can be created for any number of additional anchors. As presented in Fig. 11(b), the unknown position of the target lies at the intersection of all derived hyperbola. As in the case of circular lateration presented in Sect. 4.1, all differences of range can be expressed as a system of nonlinear equations:

$$\begin{aligned} \forall i,\, j\ (i \neq j) : d_{ij} &= d_i - d_j \\ &= \sqrt{(x_i - x)^2 + (y_i - y)^2} - \sqrt{(x_j - x)^2 + (y_j - y)^2} \end{aligned} \quad (7)$$

where d_{ij} is the difference of range between the target and the i-th (j-th) anchor. The method for solving the system of nonlinear equations is the same as for circular lateration. More precisely, a Taylor series expansion is derived as

$$\frac{\partial d_{ij}}{\partial \tilde{x}} = \frac{\tilde{x} - x_i}{\tilde{r}_i} - \frac{\tilde{x} - x_j}{\tilde{r}_j} \triangleq a_{ij,1}$$

$$\frac{\partial d_{ij}}{\partial \tilde{y}} = \frac{\tilde{y} - y_i}{\tilde{r}_i} - \frac{\tilde{y} - y_j}{\tilde{r}_j} \triangleq a_{ij,2}$$

$$\tilde{d}_i = \sqrt{(x_i - \tilde{x})^2 + (y_i - \tilde{y})^2}$$

$$\tilde{d}_j = \sqrt{(x_j - \tilde{x})^2 + (y_j - \tilde{y})^2}$$

and it can written under the same form used in (4). The analogy with the least-square error minimization and the position estimation detailed in (5) and (6) is straightforward and is not detailed here.

5 Dynamic Localization and Tracking

Several filtering techniques exist and were successfully applied in the domain of wireless sensor networks [13, 26, 37]. In this section, we detail the steps of a dynamic tracking algorithm based on the Kalman-class of filters.

The Kalman filter [8] is a recursive filter that can be used to estimate the dynamic state of a system at a given time t_k by using noisy measurements issued at time $t_{k-1}, t_{k-2}, t_{k-3}, \ldots, t_0$. Kalman filtering applications include radar localization, road navigation when the GPS is temporarily not available, and control theory (estimation and forecasting). It proves useful to predict the state of the system at a future time. The underlying dynamic system model is presented in Figs. 12 and 13. It assumes that the state of the system $\mathbf{x}_k$ at time t_k can be derived from the state at t_{k-1} by using the following model:

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_{k-1} + \mathbf{w}_{k-1} \tag{8}$$

where

- $\mathbf{F}_k$ is transfer function of the system that receives as input the previous state $\mathbf{x}_{k-1}$
- $\mathbf{B}_k$ is the control-input model that is applied to a control vector $\mathbf{u}_k$ (open-loop control [4])
- $\mathbf{w}_k$ is an additive process variability which is supposed to be a zero-mean Gaussian distributed random value, i.e., $\mathbf{w}_k \sim \mathcal{N}(0, \boldsymbol{\sigma}_k^{\mathrm{w}})$

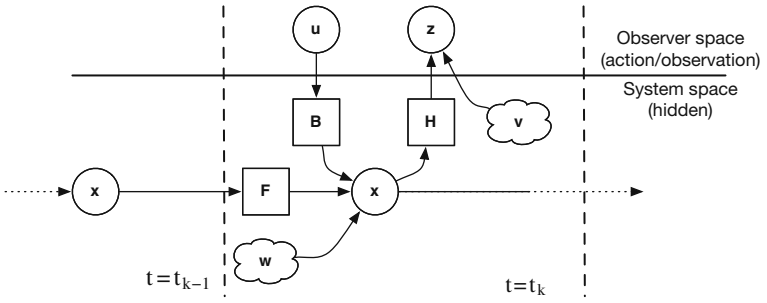


Fig. 12 Underlying model for the design of the customizable Kalman filter. The variables are noted as *ovals*, the transition functions as *squares*, and the noise processes as a *cloud*

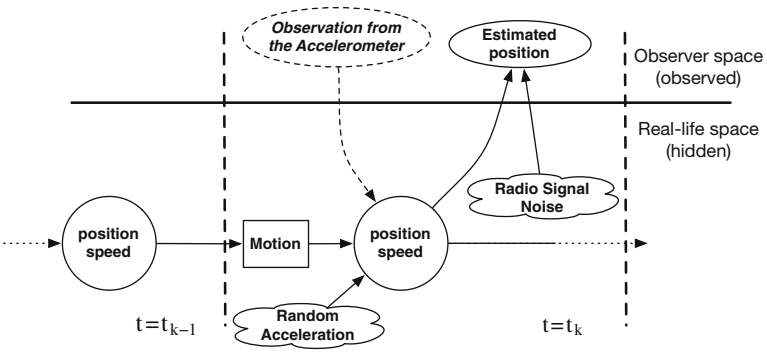


Fig. 13 Physical interpretation for the Kalman filter presented in Fig. 12. Note that the samples collected by the accelerometer are only used in the variant proposed in Sect. 7. It is therefore noted as *dashed*

The system state is supposed to be observed by means of a noisy measurement of the real (hidden) system state. According to Kalman's model in Fig. 12, this measure can be expressed as: $\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k$, where $\mathbf{H}_k$ is the observation model, i.e., the transfer function of the measurement device and $\mathbf{v}_k$ is the observation noise which is supposed to be a zero-mean Gaussian process, i.e., $\mathbf{v}_k \sim \mathcal{N}(0, \boldsymbol{\sigma}_k^{\mathrm{v}})$. Starting from there, several variants will be developed in this chapter.

The Kalman filtering process is *iterative*, i.e., the estimation of the current state of the system only depends only the previous state. In that it is different from batch estimation techniques which make use of *all* observations of the past. The state of the filter is represented by two key variables: $\tilde{\mathbf{x}}_{k|k}$ and $\mathbf{P}_{k|k}$ where the notation $\tilde{\mathbf{x}}_{i|j}$ represents the estimate of $\mathbf{x}$ at $t = t_i$ given the previous observations z_i $(0 \leq i \leq j)$:

$$\begin{aligned}\tilde{\mathbf{x}}_{i|j} &\triangleq \tilde{\mathbf{x}}_{i|\{z_0,z_1,\ldots,z_j\}}\\ \mathbf{P}_{i|j} &\triangleq \mathbf{P}_{i|\{z_0,z_1,\ldots,z_j\}}\end{aligned}$$

The matrix $\mathbf{P}$ is referred to as *the error covariance matrix* since it used to represent a measure of the estimated accuracy of the state estimate. More precisely, it is defined as

$$\begin{aligned}\mathbf{P}_{k|k-1} &\triangleq \mathbb{E}\left[\left(\mathbf{x}_k - \tilde{\mathbf{x}}_{k|k-1}\right)\left(\mathbf{x}_k - \tilde{\mathbf{x}}_{k|k-1}\right)^T\right]\\ &= \operatorname{cov}\left(\mathbf{x}_k - \tilde{\mathbf{x}}_{k|k-1}\right)\end{aligned}$$

Note that $\mathbf{P}_{k|k-1}$ is an a priori estimate of the matrix $\mathbf{P}$, which means that it is our best estimate prior to assimilating the measurement at t_k. It will be shown later how $\mathbf{P}_{k|k}$ can be derived from $\mathbf{P}_{k|k-1}$.

5.1 Kalman Filtering Loop

The Kalman filtering process is made of two inductive phases: *predict* and *update*. The predict phase uses the state estimate from the previous timestep to produce an estimate of the state at the current timestep. In the update phase, measurement information at the current timestep is used to refine this prediction to arrive at a new, (hopefully) more accurate state estimate, again for the current timestep. The process steps can be expressed as the following iterations:

I. Time Update

1. Project a state ahead

$$\tilde{\mathbf{x}}_{k|k-1} = \mathbf{F}_k\tilde{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u}_{k-1}$$

2. Project the error covariance ahead

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \boldsymbol{\sigma}_k^w$$

II. Measurement Update

3. Compute the Kalman gain

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \boldsymbol{\sigma}_k^v\right)^{-1}$$

4. Update the estimate with measurement $\mathbf{z}_k$

$$\tilde{\mathbf{x}}_{k|k} = \tilde{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \left(\mathbf{z}_k - \mathbf{H}_k \tilde{\mathbf{x}}_{k|k-1}\right)$$

5. Update the error covariance

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\, \mathbf{P}_{k|k-1}$$

5.2 Filtering Process when the Speed and the Position are Unknown

The problem of localization in the context of wireless sensor networks can be applied to localizing a mobile target (human operator or vehicle) that moves at a constant speed and whose acceleration is unknown. In such a situation, the second law of Newton yields

$$\mathbf{x}(t_k) = \mathbf{x}(t_{k-1}) + \mathbf{s}\ \Delta t + \mathbf{a}\frac{\Delta t^2}{2}$$

where $x(t_k)$ and $x(t_{k-1})$ are the current and previous positions, respectively (units: [m]); s is the speed (units: [ms^{-1}]), a is the acceleration (units: [ms^{-2}]), and Δt is the difference of time between $t = t_{k-1}$ and $t = t_k$, i.e., $\Delta t \triangleq t_k - t_{k-1}$ (units: [s]). This model can be written under the form equivalent to (8), where the unknown term $\mathbf{x}_k$ is the position *and* the velocity of the target, i.e., $\mathbf{x}_k = [x\ y\ s_x\ s_y]^T$. Moreover, on this model, the acceleration of the target is interpreted as the random variation $\mathbf{w}_k$ applied to $\mathbf{x}_k$. The relation (8) can be written as

$$\underbrace{\begin{bmatrix} x \\ y \\ s_x \\ s_y \end{bmatrix}_k}_{\mathbf{x}_k} = \underbrace{\begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{F}_k} \underbrace{\begin{bmatrix} x \\ y \\ s_x \\ s_y \end{bmatrix}_{k-1}}_{\mathbf{x}_{k-1}} + \underbrace{\begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix}_k}_{\mathbf{w}_k} \tag{9}$$

Note that, since no friction or external forces are applied to the system, we assume $\mathbf{B}_k = 0$ and $\mathbf{u}_k = 0$. The value of $\mathbf{a}_k$ is difficult to estimate since it represents the average acceleration experienced by the target node. Note that, under the Kalman filter hypotheses, it has to be a zero-mean, Gaussian-distributed rv., i.e., $a_{x,y} \sim \mathcal{N}(0, \sigma_a)$ and

$$\begin{aligned}\boldsymbol{\sigma}_k^w &= \mathrm{cov}(\mathbf{w}_k) = \mathbb{E}\left[\mathbf{w}_k \mathbf{w}_k^T\right] \\ &= \sigma_a^2 \begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \begin{bmatrix} \Delta t^2/2 & 0 & \Delta t & 0 \\ 0 & \Delta t^2/2 & 0 & \Delta t \end{bmatrix}\end{aligned} \tag{10}$$

A localization algorithm, such the lateration techniques presented in Sect. 4, is used to estimate the position of the user at each time $t = t_k$. This measurement can be expressed by the following relation:

$$\mathbf{z}_k = \underbrace{\begin{bmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \end{bmatrix}}_{\mathbf{H}_k} \mathbf{x}_k + \mathbf{v}_k$$

where $\mathbf{v}_k \sim \mathcal{N}(0, \boldsymbol{\sigma}_k^{\mathrm{v}})$ and $\boldsymbol{\sigma}_k^{\mathrm{v}}$ is the variance of the localization process and is often proportional to the power of the fading, as shown in (1). In most cases, the localization process gives n estimates of this position, each of them subject to an incertitude. As then, the vector $\boldsymbol{\sigma}_k^{\mathrm{v}}$ can be approximated by covariance of these n positions:

$$\boldsymbol{\sigma}_k^{\mathrm{v}} = \begin{bmatrix} \frac{n-1}{n}\sigma_x^{(\mathrm{est})} & 0 \\ 0 & \frac{n-1}{n}\sigma_y^{(\mathrm{est})} \end{bmatrix}$$

where $\sigma_x^{(\mathrm{est})}$ and $\sigma_y^{(\mathrm{est})}$ are the variance of the n estimates of the positions under the axes x and y, respectively. Note that, since the multilateration does not evaluate the speed, some coefficient in $\mathbf{H}_k$ are zeroed. Finally, the initial conditions have to be considered. If we know the initial position of the user, we can set

$$\begin{aligned}\mathbf{x}_{0|0} &= [x_0,\ y_0,\ 0,\ 0]^T \\ \mathbf{P}_{0|0} &= [0]\end{aligned}$$

On the other hand, if the position is not perfectly known, $\mathbf{x}_{0|0}$ can be initialized using the best approximate value for (x_0, y_0) and the covariance matrix can be filled with a large diagonal value ε, i.e.:

$$\mathbf{P}_{0|0} = \mathbf{I}.\varepsilon = \begin{bmatrix} \varepsilon\,0\,0\,0 \\ 0\,\varepsilon\,0\,0 \\ 0\,0\,\varepsilon\,0 \\ 0\,0\,0\,\varepsilon \end{bmatrix}$$

5.3 Filtering Process for an Unknown Position and an Approximated Speed

Another variant of the Kalman presented in the last section consists in removing the speed from the set of unknown state variables and, instead, to infer it from the previous positions and the update interval. Referring back to the relation (8) and the Fig. 12, the speed can be seen as the input vector $\mathbf{u}$ which is applied to the system. Also, note that in the relation (8), the value of $\mathbf{u}$ is considered at the *previous time*, i.e., $t = t_{k-1}$. As then, it is convenient to approximate this quantity as $\tilde{\mathbf{u}}_{k-1} = (\tilde{\mathbf{x}}_{k-1} - \tilde{\mathbf{x}}_{k-2})/(t_{k-1} - t_{k-2})$. This time, only the position of the target is supposed unknown and the relation (9) becomes:

$$\underbrace{\begin{bmatrix} x \\ y \end{bmatrix}_k}_{\mathbf{x}_k} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}_k} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}_{k-1}}_{\mathbf{x}_{k-1}} + \underbrace{\begin{bmatrix} \Delta t & 0 \\ 0 & \Delta t \end{bmatrix}}_{\mathbf{B}_k} \underbrace{\begin{bmatrix} \tilde{x}_{k-1} - \tilde{x}_{k-2} \\ \tilde{y}_{k-1} - \tilde{y}_{k-2} \end{bmatrix}_k}_{\mathbf{u}_{k-1}} + \underbrace{\begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix}_k}_{\mathbf{w}_k}$$

The relation (10) can be written as:

$$\begin{aligned} \boldsymbol{\sigma}_k^w &= \operatorname{cov}(\mathbf{w}_k) = \mathbb{E}\left[\mathbf{w}_k \mathbf{w}_k^T\right] \\ &= \sigma_a^2 \begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \end{bmatrix} \begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \end{bmatrix} \end{aligned}$$

The measurement of the system has the form:

$$\mathbf{z}_k = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{H}_k} \mathbf{x}_k + \mathbf{v}_k$$

The vector $\boldsymbol{\sigma}_k^{\mathrm{v}}$ can be approximated as in the previous section:

$$\boldsymbol{\sigma}_k^{\mathrm{v}} = \begin{bmatrix} \frac{n-1}{n}\, \sigma_x^{(\mathrm{est})} & 0 \\ 0 & \frac{n-1}{n}\, \sigma_y^{(\mathrm{est})} \end{bmatrix}$$

6 Accuracy and Precision

In the context of localization, the most important and user-oriented parameters are the *accuracy* and the *precision* of the considered method. Precision refers to the closeness of the different position estimates around their means (i.e., the position estimate). On the other hand, accuracy refers to the closeness of the position esti-

mates to the real position of the target. In the field of estimation theory, the notions of accuracy and precision are best known under the names of *bias* and *variance* of the estimator, respectively.

Figure 14 illustrates geometrically the difference between these two notions. More precisely, on the left part of the figure, the position estimator has a great *precision* value since all estimates are close to their mean value. However, the estimated position (i.e., the mean value) is far from the real position of the target. On the opposite, the right part of Fig. 14 presents a higher degree of *accuracy* since the average estimated position is close to the real position, though some estimates are located far from the average value.

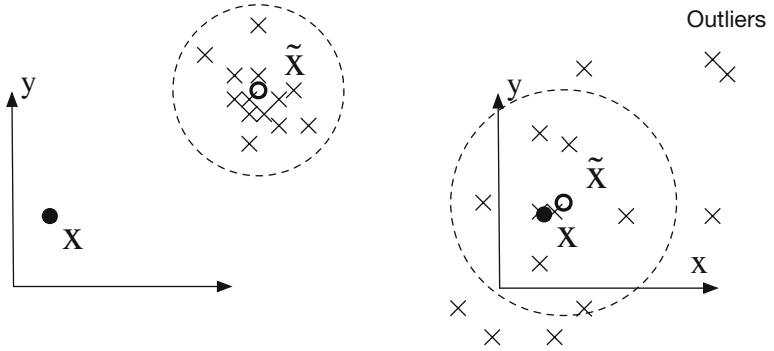


Fig. 14 Illustration of the notions of *accuracy* and *precision*. On the *left figure*, the position estimates have a great precision since they are close to their average value. On the *right figure*, the degree of accuracy is high since the estimated position matches with the real position

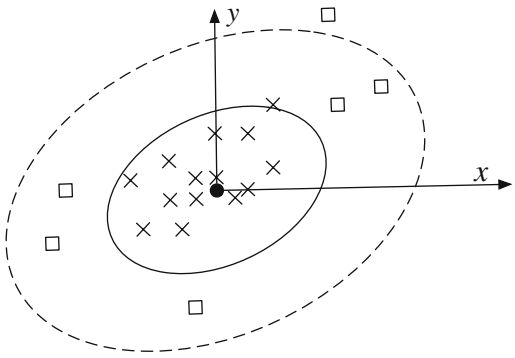


Fig. 15 Accuracy in the case of a two-dimensional localization process

Note that the estimates that diverge too much from the mean value are referred to as *outliers* and can be easily detected. For instance, two decision are rules are often used:

1. The position estimate and its corresponding distance to the target exceeds the maximum radio range. This estimate can be discarded since it would not have been possible for the anchors to overhear the beacon emitted by the target.
2. The variance of the position estimates is computed and noted as σ. Any node i where $\tilde{d}_i \geq \gamma$, where γ is a given threshold, are to be rejected.

It can be shown that, in the case of a random Gaussian error, $\gamma = \sigma$ and $\gamma = 2\sigma$ yield 68 and 95% of accuracy, respectively. In Fig. 15, one can see a typical situation where some outliers (noted as squares) are present in the position estimates (noted as crosses). On this figure, the variances along the x-axis and the y-axis were computed and drawn as a dashed ellipsoid.

Finally, it is possible to compute a *residue value* of the position estimates. This value is be defined as the average difference between the given, known distances and the distances to the location estimate:

$$\begin{aligned} W &= \mathbb{E}\left[\, ||\mathbf{x_a} - \tilde{\mathbf{x}}||_2 - \mathbf{d_a} \,\right] \\ &= \frac{1}{N} \sum_{i=1}^{N} \sqrt{(x_i - \tilde{x})^2 + (y_i - \tilde{y})^2} - d_i \end{aligned}$$

where $\mathbf{x_a}$ refers to the postion of an anchor node and $\mathbf{d}_a$ is the distance to the location estimate $\tilde{\mathbf{x}}$. A large residue value will signal an inconsistent set of equations in the lateration technique and leads to a rejection when the value of R is larger than the radio range.

7 Advanced Localization Techniques – Data Fusion by means of Machine Learning

One interesting feature of WSN is the presence of multiple sensors at each node. Therefore, it is possible to take advantage of the combination of heterogenous observations obtained from these very different sensors. For instance, most sensor nodes embed a 2-axes accelerometer. In a scenario where the target node is worn by a human personal, it is therefore possible to determine the motion state of the user (i.e., motion or stillness) and use this information in the tracking filter. More classical techniques, such as the fusion of estimators or the supervised or non supervised classification can also be used to drastically improve the final estimate of the target node.

The remainder of this section is organized as follows. In Sect. 7.1, the considered scenario is presented and some empirical considerations are introduced. Next, in Sect. 7.2, we present how the data collected at the accelerometer can be suitably and automatically classified. In Sect. 7.3 the Kalman filter introduced in Sect. 5 is suitably modified in order to take advantage of the additional motion state information. Sect. 7.4 motivates the joint use of the Kalman filter together with different data

fusion schemes. Finally, in Sect. 7.5 a fusion of localization estimators technique is presented.

7.1 Introduction

In this section, we consider a scenario where a WSN is deployed to provide localization information of the maintenance personal operating in a factory. First, based on empirical observations, it may be assumed that the natural movement of the operators is made at constant speed and is, on average, equal to $|s| = 1\,\mathrm{ms}^{-1}$. Second, the velocity of the walking from a workstation to another is subject to a normal perturbation that is given to the user's acceleration between two stillness moments. Furthermore, it can observed that humans start (or stop) walking in about one second. Since the average natural walking speed is around $|s| = 1\,\mathrm{ms}^{-1}$, we can say, with 95% of accuracy, that the variance of the acceleration is $2\sigma_a^w = 1\,\mathrm{ms}^{-2}$ under all directions which is the most pessimistic case. Therefore, the relation (10) can be written as

$$\begin{aligned}\boldsymbol{\sigma}_k^w &= \mathrm{cov}(\mathbf{w}_k) = \mathbb{E}\left[\mathbf{w}_k \mathbf{w}_k^T\right] \\ &= \frac{1}{2}\begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix}\begin{bmatrix} \Delta t^2/2 & 0 & \Delta t & 0 \\ 0 & \Delta t^2/2 & 0 & \Delta t \end{bmatrix}\end{aligned}$$

7.2 Observation of the Sensor Data – Motion vs. Static Classification

An accelerometer provides measurements about the instantaneous variations of speed over time. These variations are higher when walking that when standing still, and can be used to infer the motion state of the user. In [18], an extensive measurement campaign was conducted using an accelerometer embedded in the TMote Invents. This device provides acceleration measurements along two axes (i.e., vertical and horizontal). During a calibration stage, the accelerations of an operator who walks and stops at regular intervals is recorded and the data are tagged with the type of motion observed. Each sample is taken every 4 s. Ten measurements along each axis were collected at a rate of 100 Hz.

A first conclusions of this extensive measurement campaign is that the *absolute* values for the acceleration do not directly allow to distinguish motion from stillness. On the contrary, the variance of the acceleration enables a discrimination between the motion and non-motion configurations, as illustrated in Fig. 16. From that on, the collected data can be automatically put into two classes by means of classification techniques such as the linear discriminant function [35]. This calibration phase will be used at a later time to determine to which class a subsequent sample belongs to.

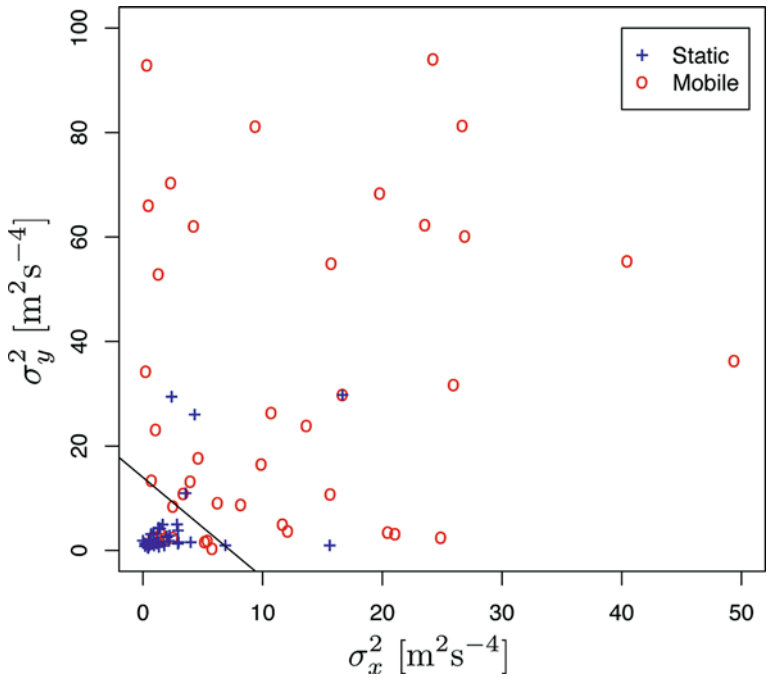


Fig. 16 Classification of the terminal mobility. It is characterized by the variance of the acceleration along the two axes of the mobile sensor

A Support Vector Machine (SVM) [35] works as follows. Let us consider a given set of training data

$$\mathcal{D} = \left\{ (\xi_{\mathbf{i}}, c_i) \mid \xi_{\mathbf{i}} \in \mathbb{R}, c_i \in \{C_1, C_2\} \right\}$$

where $\xi_{\mathbf{i}}$ is the i-th vector of data and c_i denotes the class it belongs to, i.e., C_1 or C_2. The objective of linear classification is to identify, in the axes $\{\psi_1, \psi_2, \ldots, \psi_n\}$, the hyperplane

$$\mathcal{H} \equiv \mathbf{A}.\psi = \mathbf{b}$$

such that the distance between the hyperplane and the classes C_1, C_2 is maximum. Let us consider that a data ψ_i belongs to the class C_1 (C_2) if $c_i = \alpha$ ($c_i = -\alpha$). Two parallel hyperplanes can be defined: one for each edge of the data classes.

$$\begin{cases} \mathcal{H}_1 \equiv \mathbf{A}.\psi_{\mathbf{i}} - \mathbf{b} = \alpha \quad , \quad \psi_i \in C_1 \\ \mathcal{H}_2 \equiv \mathbf{A}.\psi_{\mathbf{i}} - \mathbf{b} = -\alpha \ , \quad \psi_i \in C_2 \end{cases}$$

Figure 17 illustrates this technique in the case of a two-dimensional dataset. The classes C_1 and C_2 are represented by the circles and the triangles, respectively. It

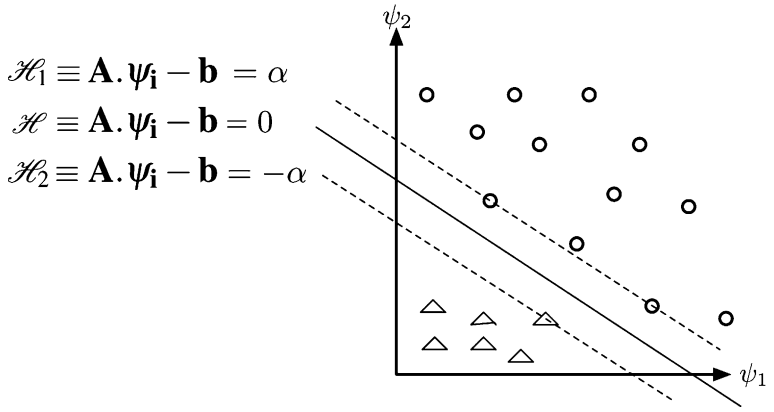


Fig. 17 Illustration of the classification method used in support vector machine

can be seen that the separation hyperplane $\mathcal{H}$ can be geometrically interpreted as the middle of the two hyperplans $\mathcal{H}_1$ and $\mathcal{H}_2$.

$$\begin{cases} \mathbf{A}.\psi_\mathbf{i} - \mathbf{b} \geq \quad \alpha \ , \quad \psi_i \in C_1 \\ \mathbf{A}.\psi_\mathbf{i} - \mathbf{b} \leq -\alpha \ , \quad \psi_i \in C_2 \end{cases}$$

Finally, in order to increase the robustness of the discrimination, it is usual to consider that the hyperplane must be located *as far as possible* from both classes. Simple geometry computation shows that the distance between the two hyperplanes $\mathcal{H}_1$ and $\mathcal{H}_1$ is

$$||\mathcal{H}_1 - \mathcal{H}_2|| = \frac{2\alpha}{||\mathbf{A}||}$$

Therefore, the maximization of this quantity implies the minimization of the value $||\mathbf{A}||$. Finally, the classification problem can be written as the optimization program

$$\begin{cases} \min ||\mathbf{A}|| \\ c_i \ (\mathbf{A}.\psi_\mathbf{i} - \mathbf{b}) \geq \alpha, \quad \forall i \in 1, \ldots, n \end{cases} \tag{11}$$

This family of problems is referred to as *quadratic programming (QP) optimization problem* and can be solved using standard techniques [34, 28, 5].

Referring back to our static-vs-motion example, the estimation of the variance of the acceleration along the two axes is $\mathbf{x} = (\sigma^2_{\text{vertical}}, \sigma^2_{\text{horizontal}})$. The two classes of interest are $C_1 \equiv \mathcal{S}$ and $C_2 \equiv \mathcal{M}$, which stands for the static and motion conditions, respectively. A linear discriminant is derived from (11) and a threshold value is used γ that determine to which condition the observation belongs to. More precisely, the sensor node can determine the motion condition of the user by sampling the values of the accelerometer(s) and computing the linear relations:

$$\begin{cases} \psi^T \mathbf{A} > \gamma \Rightarrow \mathbf{x} \in \mathcal{S} \\ \psi^T \mathbf{A} \leq \gamma \Rightarrow \mathbf{x} \in \mathcal{M} \end{cases}$$

In the proposed example, a standard SVM is used to identify the parameters $\mathbf{A}$ and γ. The resulting separating line is displayed in Fig. 16 and the observed percentage of correct classification is about 90%, which shows that the fusion of data generated by sensors monitoring the mobility condition can be integrated in the localization task. Moreover, the data fusion module was experimentally shown to both significantly reduce the error and the variance of the estimate of the mobile target position when used to suitably customize a recursive filter such as a Kalman filter [18], as presented in the next section.

7.3 A Data Fusion Approach for the Kalman Filter

Under the hypothesis that the motion state of the user is perfectly known, it is possible to further refine the Kalman model presented in Sect. 5. More precisely, the terms corresponding to the unknown speed component in the variable $\mathbf{x}_k$ can be set to $s_{x,y} = 0$ ms^{-1} or $s_{x,y} = 1$ ms^{-1} when the user is in static or in motion, respectively. The relation (9) becomes:

$$\underbrace{\begin{bmatrix} x \\ y \\ s_x \\ s_y \end{bmatrix}_k}_{\mathbf{x}_k} = \underbrace{\begin{bmatrix} 1 & 0 & \eta_k & 0 \\ 0 & 1 & 0 & \eta_k \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{F}_k} \underbrace{\begin{bmatrix} x \\ y \\ s_x \\ s_y \end{bmatrix}_{k-1}}_{\mathbf{x}_{k-1}} + \underbrace{\begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \\ \eta_k & 0 \\ 0 & \eta_k \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix}_k}_{\mathbf{w}_k} \tag{12}$$

where

$$\eta_k = \begin{cases} 0 & \text{if user is staticat } t = t_k \\ \Delta t & \text{if the user is in motionat } t = t_k \end{cases}$$

Note that relation (12) differs from (9) since the value of η_k strictly depends on the result of the classification algorithm applied to the measurements performed by the accelerometer. The relation (10) is equivalently modified:

$$\boldsymbol{\sigma}_k^w = \text{cov}(\mathbf{w}_k) = \mathbb{E}\left[\mathbf{w}_k \mathbf{w}_k^T\right] = \frac{1}{2} \begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \\ \eta_k & 0 \\ 0 & \eta_k \end{bmatrix} \begin{bmatrix} \Delta t^2/2 & 0 & \eta_k & 0 \\ 0 & \Delta t^2/2 & 0 & \eta_k \end{bmatrix}$$

All previous developments presented in the above sections still hold and can be re-used as-is. Note that this new approach requires a perfect (or *almost* perfect) knowledge about the state of the motion. Moreover, according to this new inter-

pretation of the Kalman model, this time the acceleration is supposed to be known and measurable. Since the information derived from the accelerometer contains an uncertainty (i.e., the sensor performs a noisy measure), we provide a comparison of both methods, i.e., with and without the a priori knowledge of the user's motion in the next section.

7.4 Performance Evaluation of the Fusion Schemes

Several experiments were conducted in the basement of an indoor building made of large corridors and metal structures. In a first round, the data issued from the accelerometer were processed. It has been confirmed that the variance of the acceleration is a good estimator to detect the user's behaviour, i.e., whether it is static or in motion. Figure 18 reports the Root Mean Square Error (RMSE) of the estimated position by the multilateration technique without additional filtering. One can observe that the average positioning error in our scenario is $\mu = 5.4\,\text{m}$ and that the technique presents a significant variance ($\sigma = 4.9\,\text{m}$). In a second experiment, shown in Fig. 19, the samples were collected at the accelerometer and the fusion technique was applied. It can be seen that, in this case, the average RMSE of the estimation lowers to 3.4 m while, at the same time, the variance of the error is divided by 3, i.e., $\sigma = 1.6\,\text{m}$.

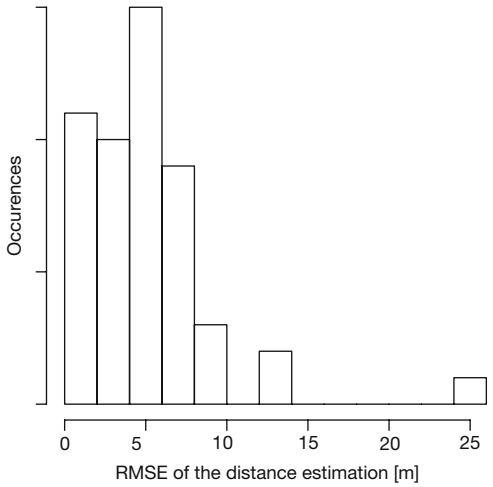


Fig. 18 Distribution of the RMSE of the position as evaluated by the multilateration algorithm

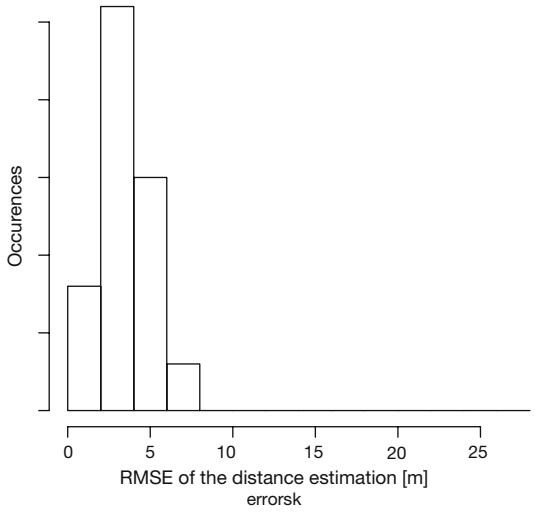


Fig. 19 Distribution of the RMSE of the position after filtering and fusion with the information of the accelerometer

7.5 Fusion of Localization Estimators

In order to further improve the localization techniques presented in this chapter and to reduce the variance of the inferred position, the *fusion of estimators* can be used. In this method, instead of considering a single estimation of the position, one generates all possible subsets containing at least $k \geq 3$ anchors. The amount of possible combinations when picking up a subset of k anchors among the N available distance estimations is

$$C_k = \binom{N}{k} = \frac{N!}{k!(N-k)!}$$

For each subset of size C_k, a multilateration technique (i.e., circular or hyperbolic) is used to determine the C_k position estimates. By repeatedly applying this for all $k \in \{N, N-1, N-2, \ldots, 3\}$, it is possible to get

$$C_{\text{tot}} = \sum_{k=3}^{N} C_k = \sum_{k=3}^{N} \frac{N!}{k!(N-k)!}$$

position estimates. These estimates are linearly combined and multiplied by a weight that is proportional to the inverse of the amount of subsets that can be derived, i.e., the more the anchors are used, the higher the weight value. This

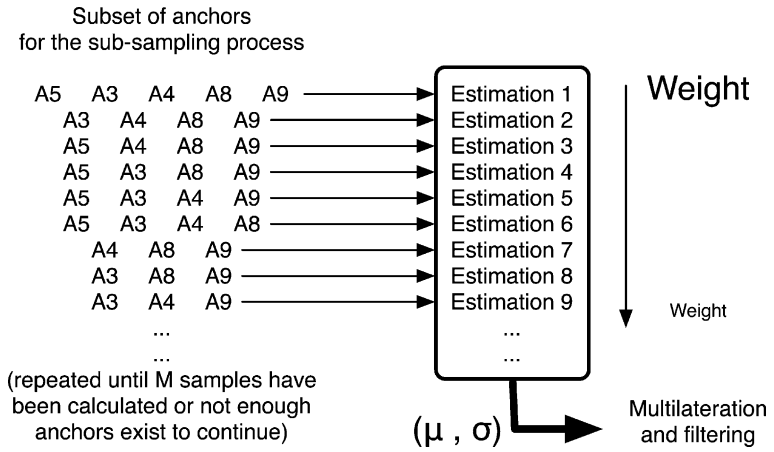


Fig. 20 Fusion of estimators and weighting of the estimates

weighted sum is efficient since the combination of multiple estimations reduces the variance of the estimation and efficiently reduces the bias in the estimation [19]. Figure 20 presents a schematic illustration of the subsampling and weighting process. The average position estimate can be computed as

$$\tilde{\mathbf{x}} = \sum_{k=3}^{N} \omega_k \sum_{i=1}^{C_k} \omega_{ki}\, \tilde{\mathbf{x}}_{ki} \tag{13}$$

where $\tilde{\mathbf{x}}_{ki}$ is the i-th estimate obtained from the fusion of estimators using k anchors and ω_{ki} is the corresponding weight. A simple method consists of employing an iid. weight, that is, each set of subsamples has the same weight ($\omega_k = 1/N - 3$) and inside each set of level k, the estimates share a same weight ($\omega_{ki} = 1/C_k$). Therefore, the relation

$$\tilde{\mathbf{x}} = \frac{1}{N-3} \sum_{k=3}^{N} \sum_{i=1}^{C_k} \frac{1}{C_k}\, \tilde{\mathbf{x}}_{ki} \tag{14}$$

gives the position estimate that derives from the fusion of estimators method.

8 Open Issues in Localization and Conclusion

In this chapter, we have been presenting several localization schemes to address the static and the dynamic localization problems in wireless sensor networks. Despite the increasing attention that the sensor network architectures have received in this last decade, several open issues remain. First, the security of the localization architecture can be easily compromised since the sensor network technology is

based on wireless networks protocols [21]. The robustness against wormholes, Sybil attacks, and the impact of the compromised networks nodes has been studied in the past [25]. Moreover, in the context of the localization, it has been shown that a secure verification of the location estimates may also be mandatory [32].

Second, the computation of the position estimates requires the collection and the dissemination of a substantial amount of data. Although the most accurate possible location estimate is an ultimate objective, the achievable level of geographical granularity and the update frequency are inherently limited by the available shared bandwidth, which is often low in the context of wireless sensor networks. Moreover, in sensor networks, an ad-hoc architecture is considered and it will have a strong impact on the performance of the data dissemination and the amount of time between the sensing and the location derivation (also referred to as *time to first fix*). As then, the deployment of a practical localization architecture requires a prior and in-depth analysis involving very different notions such as: the deployment costs, the user benefits, the network overhead consumption, and the minimal requirement constraints. These requirements are expected to vary from one considered scenario to the other and it is hard to define an holistic architecture that will be suitable and optimal (or at least, near to optimal) for any situation.

Third, all localization techniques are very sensitive to the deployment environment, but for different reasons. The range-free algorithms can be subject to diffraction, which makes angle estimates void. In the case of range-based techniques, the acquisition of the signal strength is subject to temporal and spatial fading due to multipath propagation. As a consequence, the presence of an non-homogeneous propagation environment (e.g., existence of a metallic storage facility, deployment in a mixed outdoor-to-indoor scenario, architecture of the considered building, etc.) may lead to inconsistent distance estimates. However, techniques such as the subsampling presented in Sect. 7.5 may help detecting these wrong estimates, though it remains a very difficult task. Furthermore, if a mobile scenario is considered, the signal is also subject to shadowing and non-isotropic radiation of the antennas, which means that the signal arriving at the receiver of the anchors may change rapidly, even if the target moves only a few meters away.

As a conclusion, several possible techniques can be used to provide an information of the position of a target node of the sensor network. For each technique, an associated error can be evaluated and it is interesting to note that the sources of inaccuracies are different in each considered approach. Therefore, in order to address these inaccuracies, an interesting direction is to evaluate the potential of a data fusion in a multimodel localization context [17, 15]. Indeed, if more than one localization scheme is used and a Bayesian inference technique [7] is applied, a more accurate positioning can be obtained by injecting the knowledge acquired from one technique into the subsequent ones.

Acknowledgements The authors would like to acknowledge Yann-Ael Le Borgne and Mathieu Van Der Haegen for providing them with the programming of the nodes and the analysis of the datasets. The support of the Belgian National Fund for Scientific Research (FRS-FNRS) is gratefully acknowledged.

References

1. Albowicz, J., Chen, A., Zhang, L.: Recursive position estimation in sensor networks. In: Proceedings of the Ninth International Conference on Network Protocols, p. 35. IEEE Computer Society, Washington, DC, USA (2001)
2. Bahl, P., Padmanabhan, V.N.: RADAR: an in-building RF-based user location and tracking system. In: Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 2, pp. 775–784 (2000)
3. Bischoff, U., Strohbach, M., Hazas, M., Kortuem, G.: Constraint-based distance estimation in ad-hoc wireless sensor networks. In: Proceedings of the Third European Workshop on Wireless Sensor Networks, pp. 54–68 (2006)
4. Blackmore, P., Bitmead, R.: Duality between the discrete-time Kalman filter and LQ control law. IEEE Transactions on Automatic Control 40(8), 1442–1444 (1995)
5. Bomze, I.M., Klerk, E.D.: Solving standard quadratic optimization problems via linear, semidefinite and copositive programming. Journal of Global Optimization 24, 163–185, October (2002)
6. Boukerche, A., Oliveira, H.A.B.F., Nakamura, E.F., Loureiro, A.A.F.: A novel lightweight algorithm for time-space localization in wireless sensor networks. In: Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems, pp. 336–343. ACM, New York, USA (2007)
7. Box, G.E.P., Tiao, G.C.: Bayesian Inference in Statistical Analysis (Wiley Classics Library). Wiley-Interscience, New York (1992)
8. Brown, R., Hwang, P.: Introduction to Random Signals and Applied Kalman Filtering. Wiley, New Yrok (1996)
9. Bulusu, N., Heidemann, J., Estrin, D.: GPS-less low-cost outdoor localization for very small devices. IEEE Personal Communications 7(5), 28–34 (2000)
10. Bulusu, N., Heidemann, J., Estrin, D.: Adaptive beacon placement. In: Twenty-first International Conference on Distributed Computing Systems (ICDCS-21). University of California, Los Angeles, IEEE Computer Society (2001)
11. Catedra, M.F., Perez, J.: Cell Planning for Wireless Communications. Artech House, Inc., Norwood, MA (1999)
12. Chen, J., Yao, K., Hudson, R.: Source localization and beamforming. IEEE Signal Processing Magazine 19(2), 30–39 (2002)
13. Coates, M.: Distributed particle filters for sensor networks. In: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, pp. 99–107. ACM, New York, USA (2004)
14. Costa, J.A., Patwari, N., Alfred, O. Hero, I.: Distributed weighted-multidimensional scaling for node localization in sensor networks. ACM Transactions on Sensor Networks 2(1), 39–64 (2006)
15. Cui, X., Zhao, Z., Tao, Q.: Probabilistic behavior of sensor network localization. In: ISPA Workshops, pp. 444–453 (2005)
16. Doherty, L., Pister, K.S.J., El Ghaoui, L.: Convex position estimation in wireless sensor networks. In: INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, vol. 3, pp. 1655–1663 (2001)
17. Dong, S.L., Wei, J.M., Xing, T., Liu, H.T.: Constraint-based fuzzy optimization data fusion for sensor network localization. In: Proceedings of the Second International Conference on Semantics, Knowledge, and Grid, p. 59. IEEE Computer Society, Washington, DC, USA (2006)
18. Dricot, J.M., der Haegen, M.V., Borgne, Y.A.L., Bontempi, G.: A modular framework for user localization and tracking using machine learning techniques in wireless sensor networks. In: Proceedings of the 7th IEEE Conference on Sensors (2008)
19. Good, P.I., Lunneborg, C.E.: Introduction to Statistics Through Resampling Methods and R/SPLUS. Wiley, John & Sons, New York (2005)

20. He, T., Huang, C., Blum, B.M., Stankovic, J.A., Abdelzaher, T.: Range-free localization schemes for large scale sensor networks. In: MobiCom '03: Proceedings of the 9th Annual International Conference on Mobile Computing and Networking, pp. 81–95. ACM, New York, USA (2003)
21. Hubaux, J.P., Butty, L., Capkun, S.: The quest for security in mobile ad-hoc networks. In: Proceedings of the 2nd ACM International Symposium on Mobile Ad-hoc Networking & Computing (MobiHoc'01), pp. 146–155 (2001)
22. Imieliński, T., Navas, J.C.: Gps-based geographic addressing, routing, and resource discovery. Communications of the ACM 42(4), 86–92 (1999)
23. Ji, X., Zha, H.: Sensor positioning in wireless ad hoc networks using multidimensional scaling. In: IEEE INFOCOM (2004)
24. Ko, Y.B., Vaidya, N.H.: Location-aided routing (LAR) in mobile ad hoc networks. In: International Conference on Mobile Computing and Networking (MobiCom'98). ACM/IEEE (1998)
25. Lazos, L., Poovendran, R.: SeRLoc: Robust localization for wireless sensor networks. ACM Transactions on Sensor Networks 1(1), 73–100 (2005)
26. Nakamura, E.F., Loureiro, A.A.F.: Information fusion in wireless sensor networks. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1365–1372. ACM, New York, USA (2008)
27. Niculescu, D., Nath, B.: Ad hoc positioning system (APS) using AoA. In: Proceedings of INFOCOM (2003)
28. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Series in Operations Research. Springer, New York (1999)
29. Oliveira, H.A.B.F., Nakamura, E.F., Loureiro, A.A.F., Boukerche, A.: Directed position estimation: A recursive localization approach for wireless sensor networks. In: Proceedings of IC3N'05, pp. 557–562. San Diego, CA, USA (2005)
30. Proakis, J.: Digital Communications. McGraw-Hill, New York (2000)
31. Rappaport, T.S.:Wireless Communications. Principles & Practice, 2nd Edition, Prentice-Hall, Upper Saddle River, NJ (2002)
32. Sastry, N., Shankar, U., Wagner, D.: Secure verification of location claims. In: ACM Workshop on Wireless Security, pp. 1–10. ACM Press, New York, USA (2003)
33. Savvides, A., Park, H., Srivastava, M.B.: The n-hop multilateration primitive for node localization problems. Mobile Networks and Applications 8(4), 443–451, August (2003)
34. Scozzari, A., Tardella, F.: A clique algorithm for standard quadratic programming. Discrete Applied Mathematics 156(13), 2439–2448 (2008)
35. Shawe-Taylor, J., Cristianini, N.: Support Vector Machines. Cambridge University Press, Cambridge (2000)
36. Shi, J., Müller, H.C., Marx, M.: Pedestrian detection and localization using antenna array and sequential triangulation. In: Proceedings of IEEE 2005 Conference on Intelligent Transportation Systems, pp. 126–130. Vienna, Austria, September (2005)
37. Spanos, D.P., Olfati-Saber, R., Murray, R.M.: Approximate distributed Kalman filtering in sensor networks with quantifiable performance. In: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, p. 18. IEEE Press, Piscataway, NJ, USA (2005)
38. Ward, A., Jones A. Hopper, A.: A new location technique for the active office. IEEE Personal Communications 4(5), 42–47 (1997)
39. Xu, Y., Yang, J., Jin, Z.: A novel method for fisher discriminant analysis. Pattern Recognition 37(2), 381–384 (2004)

Enhancing Underwater Acoustic Sensor Networks Using Surface Radios: Issues, Challenges and Solutions

Zhong Zhou, Hai Yan, Saleh Ibrahim, Jun-Hong Cui, Zhijie Shi, and Reda Ammar

Abstract In this chapter, we focus on a novel network architecture, UASN-MG, which has multiple surface gateways for underwater acoustic sensor networks. The surface gateways communicate with each other using surface radio and form a radio network on the water surface. Thus, two networks, an underwater acoustic network formed by underwater sensor nodes and a surface radio network formed by the surface gateways, are involved. Such a network architecture can bring about many benefits such as high throughput and energy efficiency. However, integrating multiple surface gateways into underwater acoustic sensor networks also incurs many new design challenges. In this chapter, we describe this new network architecture and its benefits in detail. Then we discuss the design challenges in this new architecture. Afterwards, three design examples are given, from different perspectives, to illustrate the techniques that cope with these challenges and optimize the network's performance. Finally, we conclude the chapter with discussions on possible future research directions.

1 Introduction

Underwater acoustic sensor networks have attracted rapidly growing research interests in last several years [1, 5, 6, 9, 17, 21]. On the one hand, underwater sensor networks enable a wide spectrum of applications, such as oceanographic data collection, pollution monitoring, offshore exploration, disaster warning, and tactical surveillance applications. On the other hand, the adverse underwater environments pose grand challenges for efficient communication and networking.

In underwater environments, radio does not work well due to its quick attenuation in water. Thus acoustic channels are usually employed. The propagation speed of acoustic signals in water is about 1.5×10^3 m/s, which is five orders of magnitude lower than the radio propagation speed in air (about 3×10^8 m/s). In addition, the bandwidth capacity of underwater acoustic channels is limited and depends on

Z. Zhou (✉)
Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA
e-mail: zhong.zhou@uconn.edu

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_11,

both transmission range and frequency [12, 31, 2, 3]. According to [13], nearly no research and commercial system can exceed 40 km × kbps as the maximum attainable range × rate product. In comparison, the product for RF (radio frequency) communications yields up to 5000 km × kbps, if we take 802.11b/a/g as an example. Moreover, underwater acoustic channels are affected by many factors such as path loss, noise, multi-path fading, and Doppler spread. All these cause high error probability in acoustic channels. In short, underwater acoustic channels feature long propagation delay, low available bandwidth, and high error probability. Besides, underwater sensor nodes are usually powered by battery, which is very hard to recharge or replace in hash underwater environments. High energy efficiency thus becomes one of the most important design considerations for such networks. All these new challenges call for innovative design philosophies and techniques, from network architecture to every network function and protocol.

Directly tackling the basic hard communication and networking problems have motivated a large strand of research efforts (such as [22, 23, 27, 32, 35–37]). In this chapter, we take a very different research route, investigating *how (powerful) surface radios help (limited) underwater acoustics*. We introduce a new network architecture, called **U**nderwater **A**coustic **S**ensor **N**etwork with **M**ultiple surface **G**ateways (UASN-MG). In UASN-MG, gateways equipped with both acoustics and radios are deployed at water surface, and they can communicate with each other using surface radios and form a radio network above water. Thus, UASN-MG involves two types of networks: one is the underwater acoustic network formed by underwater sensor nodes and the acoustic parts of gateways; the other is the surface radio network formed by the radio parts of gateways. Different from the heterogenous network architecture in chapter "Self-Organization of Sensor Networks with Heterogeneous Connectivity" [25], where sensor nodes are only different in their transmission range, surface gateway nodes in our UASN-MG are equipped with an extra set of radio transceiver which makes them much more powerful and costly than the ordinary underwater sensor nodes.

Compared with traditional underwater sensor networks (with single gateway), UASN-MG can drastically improve network performance, such as high reliability, low end-to-end delay, and high energy efficiency. However, introducing multiple surface gateways into an underwater acoustic sensor network is nontrivial, and in fact, has lots of design challenges. In this chapter, we elaborate on these challenges. Three typical design examples are given to show the way to cope with these challenges and optimize the network's performance from different perspectives.

The rest of the chapter is organized as follows. We first describe the new network architecture, UASN-MG, and discuss its benefits and advantages. Then, we discuss in depth the design challenges for such kind of networks. After that, we give three design examples. The first example is for the optimal deployment of surface gateways. We model the surface gateway deployment as an integer linear programming (ILP) problem and optimize the system's energy efficiency and end-to-end delay. In the second example, we discuss a new multi-hop routing protocol, depth-based routing(DBR), which is well suitable for UASN-MGs. DBR utilizes the depth information of every node and routes data packets greedily towards the gateways at the

water surface. Gateways can then forward the received data packets to the central sink node through the surface radio network. Through this example, we can clearly see the impacts of multiple surface gateways on the conventional layered protocols such as the routing protocol. The third example is on the cross-layer design between the physical layer and the routing layer for energy efficient reliable transfer in UASN-MGs. In this example, multiple acoustic paths are used to route data packets to multiple surface gateways. These gateways then do packet combining through the surface radio links. Power control is also integrated to improve the system's energy efficiency. This example clearly shows us the strength of cross-layer design in the multi-gateway network. We then conclude with some observations and discussions of future research directions.

2 UASN-MG Architecture and Its Benefits

Figure 1(a) gives an example of UASN-MG. In the plotted network, multiple surface gateways are deployed. Every gateway is equipped with two sets of transceivers. One is acoustic transceiver which is used to communicate with underwater sensor nodes through acoustic channels. The other is radio transceiver which is used to communicate with the control center or other gateways through radio channels.

As mentioned earlier, two types of networks are involved in this network architecture: one is the underwater acoustic network which is composed of the underwater sensor nodes as well as the acoustic parts of the surface gateways; the other is the surface radio network which is composed of the radio parts of the surface gateways. These two networks interact with each other through surface gateways. If an underwater node has data to report to the control center, the data will first go through the acoustic channels in the underwater acoustic network to one of the surface gateways, and then through the radio links in the surface radio networks to the control center.

Compared with traditional underwater sensor networks, UASN-MG introduces a new surface radio network on the top of the underwater acoustic network. Since radio channels have much higher bandwidth and are much more reliable than underwater acoustic channels, multiple gateways can bring a lot of benefits to UASN-MG with the surface radio network. Next, we will elaborate the advantages of UASN-MG from the aspects of energy efficiency, reliability and end-to-end delay, which are three important performance criteria for underwater sensor networks.

- *UASN-MG can achieve higher energy efficiency.* As shown in Fig. 1(b), in the traditional underwater acoustic sensor network, when a node has data to report, it sends the data to the control center, all through acoustic channels. It is well known that for the same transmission range, acoustic transceiver consumes much more energy than radio transceiver. While in UASN-MG, every route is a combination of both acoustic links and radio links. Energy thus can be saved by using the low energy-consuming radio links. More importantly, the new multiple surface gateway architecture can greatly reduce the collision probability on the

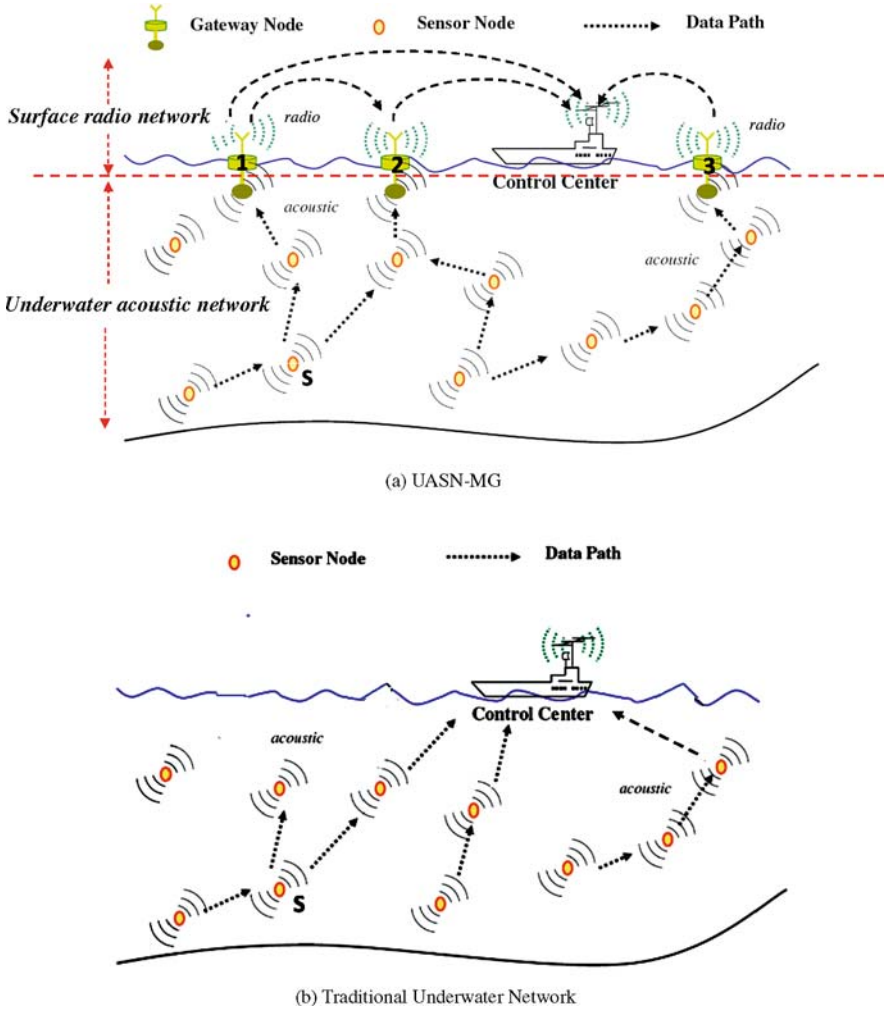


Fig. 1 Comparison between UASN-MG and traditional underwater network

acoustic channels, which leads to higher energy efficiency. With multiple surface gateways, underwater sensor nodes only need to send their packets to any of the surface gateways and it is the responsibility of the surface gateways to forward the packets to the control center (the final data sink). Hence, the traffic in the underwater acoustic network is dispersed to multiple surface gateways, instead of concentrating to one control center as in Fig. 1(b). In this way, the collision probability can be greatly reduced, especially in the area close to the control center.

- *UASN-MG can greatly improve the network reliability.* It goes without saying that multiple surface gateways contribute to more usable and efficient routes. For

example, as shown in Fig. 1(b), for the traditional underwater network, node "S" can only find one path which consists of pure acoustic channels. If any nodes along this path go wrong, this path is broken and data packets from node "S" will be lost. However, as shown in Fig. 1(a), for UASN-MG, node "S" can find routes to gateways "1" and "2". Both of them can forward the received data to the control center. Thus, if one route fails, data can still be forwarded to the control center along the other route. Multi-gateway architecture is also an effective way to combat the underwater fading environment. If a path to a gateway is in deep fading, other paths to other gateways might be in good conditions and can successfully transmit data. Thus, UASN-MG has the potential to provide much higher reliability than the traditional underwater network architecture.

- *UASN-MG can significantly reduce the end-to-end delay.* The benefits of the short delay of UASN-MG come from the following two aspects. First, since traffic is dispersed to multiple gateways, collision probability in the network will be reduced, which contributes to the low end-to-end delay since less time will be spent on the collision resolution process. Second, the acoustic route to a surface gateway in UASN-MG usually is much shorter than the acoustic route to the centralized control center in the traditional underwater network. The end-to-end delay thus can be significantly reduced considering that radio links are much more reliable and radio signal travels much faster than acoustic signal.

3 Research Challenges and Design Issues

Based on the previous discussions, no doubt UASN-MG has the great potential to bring significant benefits to underwater applications. To realize these benefits, however, poses many new design challenges. Approaches in the traditional underwater network architecture cannot be applied here directly. For example, routing protocols in traditional underwater sensor networks only try to find routes to the control center. If used in UASN-MG, they do not distribute the traffic among the gateways smartly in order to optimize the network performance. New protocols which integrate routing and traffic balancing are demanded. In fact, the impacts of UASN-MG are broad and almost every protocol layer is affected. In the following, we identify four fundamental research issues. The list is far from complete. We choose them as case studies in this chapter, with the hope that our research can cast light on this new direction in the community.

The first issue is *how to deploy surface gateways*. The network performance is heavily dependent on the deployment of surface gateways in UASN-MG. An optimal deployment not only improves the system throughput and energy efficiency, but also decreases the system end-to-end delay and collision probability. However, finding the optimal gateway locations of surface gateways is very challenging. The deployment of the gateways is not only related to the gateways themselves, but also related to the locations of underwater acoustic sensor nodes. Global network information is needed to formulate a complicated optimization problem, which

will increase the traffic overhead and computation complexity. In addition, for mobile underwater sensor networks, the surface gateways should be able to dynamically change their locations to adapt to the movement of the underwater sensor nodes.

The second issue of UASN-MG is on *medium access control (MAC)*. Existing MAC protocols designed for the traditional network architecture do not take different roles of nodes into account [4, 14, 20, 26, 32, 36]. However, in UASN-MG, surface gateways play much more important roles than ordinary sensor nodes. Further, the shallow nodes which act as traffic aggregators are much more important than deeper nodes. An efficient MAC protocol should be adaptive to the roles and the depths of the nodes. In addition, MAC protocols with multiple channels are reported to be able to greatly improve network throughput, decrease channel access delay and lower energy consumption [24, 29, 33]. UASN-MG is a natural network scenario to incorporate multiple channel technologies into practice. However, how to efficiently allocate data channels distributively among the surface gateways and the underwater sensor nodes is a nontrivial problem.

The third issue is on *efficient routing* for UASN-MG. Although a lot of routing protocols have been proposed for traditional underwater acoustic networks [11, 19, 23, 37], their performance is quite questionable in UASN-MG. In these protocols, one or multiple routes to the control center are established and used for data transmission. For example, in [37], a route pipe is established along the vector from the source to the control center. Since only one sink node exists in the traditional network, traffic will be merged to this node, which aggravates the collision around the control center. However, in UASN-MG, a packet only needs to be routed through the underwater acoustic channels to any of the surface gateways. Thus, traffic is distributed among these gateways. An intelligent routing protocol for UASN-MG should allocate traffic evenly and smartly among the gateways and find route efficiently. To devise such an intelligent protocol is not an easy task. Since every node generates its traffic independently, a node cannot make an optimal routing decision unless it knows the decisions of other nodes. However, in the long delay distributed underwater environment, how can a node obtain the decisions and status of other nodes? In addition, because of the narrow available bandwidth of acoustic channels, complex routing protocols cause too much communication overhead and thus cannot be applied here. Efficient routing protocol design for UASN-MG is very challenging.

The fourth issue is *reliable end-to-end data transfer* for UASN-MG. Forwarding data from source nodes to the control center efficiently and reliably is challenging in harsh underwater environments. Pioneering work such as [35] provides high end-to-end reliability with low energy cost for traditional underwater acoustic networks. Multiple gateways in UASN-MG give us more design choices, but they also make things more complicated. For example, to improve the end-to-end reliability, a node might transmit multiple copies along multiple roues to multiple gateways simultaneously. How to choose these routes and how to coordinate the nodes to provide certain end-to-end reliability efficiently is a big challenge.

In summary, existing techniques for traditional underwater networks cannot meet the needs of UASN-MG. New physical layer, MAC/data link layer and network layer solutions need to be developed.

4 Design Examples

This seciton gives three design examples in UASN-MGs. In the first example, we investigate the optimal deployment of surface gateways. We model the surface gateway deployment as an integer linear programming (ILP) problem and optimize the system's energy efficiency and end-to-end delay. In the second example, we discuss a new routing protocol, Depth-Based Routing (DBR), which utilizes the depth information to route data packets greedily towards the gateways at the water surface. Gateways can then forward received data packets to the central sink node through the surface radio network. This example clearly shows the impacts of multiple surface gateways on the design of conventional layered protocols such as routing. In the third example, we propose a new end-to-end reliable data transfer protocol, Multi-path Power control Transmission (MPT). MPT is a cross-layer design strategy between the physical layer and the routing layer for multi-gateway underwater acoustic networks. In MPT, multiple acoustic paths are used to route data packets to multiple surface gateways. These gateways then perform packet combining through the surface radio links. Power control is also integrated to improve the system's energy efficiency. This example shows the strength of cross-layer design in this new network architecture. More details of these design examples can be found in [10, 38, 39]

4.1 Optimal Surface Node Deployment

In this design example, we study the problem of surface gateway deployment and present guidelines for deciding the optimal number and locations of surface gateways for a given underwater acoustic sensor network. We focus on optimizing the cost of surface gateway deployment, by finding the minimum number and the locations of surface gateway nodes required to achieve a given design objective, which can be communication delay, energy consumption, fault-tolerance, or a combination of them. The surface gateway deployment problem is formulated as an integer linear programming (ILP) problem here.

4.1.1 Network Model

The surface gateway deployment problem can be modeled as a 3-D graph optimization problem. The nodes of the graph represent underwater sensors and candidate surface gateway positions. The problem is to choose a subset of the candidate

surface gateway positions to satisfy a set of flow conservation constraints, interference constraints, deployment cost constraints (e.g., the number of surface gateways), and/or the the network performance requirements.

- *Nodes*. Let V be the set of all underwater sensor nodes, T be the set of candidate surface node positions, and V' be the set of all nodes, i.e. $V' = V \cup T$.
  Let $I(v)$ be the set of nodes within the communication range of node v, i.e. $I(v) = \{w : w \in V, v \neq w, d(v, w) \leq R\}$, where $d(v, w)$ denotes the distance between v and w, and R is the communication range of sensor nodes.
- *Edges*. Let E be the set of all edges $e = (v, w)$, such that $v \in V$, $w \in I(v)$. Let $E_{\text{out}}(v)$ and $E_{\text{in}}(v)$ demote the outgoing and incoming edge set of v respectively. Then $E_{\text{out}}(v) = \{e(v, u) : (v, u) \in E\}, \forall v \in V$, and $E_{\text{in}}(v) = \{e(u, v) : (u, v) \in E\}, \forall v \in V$. Note that for any surface gateway node, the outgoing edge set is Φ, since the surface gateways only receive data (no relaying function).
- *Data Generation and Flow*. Let $g(v_i)$ be the packet generation rate at node $v_i \in V$, and let G be the total data generation rate, i.e., $G = \sum_{v \in V} g(v)$. Further, we use $f(e)$ to denote the total data flow (in both directions) on edge e.
- *Gateway Presence Indicator*. Let $x(t_i)$ be a binary variable that defines the surface gateway deployment at location t_i:

$$x(t_i) = \begin{cases} 1 & \text{if a node deployed at } t_i \\ 0 & \text{otherwise} \end{cases}, \quad \forall t_i \in T. \tag{1}$$

4.1.2 Problem Formulation

Constraint Formulation. With the definition above, we can formulate the constraints of the network.

- *Deployment Constraints*. Data can only be received at locations where surface nodes are deployed:

$$f(e_i) \leq x(t_j)G, \quad \forall t_j \in T, e_i \in E_{in}(t_j). \tag{2}$$

  In other words, no data will be received at locations without gateways.
- *Interference Constraints*. The interference model adopted in this work assumes that a node cannot send while it is receiving, which implies that the total data transfer rate sent and received at any node can not exceed the maximum capacity B of its communication link. Thus, for underwater sensor nodes we have

$$\sum_{e_o \in E_{\text{out}}(v)} f(e_o) + \sum_{e_i \in E_{\text{in}}(v)} f(e_i) < B, \quad \forall v \in V, \tag{3}$$

  and for surface candidate gateways we have

$$\sum_{e_i \in E_{\text{in}}(t)} f(e_i) < B, \quad \forall t \in T. \tag{4}$$

- *Per-Node Flow Conservation*. Flow conservation implies that for underwater sensor nodes, the sum of the flows leaving a node equals the sum of the flows entering that node plus the local data generation rate, that is

$$\sum_{e_o \in E_{\text{out}}(v)} f(e_o) - \sum_{e_i \in E_{\text{in}}(v)} f(e_i) = g(v), \quad \forall v \in V. \tag{5}$$

- *End-to-End Flow Conservation*. Each surface node can act as a sink, and all surface gateways together form a virtual sink. Thus, a packet generated by any source, must eventually be received by a surface node. End-to-end flow conservation here means that the total data generation rate must equal the total data absorption rate by all surface node sensors, that is

$$\sum_{t_j \in T} \sum_{e_i \in E_{\text{in}}(t_j)} f(e_i) = G. \tag{6}$$

- *Deployment Cost*. For the multi-sink architecture, the deployment cost of surface gateways is of critical concern. To optimize the network performance, such as minimizing delay or energy consumption, using a limited number of surface nodes, N, we should include the following constraint:

$$\sum_{t_i \in T} x(t_i) \leq N. \tag{7}$$

Objective Functions. With an objective function, by solving the set of equations (1) through (7), an optimal deployment is obtained in the form of an assignment to the set of binary variables $x(t_i)$. In the following, we present some representative objective functions we have explored, namely minimizing the expected delay, minimizing the expected energy consumption, and minimizing the maximum delay.

- *Minimizing Expected Delay*. The objective is to minimize the expected end-to-end delay for all packets. The end-to-end delay for a packet is the sum of the per-hop delay over the entire path from the source that generates the packet to the sink that receives it. Since queuing delays (caused at MAC or routing layer) are not considered in this work, the per-hop delay consists of transmission delay and propagation delay. The delay t on an edge e can be written as

$$t(e) = t_s(e) + t_p(e) = \frac{L}{B} + \frac{l(e)}{v_p},$$

where L is the packet length in bits, B is the channel capacity in bits per second, $l(e)$ is the distance between the nodes at the two ends of e, and v_p is the propagation speed of sound in water. The expected delay for all packets then can then be written as

$$E[t(e)] = \frac{1}{G}\left(\sum_{e\in E} f(e)t(e)\right),$$

and the corresponding objective function will be

$$\text{Minimize}\big(E[t(e)]\big).$$

- *Minimizing Expected Energy Consumption*. The objective is to minimize the expected end-to-end energy consumption, i.e., the energy consumed in the network for a packet to travel from its source to a sink. The energy consumed ε to transmit a packet over edge e can be written as

$$\varepsilon(e) = \pi_s(e)t_s = \pi_s(e)\frac{L}{B},$$

where $\pi_s(e)$ is the transmission power used on edge e for one unit of data.
Similar to the expected delay, the expected energy consumption can be written as

$$E[\varepsilon(e)] = \frac{1}{G}\left(\sum_{e\in E} f(e)\varepsilon(e)\right),$$

and the corresponding objective function will be

$$\text{Minimize}\big(E[\varepsilon]\big).$$

- *Minimizing Maximum Delay*. The objective functions presented so far reflect the expected performance of the overall network. A more precise optimization of performance can consider data packets generated by individual sources separately. For example, the objective can be to minimize the worst-case expected delay seen by data packets originating at any specific source node. To do that, we define $f_{v_s}(e)$ as the portion of $f(e)$ originating at source node v_s. It follows that $f(e) = \sum_{v_s\in V} f_{v_s}(e)$, $\forall e \in E$. In this case, flow conservation constraints have to be detailed such that every sub-flow has its own conservation equation as follows.

$$\begin{aligned}\sum_{e_o\in E_{\text{out}}(v_i)} f_{v_s}(e_o) - \sum_{e_i\in E_{\text{in}}(v_i)} f_{v_s}(e_i) \\ = \begin{cases} g(v_s) & \text{if } v_i = v_s \\ 0 & \text{otherwise} \end{cases}, \ \forall v_s, v_i \in V,\end{aligned} \tag{8}$$

and

$$\sum_{t_j\in T}\sum_{e_i\in E_{in}(t_j)} f_{v_s}(e_i) = g(v_s), \ \forall v_s \in V. \tag{9}$$

The objective function can then be written as

$$\text{Minimize}\left[\text{Max}_{v_s \in V}\left(\sum_{e \in E} f_{v_s}(e)t(e)\right)\right].$$

4.1.3 Simulation Results

Throughout the simulations, we fix the packet length $L = 400$ bits, the propagation speed of sound in water $v_p = 1500$ m/s, and the transmission power $\pi_s = 1$ watt (we ignore receiving and idle power, which are much less than transmission power). We also fix the area of network deployment to a square of 600 m × 600 m horizontal extent, and fix the candidate gateway deployment positions to a 5×5 mesh of points spaced 150 m apart. The communication range for both underwater sensors nodes and acoustic interface of the surface gateways nodes is $R = 150$ m. The depth of all underwater sensors is set to 100 m.

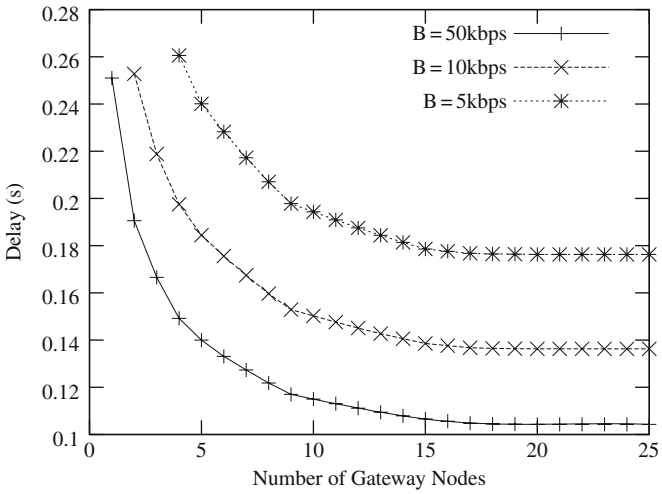


Fig. 2 Average delay, uniform underwater deployment

We vary the number of allowed surface nodes from 1 to 25 nodes and solve the optimization problem. Results show that an increase in the number of surface gateways can dramatically enhance the network performance, especially when the network is lightly loaded. For example, Fig. 2 shows that the expected delay corresponding to $B = 50$ kbps can be reduced from 0.26 to 0.16 s by using four surface gateways instead of one, as exactly shows the benefits of the multi-sink architecture. Figure 3 shows that the average energy consumption also decrease monotonically with the increase of the number of surface gateways. It is also worth noting that the improvement gained by adding a surface gateway diminishes as the number of surface gateways increases. After a certain number (a threshold) of surface gate-

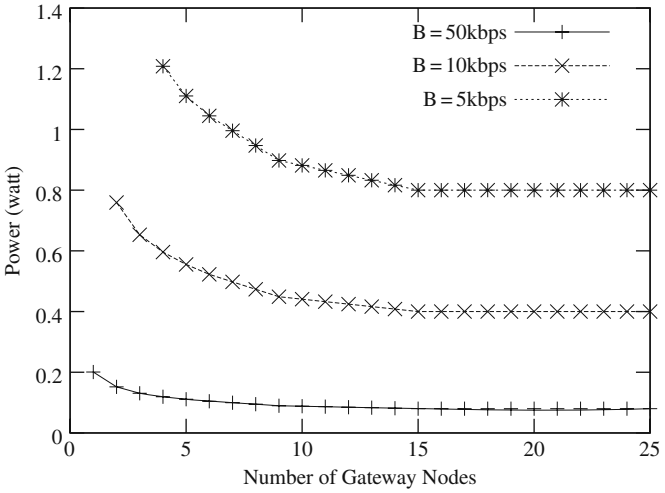


Fig. 3 Average energy, uniform underwater deployment

ways, depending on the underwater deployment (as well as other factors), additional surface nodes have no positive effect on the performance of the network. This is due to the fact that when the threshold is reached, every underwater sensor node could communicate with a surface gateway at the nearest candidate position. Therefore, any further addition of surface nodes becomes redundant.

In this example, we formulate the surface gateway deployment problem as an optimization problem, the solution of which gives the minimum number and the optimal locations of surface gateway nodes, with a variety of optimization goals. Simulation results confirm the potential for performance improvement using multiple surface gateways: reducing both average delay and energy consumption.

4.2 Efficient Routing Protocol

In this example, we propose a novel routing protocol, called *depth-based routing (DBR)*, for underwater sensor networks. DBR utilizes the properties of UASN-MGs: data sinks are usually situated at the water surface. Thus based on the depth information of each sensor, DBR greedily forwards data packets towards the water surface (i.e., the plane of data sinks). In DBR, a data packet has a field that records the depth information of its recent forwarder and is updated at every hop. The basic idea of DBR is as follows. When a node receives a packet, it forwards the packet if its depth is smaller than that embedded in the packet. Otherwise, it discards the packet. Obviously, if there are multiple data sinks deployed at the water surface, DBR can naturally take advantage of them. Packets reach any of the sinks are treated as successfully delivered to the final destination since these water-surface sinks can communicate with each other efficiently through radio channels, which have much higher bandwidth and much lower propagation delays.

To summarize, the main advantages of DBR are as follows. (1) It does not require full-dimensional location information. (2) It can handle dynamic networks with good energy efficiency. (3) It takes advantage of multiple-sink network architecture without introducing extra cost.

4.2.1 Network Model

In the network, multiple surface gateways equipped with both radio-frequency (RF) and acoustic modems are deployed at the water surface. Underwater sensor nodes with acoustic modems are distributed in the interested 3-D area, with each likely to be a data source. They can collect data and also help relay data to the gateways. Since all the gateways have RF modems, they can communicate with each other very efficiently via radio channels. Hence, if a data packet arrives at any gateway, we assume it can be delivered to other gateways or remote data centers efficiently. This assumption can be easily validated by the fact that acoustic signal propagates (at a speed of 1.5×10^3 m/s in water) five orders of magnitudes slower than radio (with a propagation speed of 3×10^8 m/s in air). We do not consider the communication between surface gateways in this example. Instead, we assume that a packet reaches the destination as long as it is successfully delivered to any gateway.

Furthermore, we assume that each underwater node knows its depth information, namely the vertical distance from itself to the water surface. In practice, depth information can be obtained easily with a depth sensor. In comparison, obtaining full-dimensional location information is much more difficult.

4.2.2 Protocol Overview

DBR is a greedy algorithm that tries to deliver a packet from a source node to gateways. During the course, the depth of forwarding nodes decreases while the packet approaches the destination. If we reduce the depth of the forwarding node in each step, eventually a packet can be delivered to the water surface. In DBR, a sensor node distributively makes its decision on packet forwarding, based on its own depth and the depth of the previous sender. This is the key idea of DBR.

In DBR, upon receiving a packet, a node first retrieves the depth d_p of the packet's previous hop, which is embedded in the packet. The receiving node then compares its own depth d_c with d_p. If the node is closer to the water surface, i.e., $d_c < d_p$, it will consider itself as a qualified candidate to forward the packet. Otherwise, it just simply drops the packet. In the latter case, the packet comes from a (better) node which is closer to the surface. So it is not desirable for the receiving node to forward the packet.

It is very likely that multiple neighboring nodes of a forwarding node are qualified candidates to forward a packet at the next hop. If all these qualified nodes try to broadcast the packet, high collision and high energy consumption will result. Therefore, to reduce collision as well as energy consumption, the number of forwarding nodes needs to be controlled. Moreover, due to the inherited multiple-path feature of DBR (in which each sensor node forwards packets in a broadcasting fashion

using an omnidirectional acoustic channel), a node may receive the same packet multiple times. Consequently, it may send the packet multiple times. To improve energy efficiency, ideally a node needs to send the same packet only once. We will address the techniques of suppressing redundant packets in the next section.

4.2.3 Redundant Packet Suppression

To save energy as well as reduce collision, redundant packets need to be suppressed. There are two major reasons for redundant packets. One is that multiple paths are naturally used to forward packets. The other is that a node may send a packet many times. Although multiple paths in DBR can not be completely eliminated, we use a *priority queue* to reduce the number of forwarding nodes, and thus control the number of forwarding paths. To solve the second problem, a *packet history buffer* is used in DBR to ensures that a node forwards the same packet only once in a certain time interval.

In DBR, each node maintains a *priority queue* $Q1$ and a *packet history buffer* $Q2$. An item in $Q2$ is a unique packet ID, which is composed of Sender ID and Packet Sequence Number. When a node successfully sends out a packet, it inserts the unique ID of the packet into $Q2$. When $Q2$ is full, the new item will replace the Least Recently Accessed (LRA) item. In other words, $Q2$ maintains a recent history of the packets the node has sent.

An item in $Q1$ includes two components: a packet and the scheduled sending time for the packet. The priority of an item in $Q1$ is represented by the scheduled sending time. More specifically, an item with earlier sending time has a higher priority. When a node receives a packet, instead of sending the packet immediately, it first holds the packet for a certain amount of time, called *holding time*. The scheduled sending time of a packet is computed based on the time when the packet is received and the holding time for the packet.

At a node, an incoming packet is inserted into $Q1$ if it has not been sent by the node before (i.e., its unique ID is not in $Q2$) and it was sent from a lower node (i.e., a node with a larger depth, $d_p > d_c$). If a packet currently in $Q1$ is received again during the holding time, the packet will be removed from $Q1$ if the new copy is from a node with a smaller or similar depth ($d_p \leq d_c$), or its scheduled sending time will be updated if the new copy is from a lower node ($d_p > d_c$). After a node sends out a packet as scheduled, the packet is removed from $Q1$ and its unique ID inserted into $Q2$.

4.2.4 Holding Time Calculation

As mentioned earlier, a node uses *holding time* to schedule packet forwarding. At a node, the holding time for a packet is calculated based on d, the difference between the depth of the packet's previous hop and the depth of the current node. Nodes with different depths will have different holding times even for the same packet. In order to reduce the number of hops along the forwarding paths to the water surface, DBR tries to select the neighboring node with the minimal depth to be the first one to

forward a packet. It also tries to prevent other neighboring nodes from forwarding the same packet to reduce energy consumption.

Figure 4 shows an example. Node S is the sender, and nodes n_1, n_2, and n_3 are all its one-hop neighboring nodes. The solid line circle represents the transmission range of node S. When node S broadcasts a packet, all neighboring nodes will receive this packet. Node n_3 is below S so it discards the packet. Although nodes n_1 and n_2 are both qualified forwarding nodes, node n_1 is preferred to forward the packet. The forwarding of node n_2 is prevented if it receives the packet from n_1 before its own scheduled sending time for the packet.

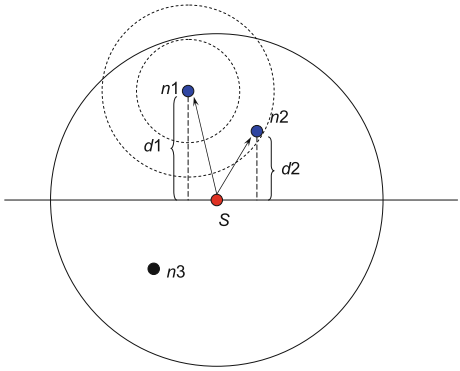


Fig. 4 Forwarding node selection

Based on the above analysis, we can see that the holding time must satisfy two conditions: (1) the holding time should decrease with the increase of d; and (2) the difference between the holding times of two neighboring nodes should be long enough so that the forwarding of the node with the smaller depth can be heard by the other node timely (before the lower node starts its own packet forwarding).

Let us still take Fig. 4 as an example to show how we calculate the holding time. We express the holding time using a linear function of d as follows, where d is the depth difference of the current node and the previous one.

$$f(d) = \alpha \cdot d + \beta \tag{10}$$

Let d_1 and d_2 be the depth difference at nodes n_1 and n_2, respectively. Let us assume that n_1 receives a packet from S at time t_1, n_2 receives the packet at time t_2, and t_{12} is the propagation delay between n_1 and n_2. Then the two conditions can be represented by the following inequalities:

$$f(d_1) < f(d_2), \tag{11}$$

and

$$t_1 + f(d_1) + t_{12} \leq t_2 + f(d_2). \tag{12}$$

Substituting $f(d)$ with our linear expression, we have

$$\alpha \leq \frac{(t_2 - t_1) - t_{12}}{d_1 - d_2}, (\alpha < 0).$$

Here α is negative. As long as $|\alpha| \geq \frac{(t_1-t_2)+t_{12}}{d_1-d_2}$, both conditions can be met. Considering the worst positions for n_1 and n_2, we can choose $|\alpha| = \frac{2\tau}{d_1-d_2}$, where $\tau = R/v_0$ is the maximal propagation delay of one hop (R is the maximal transmission range of a sensor node and v_0 is the sound propagation speed in water).

The value of α depends on $(d_1 - d_2)$, the depth difference of nodes n_1 and n_2. For a node's one-hop neighbors, α can vary between 0 and R, the maximal transmission range of a sensor node. When $(d_1 - d_2)$ approaches 0, $\alpha \to -\infty$. It shows that we cannot find a constant α to make condition (2) always satisfied. Instead, we use a global parameter δ to replace $(d_1 - d_2)$ for the holding time calculation. Therefore $\alpha = -\frac{2\tau}{\delta}$. We guarantee that node n_1 will forward a packet first and prevent the forwarding of node n_2 if $d_1 - d_2 \geq \delta$.

Let the node with minimal depth have holding time 0. We can compute the β by solving the following equation:

$$-\frac{2\tau}{\delta} R + \beta = 0$$

Substituding α and β in the linear function (1), we have the definition of $f(d)$ as follows:

$$f(d) = \frac{2\tau}{\delta} \cdot (R - d), \delta \in (0, R]. \tag{13}$$

When a small δ is chosen, nodes have longer holding times (if their depth differences with the previous forwarder are not exactly R). This may result in longer end-to-end delays. At the same time, the forwarding at these nodes is more likely to be suppressed by the forwarding from a neighboring node closer to the water surface, which results in lower energy consumption.

4.2.5 Protocol Summary

We summarize the packet forwarding algorithm in Fig. 5.

Upon receiving a packet, a node first checks if it is a qualified forwarder for the packet based on the depth information and the depth threshold d_{th}. If it is not a qualified forwarder, it searches the packet in $Q1$ and removes the packet with the same packet ID since another (better) node has already forwarded the packet. If the node is a qualified forwarder, it searches the packet in the packet history buffer $Q2$. If the packet is found in $Q2$, it is dropped as it has been forwarded recently. Otherwise, the node calculates the sending time for the packet based on the current system time and the holding time and inserts the packet into the priority queue $Q1$. Note that if the packet is already in $Q1$, the sending time is updated to the earlier

Algorithm 1 Algorithm for packet forwarding in DBR

```
ForwardPacket(p)
 1: Get previous depth d_p from p
 2: Get node's current depth d_c
 3: Compute Δd = (d_p – d_c)
 4: IF Δd < Depth Threshold d_th THEN
 5:    IF p is in Q1 THEN
 6:       Remove <p, Sending Time> from Q1
 7:    ENDIF
 8:    Drop p
 9:    return
10: ENDIF
11: IF p is in Q2 THEN
12:    Drop p
13:    return
14: ENDIF
15: Update p with current depth d_c
16: Compute Holding Time HT
17: Compute Sending Time ST
18: IF p is in Q1 THEN
19:    Get previous sending time of p ST_p
20:    Update p's Sending Time with min(ST, ST_p)
21: ELSE
22:    Add the item <p, ST> into Q1
23: ENDIF
```

Fig. 5 DBR packet forwarding algorithm

time. Later, the packets enqueued in $Q1$ will be sent out according to their scheduled sending times.

4.2.6 Simulation Results

All simulations are performed using the Network Simulator (ns2) [34] with an underwater sensor network simulation package (called Aqua-Sim) extension. In our simulations, sensor nodes are randomly deployed in a 500 m × 500 m × 500 m 3-D area. Multiple sinks are randomly deployed at the water surface. The sensor nodes follow the random-walk mobility pattern. Each sensor node randomly selects a direction and moves to the new position with a random speed between the minimal speed and maximal speed, which are 1 and 5 m/s respectively unless specified otherwise. For the communication between sensor nodes, we set the parameters similar to a commercial acoustic modem, LinkQuest UWM1000 [16]: the bit rate is 10 k bps; the maximal transmission range is 100 m (in all directions); and the power consumption in sending, receiving and idling mode are 2 w, 0.1 w, and 10 mw respectively.

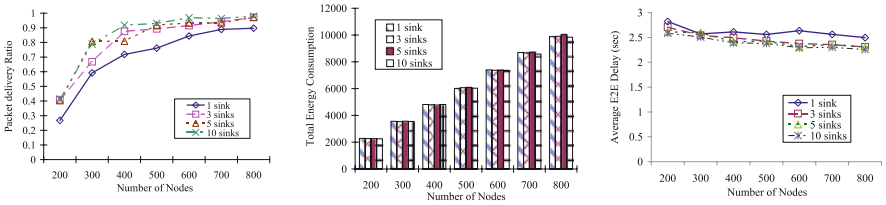


Fig. 6 DBR's performance with different number of surface sinks (gateways)

The same broadcast Media Access Control (MAC) protocol as in [37] is used in our simulations. In this MAC protocol, when a node has a packet to send, it first senses the channel. If the channel is free, it continues to broadcast the packet. Otherwise, it backs off. The packet will be dropped if the maximal number of backoffs have been reached.

Figure 6(a) shows how the packet delivery ratio changes with different number of surface sinks. DBR with multiple sinks has a better packet delivery ratio than DBR with only one sinks. Since DBR is a greedy algorithm trying to deliver data packets to the water surface, increasing the number of sinks at the water surface will increase the chance that a packet is received by a sink. This explains the higher delivery ratio when multiple sinks are deployed.

The total energy consumption for different number of sinks is shown in Fig. 6(b). We observe that the energy consumption is almost the same for different number of sinks. The reason is that in DBR, the number of sinks does not affect the forwarding process. Therefore, all different settings have almost the same energy consumption.

From Fig. 6(c), we observe that DBR with multiple sinks has a slightly better average end-to-end delay than DBR with one sink. This is because in the multiple-sink case, a packet is considered successfully delivered whenever it reaches any of the sinks.

In this example, we presented Depth Based Routing (DBR), a routing protocol based on the depth information of nodes, for UASN-MGs. DBR uses a greedy approach to deliver packets to the sinks at the water surface. Different from other geographical-based routing protocols that require full-dimensional location information of nodes, DBR only needs the depth information, which can be easily obtained locally at each sensor node. Further, DBR can naturally takes advantages of the multiple-sink underwater sensor network architecture without introducing extra cost.

4.3 Cross Layer Design

In this example, we will present a cross layer design scheme, **M**ulti-path **P**ower-control **T**ransmission (MPT), for time-critical applications in underwater sensor networks. MPT views the multiple surface gateways as one virtual sink node. Any underwater sensor nodes may find multiple routes to this virtual sink node. MPT takes full advantage of these routes for end-to-end reliable data transfer. Specifi-

cally, in MPT, multiple copies of a packet are transmitted along multiple routes to the virtual sink node, these copies are then combined at the destination. Distributed power control strategies at the physical layer are used to improve the overall energy efficiency. And there is no hop-by-hop retransmission, as contributes to low end-to-end delays.

MPT is divided into the following three parts: *multi-path routing*, *source initiated power-control transmission*, and *destination packet combining*.

4.3.1 Multi-Path Routing

We assume that some multi-path routing protocols such as those in [15, 18] are available. The basic procedure of multi-path routing is illustrated in Fig. 7. When the source node has some packets to send, it will flood a "Route Request" message to the destination. Any intermediate nodes who receive this "Route Request" for the first time will forward it. When the destination (the virtual sink node) receives "Route Request" messages, it will reply with "Route Reply" messages reversely along the paths of the corresponding "Route Request" messages. The destination can also make path selection. For example, it can select node-disjoint paths and send "Route Reply" back on them. After the source node receives the "Route Reply" messages, the routes between the source and the destination are established.

From the received "Route Reply" messages, the source node gets to know some path characteristics, such as the number of available paths, m, and the hop lengths of the paths. Based on this information, the source node will determine the optimal number of paths, m^*, and select m^* paths from m available paths. It also needs to calculate the optimal power level that every intermediate node on these paths should use for packet transmission.

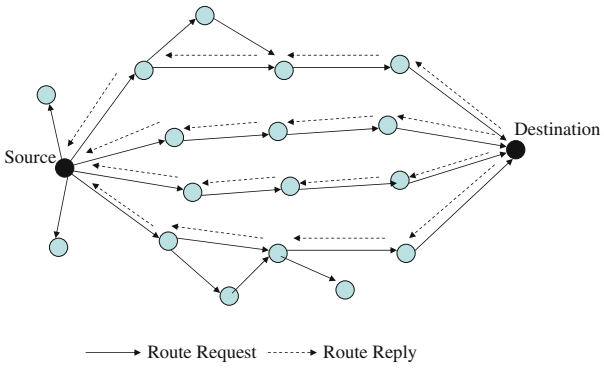


Fig. 7 Basic procedure of multi-path routing

4.3.2 Source Initiated Power-Control Transmission

In this phase of MPT, the same packets sent from the source node are transmitted by the intermediate nodes along all the selected paths using the specified transmission power. The packet format is shown in Fig. 8. Every packet should include the source identification (Source ID), the destination identification (Destination ID), as well as the packet sequence number (Sequence No.) in the packet header. The source node should also include power parameters in the packet header. These parameters specify the required power level that every intermediate node should use to relay the packet. In addition, we assume some coding schemes with strong error correction capability, such as forward error coding (FEC), are used in the header of every packet. In this way, the header part of every packet can be decoded correctly with high probability. Since the packet header is usually much smaller than the data part, the overhead incurred in the header error correction process is almost negligible. For the large data part, we do not use any error correction coding schemes because of their inefficiency in fading environments [7]. But some error detection coding schemes, such as cyclical redundant checking (CRC), are still used in the data part to check data errors.

When an intermediate node receives a packet, it will decode and check the header. If the header is correct or can be recovered by the adopted FEC scheme, this node relays the whole packet to its next hop with the specified power level without further checking the data part. Otherwise, it will simply drop the packet.

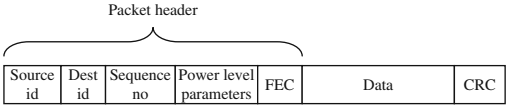


Fig. 8 Packet format

4.3.3 Destination Packet Combining

When a surface gateway receives a copy of the original packet from one path (the data part of this copy may be corrupted during the transmission process), it will forward this copy to the control center through the surface radio network. the control center first checks whether this copy is correct or not. If there is no error with this copy, it successfully receives the original packet. Otherwise, the control center will keep this corrupted copy in its buffer. After multiple corrupted copies of the original packet are received, the destination will combine them to recover the original one.

In MPT, we use a simple packet combining technique, which is illustrated in Fig. 9. Assuming the destination receives m^* copies of the same packet from m^* paths, for the i-th bit in this packet, it determines its output b_i as

$$b_i = \begin{cases} 1 & \sum_{k=1}^{m} b_{ik} > \dfrac{m^*}{2}, \\ 0 & \sum_{k=1}^{m} b_{ik} < \dfrac{m^*}{2}, \end{cases} \tag{14}$$

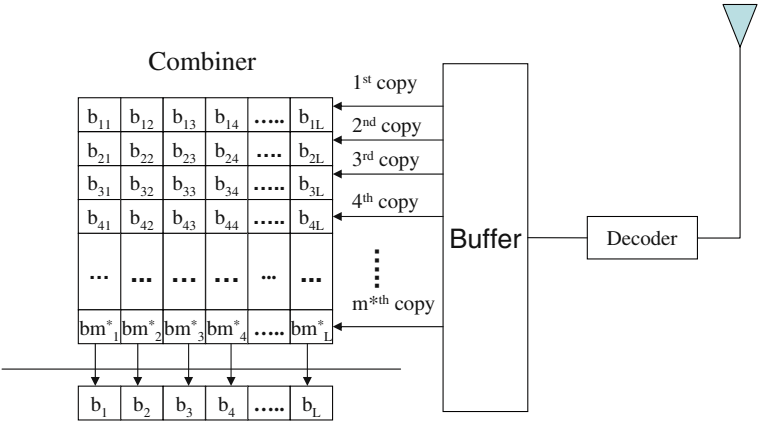


Fig. 9 Packet combining at the destination

where b_{ik} denotes the kth copy of the ith bit. To simply put, if the bits in the majority of the copies are "1", then the corresponding bit of the original packet is decoded as "1"; otherwise, it is decoded as "0". We choose this "majority voting" method mainly because of its simplicity. It should be noted that [7, 30] propose more complicated packet combining schemes, which can further reduce packet error rate. However, these schemes need to search through all detectable error patterns in order to recover the original packet, as will introduce significant processing delay and increase the complexity of the destination node, especially when the data packet is large. On the other hand, our simple "majority voting" scheme has a constant processing delay and is simple to implement. Our simulation results in Sect. 4.3.5 show that even this simple packet combining technology can still achieve significant performance improvement.

4.3.4 Optimal Energy Distribution

In MPT, given m available paths, n_i hops on path i ($1 \leq i \leq m$), and the required end-to-end packet error rate (PER), P_{req}, the source node needs to distribute transmission energy for each hop along the m path in order to minimize the overall energy consumption for one packet transmission. This problem can be formulated as follows:

$$\begin{aligned} \min \quad & \sum_{i\in(1,2,\ldots m)} \sum_{j\in(1,2,\ldots n_i)} E_{ij} L \\ \text{s.t.} \quad & P_e \leq P_{\text{req}}, \\ & 0 \leq E_{ij} \leq E_{\max}, \end{aligned} \tag{15}$$

where L is the packet length in bits; E_{ij} is the average transmitting energy per bit of the jth hop on the ith path; P_e is the average end-to-end packet error rate (PER); and $E_{\max}$ is the maximal transmitting energy per bit of every node, which is stipulated by the system hardware constraints.

In Eq. (15), the first constraint specifies that the average end-to-end PER, P_e, should be smaller than the system requirement P_{req}. The second constraint is to guarantee the transmitting energy per bit of every node will not exceed its maximal allowable transmitting energy per bit $E_{\max}$. In this paper, we ignore the energy consumption for data receiving and processing. This is because in underwater acoustic communication, data receiving and processing consumed much less energy than data transmitting [8].

The above optimization problem is hard to solve because of the following reasons: (1) with the import of the packet combining technique, the expression for the end-to-end PER, P_e, is quite complicated and not convex; (2) In (15), the large number $\left(\sum_{i\in(1\ldots m)} n_i\right)$ of involved variables leads high computation complexity of the source node.

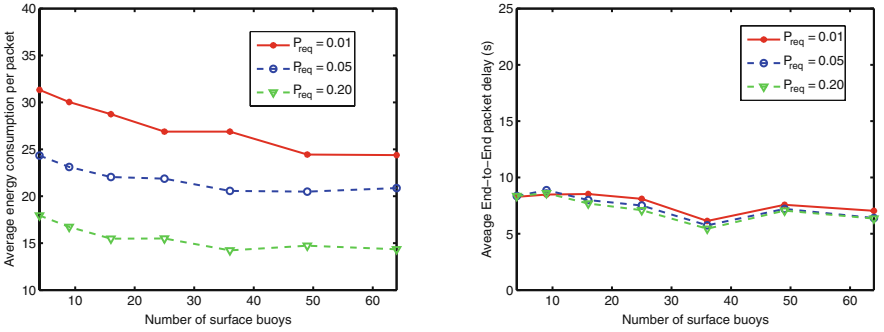
In order to solve this complicated optimization problem, we divide the solving process into two steps. In the first step, we do optimal energy distribution among all available paths. This optimal distribution needs to guarantee that the average end-to-end PER requirement can be satisfied. In the second step, we do optimal energy distribution among all nodes along each selected path.

4.3.5 Simulation Results

Based on NS-2, we implement a simulation package for underwater sensor networks.we modify AODV (Ad hoc On-Demand Distance Vector) [28] and make it support multiple path routing. We gradually change the number of surface buoys from 4 to 64. Figure 10(a) clearly shows that with the increase of surface buoys, the average overall energy consumption per packet will decrease. And the decreasing rate will slow down with the increase of surface buoy. This is because of the higher collision probability introduced by more surface buoys and more paths.

From Fig. 10(b), we can see that the end-to-end packet delay changes slowly with the number of surface buoys. This is because with more surface buoys, every node has higher probability to find shorter path to the surface buoy, which contributes to a shorter end-to-end packet delay. On the other side, with more surface buoys and more available paths, higher collision probability is introduced in the networks and thus more time is wasted in the collision resolution. Figure 10(b) shows us that in our simulation setting, when the surface buoys reach 36, the end-to-end delay reaches its minimum.

In this example, we have presented MPT, an energy efficient multi-path transmission scheme for time-critical services in underwater sensor networks. Our simulation study shows that MPT can achieve high energy efficiency with small end-to-end delay under certain end-to-end PER requirements.



(a)Averageenergyconsumptionperpacket (b)Averageend-to-endpacketdelay

Fig. 10 Performance with varying number of surface gateways

5 Conclusions

In this chapter, we discuss a new network architecture, UASN-MG, which has multiple surface gateways for underwater acoustic sensor networks. Two networks, an acoustic network formed by underwater sensor nodes and a surface radio network formed by the surface gateways, are involved in the architecture. UASN-MG brings about many benefits such as high energy efficiency and reliability. We have discussed the design challenges of UASN-MG and shown that UASN-MG has impacts on almost every layer of the protocol stack. To optimize the network's performance, We need to examine and re-consider all protocols. Three design examples are given in this chapter. The first example discuss the optimal deployment of surface gateways, which is formulated as an integer linear programming(ILP) problem to optimize the system's energy efficiency and end-to-end delay. In the second example, we discuss a new routing protocol, depth-based routing(DBR), which is well suitable for UANS-MGs. The third example presents a new cross layer design scheme, multipath power control transmission (MPT), which can provide reliable end-to-end data transfer with high energy efficiency and low end-to-end delay.

For future work in this area, we would like to suggest the following research directions. (1) Adaptive MAC protocols. In UASN-MG, different nodes play different roles in the network and have different importance to the networks. The more important nodes should have the higher priorities to access the communication channel in the MAC layer. Adaptive MAC protocols can change the node's priority according its role and its depth information in order to optimize the network's performance. To the best of our knowledge, no such adaptive MAC protocols have been proposed so far. (2) Combined optimal deployment of sensor nodes and surface gateways. To optimize the system performance, we should deploy the underwater sensor nodes and the surface gateway together. Such a combined deployment is complicated yet an interesting research topic. (3) Cross-layer optimization for UASN-MG. Multiple surface gateways greatly complicate the network design and bring about many issues which cannot be solved solely by a separated layer. A good

cross-layer design framework is imperative for the further development of efficient multi-gateway underwater communication systems.

References

1. Akyildiz, I.F., Pompili, D., Melodia, T.: Underwater acoustic sensor networks: Research challenges. Ad Hoc Networks (Elsevier) **3**(3), 257–279 (2005).
2. Capellano, V.: Performance improvement of a 50 km acoustic transmission through adaptive equalization and spatial diversity. In: Proceedings of Oceans, Nova Scotia, Canada (1997).
3. Capellano, V., Loubet, G., Jourdain, G.: Adaptive multichannel equalizer for underwater communications. In: Proceedings of Oceans, Ft. Lauderdale, FL, USA (1996).
4. Casari, P., Rossi, M., Zorzi, M.: Fountain codes and their application to broadcasting in underwater networks: Performance modeling and relevant tradeoffs. In: Proceedings of the 3rd ACM International Workshop on Underwater Networks (WUWNET), pp. 11–18 (2008).
5. Chitre, M., Shahabudeen, S., Stojanovic, M.: Underwater acoustic communication and networks: Recent advances and future challenges. Marine Technology Society Journal **42**(1), 103–116 (2008).
6. Cui, J. H., Kong, J., Gerla, M., Zhou, S.: Challenges: Building scalable mobile underwater wireless sensor networks for aquatic applications. IEEE Network, Special Issue on Wireless Sensor Networking **20**(3), 12–18 (2006).
7. Dubois-Ferriere, H., Estrin, D., Vetterli, M.: Packet combining in sensor networks. In: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems, pp. 102–115 (2005).
8. Freitag, L., Grund, M., Singh, S., Partan, J., Koski, P., Ball, K.: The WHIO micro-modem: An acoustic communications and navigation system for multiple platforms. In: Proceedings of MTS/IEEE OCEANS, Vol. 2, pp. 1086–1092 (2005).
9. Heidemann, J., Li, Y., Syed, A., Wills, J., Ye, W.: Research challenges and applications for underwater sensor networking. In: Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), pp. 228–235 (2006).
10. Ibrahim, S., Cui, J. H., Ammar, R.: Surface gateway deployment for underwater sensor networks. In: Proceedings of IEEE Military Communications Conference (MILCOM) (2007).
11. Jornet, J. M., Stojanovic, M., Zorzi, M.: Focused beam routing protocol for underwater acoustic networks. In: Proceedings of the 3rd ACM International Workshop on Underwater Networks (WUWNET), pp. 75–81 (2008).
12. Kaya, A., Yauchi, S.: An acoustic communication system for subsea robot. In: Proceedings of OCEANS, Vol. 3, pp. 765–770, September (1989).
13. Kilfoyle, D. B., Baggeroer, A. B.: The state of the art in underwater acoustic telemetry. IEEE Journal of Oceanic Engineering, **5**, 4–27 (2000).
14. Kredo, K. B. II, Mohapatra, P.: A hybrid medium access control protocol for underwater wireless networks. In: Proceedings of the 2nd ACM International Workshop on Underwater Networks (WUWNET), pp. 33–40 (2007).
15. Leung, R., Liu, J., Poon, E., Chan, A. L. C., Li, B.: MP-DSR: A QoS-aware multi-path dynamic source routing protocol for wireless ad-hoc networks. In: Proceedings of IEEE International Conference on Local Computer Networks. IEEE Computer Society, Los Alamitos, CA, USA (2001). DOI http://doi.ieeecomputersociety.org/10.1109/LCN.2001.990778.
16. Linkquest. http://www.link-quest.com/.
17. Liu, L., Zhou, S., Cui, J. H.: Prospects and problems of wireless communication for underwater sensor networks. Wireless Communications & Mobile Computing **8**, 977–994 (2008).
18. Marina, M., Das, S.: On-demand multi path distance vector routing in ad hoc networks. In: Proceedings of International Conference on Network Protocols (ICNP), pp. 0–14. IEEE Computer Society, Los Alamitos, CA, USA (2001). DOI http://doi.ieeecomputersociety.org/10.1109/ICNP.2001.992756.

19. Nicolaou, N., See, A., Xie, P., Cui, J. H., Maggiorini, D.: Improving the robustness of location-based routing for underwater sensor networks. In: Proceedings of MTS/IEEE OCEANS, pp. 1–6 (2007).
20. Park, M. K., Rodoplu, V.: UWAN-MAC: An energy-efficient MAC protocol for underwater acoustic wireless networks. IEEE Journal of Oceanic Engineering **32**(3), 710–720 (2007).
21. Partan, J., Kurose, J., Levine, B. N.: A survey of practical issues in underwater networks. In: Proceedings of ACM International Workshop on Underwater Networks (WUWNet), pp. 17–24 (2006). URL http://prisms.cs.umass.edu/brian/pubs/partan.wuwnet2006.pdf.
22. Peleato, B., Stojanovic, M.: MAC protocol for ad-hoc underwater acoustic sensor networks. In: Proceedings of the 1st ACM International Workshop on Underwater Networks, pp. 113–115 (2006).
23. Pompili, D., Melodia, T., Akyildiz, I. F.: Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks. In: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking (Mobicom), pp. 298–309 (2006).
24. Pompili, D., Melodia, T., Akyildiz, I. F.: A distributed CDMA medium access control for underwater acoustic sensor networks. In: Proceedings of Mediterranean Ad Hoc Networking Workshop (2007).
25. Prasath, A., Venuturumilli, A., Ranganathan, A., Minai, A.A.: A self-organizing heuristic for building optimal heterogeneous ad-hoc sensor networks. In: Proceedings of the IEEE International Conference on Networking, Sensing and Control, pp. 774–779 (2006).
26. Rodoplu, V., Gohari, A. A., Tang, W.: Towards automated design of MAC protocols for underwater wireless networks. In: Proceedings of the 3rd ACM International Workshop on Underwater Networks (WUWNET), pp. 67–74 (2008).
27. Rodoplu, V., Park, M. K.: An energy-efficient MAC protocol for underwater wireless acoustic networks. In: Proceedings of MTS/IEEE OCEANS, Vol. 2, pp. 1198–1203 (2005).
28. Royer, E. M., Perkins, C. E.: Ad-hoc on-demand distance vector routing. In: Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, Vol. 2, pp. 90–100 (1999).
29. Salva-Garau, F., Stojanovic, M.: Multi-cluster protocol for ad hoc mobile underwater acoustic networks. In: Proceedings of the IEEE Oceans Conference, pp. 91–98 (2003).
30. Sindhu, P.S.: Retransmission error control with memory. IEEE Transactions on Communications **25**(5), 473–479 (1977).
31. Suzuki, M., Nemoto, K., Tsuchiya, T., Nakanishi, T.: Digital acoustic telemetry of color video information. In: Proceedings of OCEANS, pp. 693–696, September (1989).
32. Syed, A.A., Ye, W., Heidemann, J.: T-Lohi: A new class of MAC protocols for underwater acoustic sensor networks. In: Proceedings of Infocom (2008).
33. Tan, H.X., Seah, W. K. G.: Distributed CDMA-based MAC protocol for underwater sensor networks. In: Proceedings of the 32nd IEEE Conference on Local Computer Networks, pp. 26–36 (2007).
34. The ns mannual. http://www.isi.edu/nsnam/ns/doc/index.html (2002).
35. Xie, P., Cui, J. H.: SDRT: A reliable data transport protocol for underwater sensor networks. UCONN CSE Technical Report: UbiNet-TR06–03 (BECAT/CSE-TR-06–14) (2006).
36. Xie, P., Cui, J. H.: R-MAC: An energy-efficient MAC protocol for underwater sensor networks. In: Proceedings of International Conference on Wireless Algorithms, Systems, and Applications (WASA) (2007).
37. Xie, P., Lao, L., Cui, J. H.: VBF: Vector-based forwarding protocol for underwater sensor networks. In: Proceedings of IFIP Networking (2006).
38. Yan, H., Shi, Z., Cui, J. H.: DBR: Depth-based routing for underwater sensor networks. In: Proceedings of IFIP Networking, pp. 1–13 (2008).
39. Zhou, Z., Cui, J. H.: Energy efficient multi-path communication for time-critical applications in underwater sensor networks. In: Proceedings of ACM MOBIHOC, pp. 221–230 (2008).

Communication Through Soil in Wireless Underground Sensor Networks – Theory and Practice

M. Can Vuran and Agnelo R. Silva

Abstract Wireless Underground Sensor Networks (WUSNs) constitute one of the promising application areas of the recently developed wireless sensor networking techniques. WUSN is a specialized kind of WSN that mainly focuses on the use of sensors at the subsurface region of the soil, that is, the top few meters of the soil. For a long time, this region has been used to bury sensors, usually targeting irrigation and environment monitoring applications, although without wireless communication capability; WUSNs promise to fill this gap and to provide the infrastructure for novel applications. The main difference between WUSNs and the terrestrial WSNs is the communication medium. In fact, the differences between the propagation of electromagnetic (EM) waves in soil and in air are so significant that a complete characterization of the underground wireless channel was only available recently. This chapter presents advanced channel models that were developed to characterize the underground wireless channel considering the characteristics of the propagation of EM waves in soil and their relation with the frequency of these waves, the soil composition, and the soil moisture. Additional important aspects such as the burial depth, the reflection, the refraction, and multi path fading effects on the EM waves are also considered. The results from the field experiments in conjunction with the simulation results, both considering the path loss and the bit error rate, prove the feasibility of WUSNs. The chapter concludes with an outlook on potential research topics that are essential for the realization of WUSNs.

1 Introduction

Wireless Underground Sensor Networks (WUSNs), which consist of wireless sensors buried underground, are a natural extension of the wireless sensor network phenomenon and have been considered as a potential field that will enable a wide variety of novel applications that were not possible before. Compared to the

M.C. Vuran (✉)
Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, USA
e-mail: mcvuran@cse.unl.edu

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_12,

current underground sensor networks, which use wired communication methods for data retrieval, WUSNs have several remarkable merits, such as concealment, ease of deployment, timeliness of data, reliability and coverage density [2]. The realization of wireless underground communication and networking techniques will lead to potential applications in the fields of *agriculture, border patrol, assisted navigation, sports field maintenance, intruder detection*, and *infrastructure monitoring*. Many of these novel applications will be possible because WUSNs can provide localized and real-time data about a specific soil region and its surrounding area.

Despite its potential advantages, several open research problems exist to make WUSNs feasible. The main challenge is the realization of efficient and reliable underground wireless communication between buried sensors. Aspects such as temperature, weather, soil composition, soil moisture, and soil homogeneity directly impact the connectivity and communication success. In fact, underground communication is one of the few fields where the environment has a significant and direct impact on the communication performance. In addition to these factors, the burial depth and the frequency of the EM wave also have strong effect over the wireless underground communication. Hence, characterization of the wireless underground channel is essential for the proliferation of communication protocols for WUSNs.

In this chapter, we introduce theoretical models to characterize the underground wireless channel. The models developed in this chapter characterize not only the propagation of EM wave in soil, but also other effects on the communication related to multi path effects, soil composition, water content, and burial depth. The results obtained from this formalization reveal that underground communication is severely affected by frequency and soil properties, and more specifically by the volumetric water content (VWC) of soil. Moreover, the effects of weather and season changes are investigated by considering two soil types as examples. Such theoretical models are essential for laying out the foundations for efficient communication in this environment. In particular, the 300–900 MHz frequency band, which is suitable for small size antenna and sensor development, is investigated and the results of field experiments realized at 433 MHz are compared with the theoretical models. Moreover, the realization of field experiments also revealed important issues not considered in the theoretical models, such as the effects of the antenna orientation. Finally, challenges for the feasibility of WUSNs are highlighted.

The rest of this chapter is organized as follows. In Sect. 2, an overview on the wireless underground communication is provided along with a classification for the underground networks. In Sect. 3, some recent WSN solutions related to the underground enviroment are presented. The propagation characteristics of 300–900 MHz EM waves in soil are analyzed in Sect. 4.1. In Sects. 4.2 and 4.3, the characteristics of the underground channel in soil are modeled taking in consideration the effects of the reflection of surface as well, the multi path fading, and the volumetric water content in soil. The results of the underground-to-underground communication experiments, at 433 MHz, are presented in Sect. 5. Then, in Sect. 6, the challenges in WUSN design are summarized according to the outcomes from the simulations based on the theoretical model and also from the experimental results.

2 Classification of Underground Communication Networks

As shown in Fig. 1, Wireless Underground Communication Networks (WUCNs) can be mainly classified into two: wireless communication networks for mines and tunnels and wireless underground sensor networks (WUSNs). There exist several solutions that focus on underground communication in mines and/or tunnels [7, 16, 21, 32]. In this context, although the network is located *underground*, the communication takes place *through the air*, i.e., through the voids that exist underground. Consequently, even though the communication in these voids are more challenging than that in terrestrial WSNs, the channel characteristics exhibit similarities with the terrestrial WSNs.

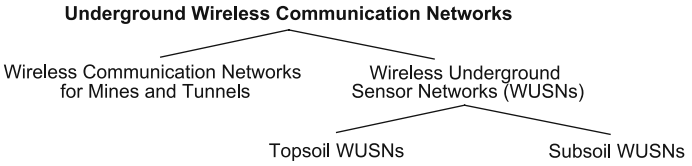


Fig. 1 Classification of wireless underground communication networks (WUCNs)

This chapter focuses on WUSNs, which have components, i.e. the sensors, that are buried underground and that communicate through soil. The majority of the applications for WUSNs – intelligent agriculture, environmental monitoring, and security – restricts the buried sensors at the *subsurface* of the soil, which is defined as the top few meters of the soil. This region of the soil has generally been classified into two regions [17]:

- The *topsoil region*, which refers to the first 30 cm of soil, or the root growth layer, whichever is shallower.
- The *subsoil region*, which refers to the region below the topsoil, i.e., usually the 30-100 cm region.

The characteristics of the soil for the topsoil and subsoil regions not only present different agricultural characteristics but also present distinct scenarios for WUSNs applications due to two main reasons:

- These soil regions may be distincts in terms of soil texture and water content [17], which are significant parameters that affect the wireless communication channel, as we will see in Sect. 4. Moreover, the nodes buried at the topsoil region may take better advantage of the reflection effect from the surface of ground.
- The plowing and similar mechanical activities occur exactly at the topsoil region. Hence, for certain agriculture applications, such as crop irrigation management, the burial depth must be higher than the topsoil region, which may be different for other applications, such as border patrol. In other words, for certain WUSN applications, the sensors must be buried at the subsoil region.

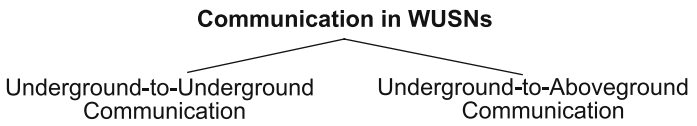


Fig. 2 Classification of communication in WUSNs

Accordingly, as shown in Fig. 1, WUSNs can be classified as a function of the deployment region: *Topsoil WUSN*, if the WUSN is deployed in the topsoil region, or *Subsoil WUSN*, if deployed in the subsoil region.

In addition to nodes buried underground, the existence of aboveground nodes is also necessary for WUSN due to data retrieval purposes. Accordingly, two communication types coexist in WUSNs as shown the Fig. 2. *Underground-to-underground* communication refers to the information exchange between buried sensors for network management and data relay purposes. The data in the network is then collected through *underground-to-aboveground* communication. This type of communication also includes command and control information from aboveground stations to underground. The classification in Fig. 2 also highlights significant differences on the challenges related to each option. Underground-to-aboveground communication presents better quality compared to underground-to-underground since certain portion of the communication takes place over the air [30]. On the other hand, underground-to-underground communication usually presents more difficult challenges for the design and the deployment of WUSNs. In this chapter, we focus on underground-to-underground communication for WUSNs.

3 Recent Developments

In this section, recent developments related to wireless communication for the underground environment are presented. The majority of such developments are not actually WUSN applications or, at least, do not present the same challenges discussed in this chapter. However, the following examples supply an overall vision about the challenges in this area and also present potential applications for WUSNs.

Wireless sensor networks have been used to monitor underground mines to guarantee the safety of mine workers [8, 16]. Similarly, the characteristics of the wireless channel in tunnels are been investigated [4, 32]. As mentioned in Sect. 2, although the mine is underground, the communication among the sensors is through the air in the mine or tunnel.

A shallow depth WSN was used for predicting landslides [28]. This network consists of MICA2 motes [10] that are interfaced with stain gauges and can operate at low depths (25–30 cm). In this design, although the sensors are buried underground, the communication takes place over the air. Another similar example is a sensor network that is constructed to detect the volcano activities. The antenna of the sensors has to be placed above ground to create reliable links [38].

Structural health monitoring (SHM) is another application that has gained interest in wireless sensor networks community. Two examples of such WSN application is Wisden, a data acquisition system for SHM [24, 40] and Duranode [25]. Although underground systems such as sewers also require structural monitoring, these approaches only work with communication through air techniques.

The largest residential water management project in Europe uses sensors to gather information for inspection and cleaning systems in the Emscher sewer system [12]. Similarly, a sensor network is used in other sewer system where the manhole cover is converted into the slot antenna and the *underground* sensors can communicate with the aboveground nodes through radiation from it [21]. Again, although the system resides *underground*, the communication between the sensors is performed *through air*.

A glacier monitoring network, based on sensor network, was deployed in Norway [20]. This system aims to measure the parameters of ice caps and glaciers using sensors beneath the glaciers. To avoid wet ice, the base stations are connected to two wired transceivers 30 m below the surface. Using high transmit powers (100 mW), these under*ice* sensors can finally communicate with the sensors that are placed at deeper locations (up to 80 m from the surface). This application is not a typical underground scenario, however it presents challenges similar to the ones for WUSNs.

In addition to these applications, there are several experimental work focusing on the EM wave propagation through soil and rock. As part of the first studies related to the wireless communication through soil, the electromagnetic field principles of a vertical electric dipole in a conducting half-space over the frequency range from 1 to 10 MHz is analyzed [35]. Similarly, the communication through soil is regarded as an electromagnetic wave transfer through the transmission line and experiments at the frequency range of 1–2 GHz are realized, providing a propagation model [37].

Moreover, experiments using ground-penetrating radars were performed [11, 22, 35, 37]. This specific research area is called *Microwave Remote Sensing* [6, 9] and part of the theoretical model presented in Sect. 4 is based on the results from this research area. As an example of the use of the principles of the surface-penetrating radar, an experiment was done for determining the attenuation and relative permittivity values of various materials, including soil, at 100 MHz [11]. A typical application for Microwave Remote Sensing is the detection of landmines based on differences between the dielectric constants of soil and the landmine. For instance, it was shown that the soil composition has significant effects on the Ground Penetrating Radar (GPR) detection of landmines [22].

Although significant insight in EM wave propagation through soil can be gathered from these works, none of the existing work provides a complete characterization of underground communication. More specifically, neither the channel characteristics nor the multi path effects due to obstacles in soil or the nonhomogeneous feature of soil have been analyzed before. In the following section, these important issues are addressed aiming the establishment of the foundations for underground wireless networks.

4 Underground Channel Model: The Theory

In this section, a brief introduction about the physical properties of the soil is presented. Also the effects of these physical properties on the underground wireless communication are explained. The soil is a *dielectric material*, characterized by a *dielectric constant*. The propagation of EM waves is directly related to the dielectric constant of the material. More specifically, a smaller value of the dielectric constant basically implies better conditions for the propagation of EM waves. The soil medium behaves as a dielectric material composed of air, *bound* water, *free* water, and bulk soil. If the soil presents small density and high porosity, the performance of the propagation of EM waves is better due to the high quantity of air. However, the presence of water in soil has a contrary effect on the communication. The quantity of water in the soil, which is usually measured as the volumetric water content (VWC) of the soil, is the main factor that contributes to the EM wave attenuation [19].

The dielectric constant of the soil varies as a function of its components [26]. Soil composition is generally classified in terms of the percent of sand, clay, and silt, as shown in Fig. 3. Depending on the amount of clay, silt, and sand, the soil texture receives a particular name or classification [13]. For instance, the point *P* in Fig. 3 represents a soil texture with a homogeneous mixture of clay, silt, and sand, and it is classified as *Clay loam*.

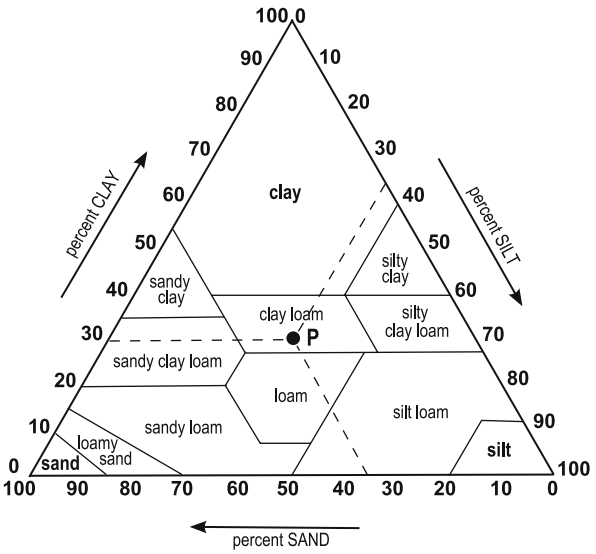


Fig. 3 Soil texture triangle showing the United States Department of Agriculture (USDA) classification system based on grain size (clay: < 0.002 mm, silt: 0.002–0.050 mm, sand: 0.050–2.0 mm). Point P represents a soil that contains 35% sand, 35% silt, and 30% clay, and is a clay loam. The soil texture is one of the factors that affect the wireless underground communication

Besides clay, silt, and sand, the soil also contains water. The VWC of the soil represents the fraction of water in the soil. However, the water can be classified into two: the *bound water*, which corresponds to water molecules tightly held to the surface of the soil particles, and the *free water*, which corresponds to water molecules *free* of action of soil particles [6, 9]. As a result, for the same VWC, a sample of soil can contain more free water depending on the soil texture. More specifically, the quantity and also the type of clay determine the amount of bounded water in the soil. As will be explained later in Sects. 4.1 and 5.5, the dielectric properties of the bounded water are different from the free water. Therefore, the attenuation of the EM waves and the change in the signal velocity varies as a function of the amount of bounded water and the amount of free water. In this way, it is clear that the VWC parameter alone is not sufficient to characterize the propagation of EM waves, but the parameters related to the soil texture are also necessary to complement the VWC information.

Besides the VWC, the frequency of the signal also affects the propagation of the EM signal through the soil. It is well known that the dielectric constant of the soil is a non-linear function of the frequency. In other words, it is not simple to characterize the EM signal propagation as a function of frequency. Therefore, many studies were performed, for different frequency ranges, in order to determine the effects in the signal propagation for a specific frequency range. It was shown that frequency values around 1 GHz present reasonable soil dielectric constant values for wireless communication and microwave remote sensing applications [26, 19, 15]. Frequency values smaller than 300 MHz could result in smaller attenuation for the EM signal. However, when the frequency decreases, its wavelength increases and the size of antenna would also increase. Hence, usually the use of frequency values smaller than 300 MHz for WUSN scenarios is impractical.

The main difference between the well established techniques in terrestrial wireless sensor networks and WUSNs is the communication medium, which prevents a straightforward characterization of underground wireless channel. The main challenges related to the use of soil as a communication medium are summarized below:

- *Attenuation*. EM waves encounter much higher attenuation in soil compared to air and this fact severely hampers the communication quality. As an example, efficient communication between sensors nodes above and below ground is shown to be possible only at the distance of 0.5 m when the 2.4 GHz frequency is used [30].
- *Reflection*. The ground surface may cause the reflection effect, which can have positive or negative effects over the communication.
- *Multi path fading*. Unpredictable obstacles in soil such as rocks and roots of trees make EM waves being refracted and scattered also cause problems in the communication.

Since underground communication and networking are primarily affected by the wireless channel capabilities, advanced models and techniques are necessary to completely characterize the underground wireless channel and lay out the foun-

dations for efficient communication through soil. In Sect. 4.1, a basic model for the underground wireless channel that mainly considers the frequency of the EM waves, the soil composition, the soil density, and static values for VWC is present. In Sect. 4.2, the previous model is enhanced with the effects of multi path fading and reflection from ground surface. Finally, the effects of VWC variations in soil are considered in Sect. 4.3.

4.1 Signal Propagation Through Soil

The study of the propagation of EM waves through the soil begins with the basic model of the propagation of EM waves over-the-air, followed by the addition of the path loss factor specifically considering the properties of soil. From Friis equation [29], it is well known that the received signal strength (RSS) in free space at a distance r from the transmitter is expressed in logarithmic form as

$$P_r = P_t + G_r + G_t - L_0 \ , \tag{1}$$

where P_t is the transmit power, G_r and G_t are the gains of the receiver and transmitter antennae, and L_0 is the path loss in free space in dB, which is given by

$$L_0 = 32.4 + 20 \ \log(d) + 20 \ \log(f) \ . \tag{2}$$

where d is the distance between the transmitter and the receiver in kilometers, and f is the operation frequency in Mhz.

For the propagation in soil, an additional factor should be included in Friis equation (1) due the attenuation of the EM wave caused by the soil medium. As a result, the received signal is expressed as [19]:

$$P_r = P_t + G_r + G_t - L_0 - L_s \ , \tag{3}$$

where L_s stands for the additional path loss caused by the propagation in soil, which is calculated by considering the following two main differences of EM wave propagation in soil compared to that in air:

- The signal velocity, and hence, the wavelength λ, is different.
- The amplitude of the wave will be attenuated according to the frequency.

Therefore, the additional path loss, L_s, in soil is composed of two components:

$$L_s = L_\beta + L_\alpha \ , \tag{4}$$

where L_β is the attenuation loss due to the difference of the wavelength of the signal in soil, λ, compared to the wavelength in free space, λ_0, and L_α is the transmission loss caused by attenuation with attenuation constant α. Consequently, $L_\beta = 20 \log(\lambda_0/\lambda)$ and $L_\alpha = e^{2\alpha d}$.

Considering that in soil, the wavelength is $\lambda = 2\pi/\beta$ and in free space $\lambda_0 = c/f$, where β is the phase shifting constant, $c = 3 \times 10^8$ m/s, and f is the operating frequency in Hz, then, L_β and L_α can be represented in dB as follows:

$$L_\beta = 154 - 20\ \log(f) + 20\ \log(\beta)\ ,\ L_\alpha = 8.69\alpha d\ . \tag{5}$$

Knowing that the path loss in free space is $L_0 = 20\ \log(4\pi d/\lambda_0)$, finally, we have the main formula for the underground channel that expresses the attenuation caused by the soil medium. The path loss, L_p, of an EM wave in soil is as follows (after the conversion of the units kilometers and MHz in L_0 to meters and Hz, respectively):

$$L_p = 6.4 + 20\ \log(d) + 20\ \log(\beta) + 8.69\alpha d\ , \tag{6}$$

where distance, d, is given in meters, the attenuation constant, α, is in 1/m and the phase shifting constant, β, is in radian/m.

Note that the path loss, L_p, in (6) depends on the attenuation constant, α, and the phase shifting constant, β. The values of these parameters depend on the dielectric properties of soil which are based on the Peplinski semi-empirical soil dielectric model [26]. The specific model presented below is related to dielectric properties of soil in the 0.3–1.3 GHz band.

Using the Peplinski's principle [26], the dielectric properties of soil can be calculated as follows:

$$\varepsilon = \varepsilon' - j\varepsilon''\ , \tag{7}$$

$$\varepsilon' = 1.15\left[1 + \frac{\rho_b}{\rho_s}\left(\varepsilon_s^{\alpha'} - 1\right) + m_v^{\beta'}\varepsilon_{f_w}'^{\alpha'} - m_v\right]^{1/\alpha'} - 0.68,\quad \varepsilon'' = \left[m_v^{\beta''}\varepsilon_{f_w}''^{\alpha'}\right]^{1/\alpha'}, \tag{8}$$

respectively, where ε is the relative complex dielectric constant of the soil-water mixture, m_v is the volumetric water content of the mixture, ρ_b is the bulk density in grams per cubic centimeter, $\rho_s = 2.66\,\mathrm{g/cm^3}$ is the specific density of the solid soil particles, $\alpha' = 0.65$ is an empirically determined constant, and β' and β'' are empirically determined constants, dependent on soil-type and given by

$$\beta' = 1.2748 - 0.519S - 0.152C\ ,\quad \beta'' = 1.33797 - 0.603S - 0.166C\ , \tag{9}$$

where S and C represent the mass fractions of sand and clay, respectively. The quantities ε'_{f_w} and ε''_{f_w} are the real and imaginary parts of the relative dielectric constant of free water. Note that, at this point of the model, the influences of free water and bounded water are both considered in the above formula. The mass fractions of sand and clay considered in (9) and also the volumetric water content m_v are used to determine the amount of free water and bounded water in the soil. This distinction is important because the amount of free water causes a stronger attenuation effect for EM waves propagation when compared with the effects of the bounded water.

The Peplinski principle [26] governs the value of the complex propagation constant of the EM wave in soil, which is given as $\gamma = \alpha + j\beta$ with

$$\alpha = \omega \sqrt{\frac{\mu\varepsilon'}{2}\left[\sqrt{1+\left(\frac{\varepsilon''}{\varepsilon'}\right)^2}-1\right]}, \quad \beta = \omega \sqrt{\frac{\mu\varepsilon'}{2}\left[\sqrt{1+\left(\frac{\varepsilon''}{\varepsilon'}\right)^2}+1\right]}, \quad (10)$$

where $\omega = 2\pi f$ is the angular frequency, μ is the magnetic permeability, and ε' and ε'' are the real and imaginary parts of the dielectric constant as given in (8), respectively. Consequently, the path loss, L_p, in soil can be found by using equations (7), (8), (9), and (10) in (6).

The analysis of the above equations shows that the complex propagation constant and hence, the path loss of the EM wave in soil, are dependent on the following factors:

- *Operating frequency*, f, which is, the chosen frequency for the sensor nodes;
- *Composition of soil* in terms of sand and clay fractions, S and C, which depend on the deployment region of the sensor nodes;
- *Bulk density,* ρ_b, indirectly expressing the amount of air in the soil, which also depends on the deployment region of the sensor nodes;
- *Soil moisture or volumetric water content (VWC)*, m_v, which depends on the deployment region as well as time.

The path loss shown in (6) is evaluated using MATLAB and the results are shown in Fig. 4. Table 1 shows the input parameters for this evaluation, which reflect a typical soil condition [2, 26].

Table 1 Parameters used in the model evaluation in Fig. 4

Inter-node distance	Sand	Clay	Bulk density	Particle density
3 m	50%	15%	1.5 g/cm^3	2.66 g/cm^3

The operating frequency is chosen between 300 and 900 MHz. The choice for this smaller frequency range is due to the recent experiments for underground communication using MICAz nodes. These motes operate in the 2.4 GHz band and the results show that the communication range can be extended only up to 0.5 m at this band [30]. Hence, lower frequency bands are necessary for feasible WUSN solutions. However, decreasing operating frequency below 300 MHz also implies that the size of the antenna will increase, which can also prevent practical implementation of WUSNs. Considering that recent MICA2 motes operate in the 400–500 MHz range, while still preserving small antenna sizes, the operating frequencies between 300 and 900 MHz are interesting for our theoretical investigation.

The difference between propagation in soil and that in free space is shown in Fig. 4. The path loss, L_p, which is given in (6), is shown in dB versus operating frequency, f, for different values of VWC. The inter-node distance is fixed ($d = 3$m).

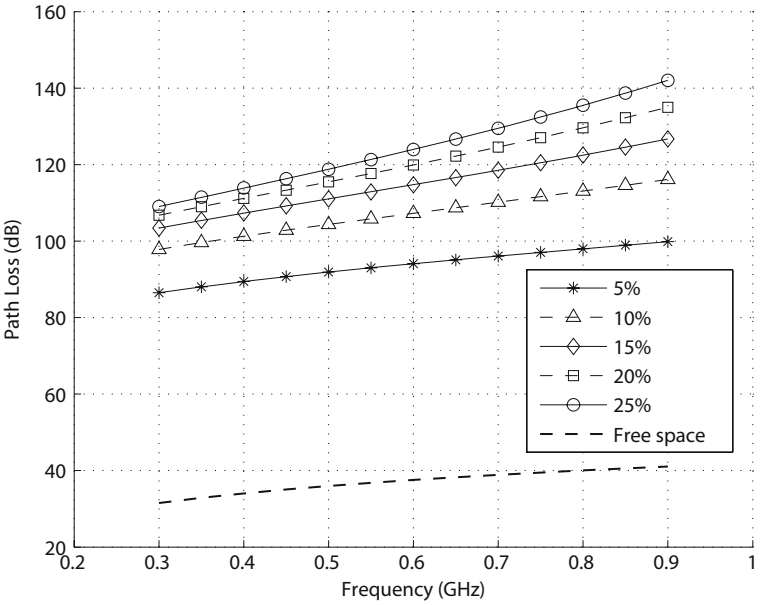


Fig. 4 Path loss vs. operating frequency and volumetric water content

This evaluation shows that, as the operating frequency, f, increases, the path loss also increases. This result motivates the use of lower frequencies for underground communication.

Figure 4 also highlights the effects of volumetric water content on the path loss, L_p. Since the attenuation significantly increases with higher water content, an increase of $\sim$ 30 dB is possible with a 20% increase in the volumetric water content of the soil. This effect is particularly important since water content not only depends on the location of the network but it also varies during different seasons as will be investigated in Sect. 4.3.

4.2 Underground Channel Characteristics

In Sect. 4.1, a basic model for the wireless underground channel has been presented and this model does not take into account the effects of the burial depth and the heterogeneity in soil. Moreover, besides the attenuation in soil, reflection from the ground surface and multi path spreading and fading effects also influence the performance of the wireless underground communication. In order to provide a complete characterization of the wireless channel in soil, this section adds the mentioned missing aspects to the original model. First, the effect of reflection from the ground surface on path loss is analyzed in Sect. 4.2.1. Second, the multi path effects are characterized by using a Rayleigh channel model and the bit error rate for underground wireless channel is derived in Sect. 4.2.2.

4.2.1 Reflection from Ground Surface

For WUSNs, which have the nodes buried at the first few meters layer of the soil, the underground communication can be visualized as a composition of two main paths for signal propagation as shown in Fig. 5. The first path is the direct path between the nodes and the second path is the reflection path due to the ground surface. While the direct path constitutes the main component of the received signal, the reflected path also affects communication.

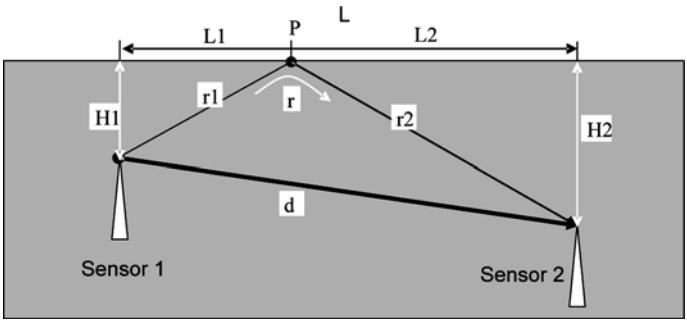


Fig. 5 Illustration of the two-path channel model

Naturally, the reflection effect depends on the burial depth of the nodes. If we continuously increase the burial depth, there is a point where the effect of reflection can be neglected and the channel can be considered as a single path. This occurs because having the reflected path a greater length than the direct path, its soil attenuation prevents it to arrive at the receiver node with enough strength. In this case, only the direct path is considered. Hence, the path loss is given in (6) as investigated in Sect. 4.1.

However, if the sensors are buried near the surface of ground, the influence of the wave reflection by ground surface should be considered. It occurs because the reflected waves are received with enough strength at the receiver, in addition to the directed waves. This scenario is usually the case for the Topsoil WUSNs applications with the burial depth generally varying from 0 to 30 cm. In this case, the total path loss of two-path channel model can be expressed as follows:

$$L_f(dB) = L_p(dB) - V_{dB} \ , \tag{11}$$

where L_p is the path loss due to the single path given in (6) and V_{dB} is the attenuation factor due to the second path in dB, i.e., $V_{dB} = 10 \ \log \mathrm{V}$.

Consider the case where the sensors are buried as illustrated in Fig. 5. The burial depths are H_1 and H_2, the horizontal inter-node distance is L, and the actual inter-node distance is d. From electromagnetic principles, we can derive the attenuation factor, V, as follows [4]:

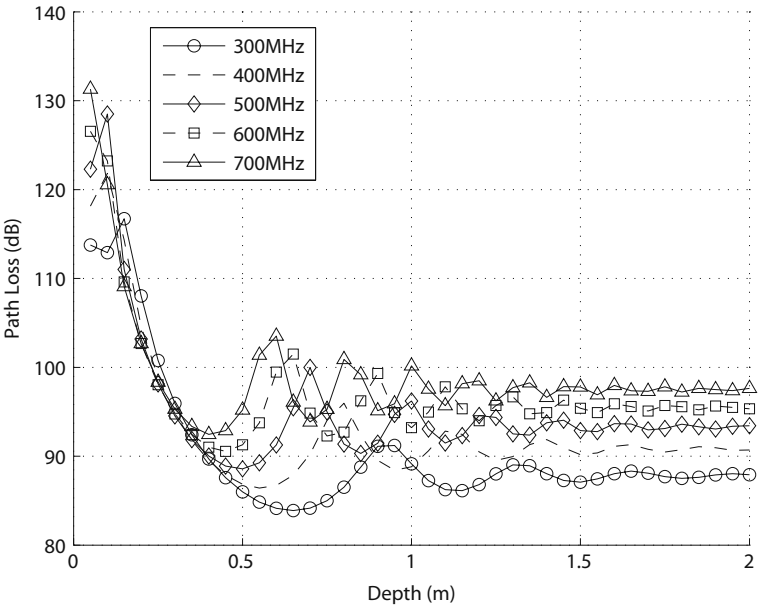


Fig. 6 Two-path channel model: path loss vs. depth for different operating frequencies

$$V^2 = 1 + (\Gamma \cdot \exp(-\alpha \Delta(r)))^2 - 2\Gamma \exp(-\alpha \Delta(r)) \cos\left(\pi - \left(\phi - \frac{2\pi}{\lambda}\Delta(r)\right)\right), \tag{12}$$

where, Γ and ϕ are the amplitude and phase angle of the reflection coefficient at the reflection point P, $\Delta(r) = r - d$, is the difference of the two paths, λ is the wavelength in soil, and α is the attenuation as given in (10).

In Figs. 6 and 7, the path loss is shown as a function of depth, H, operating frequency, and volumetric water content. The evaluations are using (12) in (11). In these figures, we assume that the sensors are buried at the same depth, i.e., $H_1 = H_2 = H$ and hence $d = L$. Table 2 shows the input parameters for this evaluation.

Table 2 Parameters used in the model evaluation in Fig. 6

Inter-node distance	Sand	Clay	Bulk density	Particle density	VWC
3 m	50%	15%	1.5 g/cm^3	2.66 g/cm^3	5%

In Fig. 6, the path loss is shown as a function of burial depth, H, for various operating frequency, f. It can be observed that for the two-path model, the operating frequency, f, has a higher impact on the communication channel and the degree of the effect depends on the bury depth, H. It occurs because the operating frequency f indirectly appears, as parameter λ, in the formula for the attenuation factor, V, in (12).

As shown in Fig. 6, for a particular operating frequency, it is possible to determine what is the optimum burial depth where the path loss is minimized. This result is particularly important for the topology design of WUSNs, where the burial depth of the nodes could be tailored to the operating frequency of these devices. In Fig. 6, it can also be observed that the effects of reflection, and hence, the fluctuations in path loss, diminish as the bury depth, H, increases. More specifically, the underground channel exhibits a single-path characteristic when the burial depth is higher than a threshold value. As shown in Fig. 6, this threshold value is near 2 m. However, this value of 2 m is specific for the parameters used in this evaluation and different values can be provided by the evaluation of the model with different parameters, such as VWC and soil composition. It is important to observe that, considering the usual burial depth proposed by many Subsoil WUSNs, it is expected the adoption of the two-path channel model for all the Topsoil WUSNs and the majority of the Subsoil WUSNs.

Comparing the results of this model in Fig. 6 with the single-path model results for the inter-node distance of 3 m, shown in Fig. 4, it is observed that the two path model presents a slightly smaller path loss, despite the fluctuations based on depth and volumetric water content. These results highlights that the reflection effects are generally positive for the underground communication.

The effects of frequency and the volumetric water content are shown in Fig. 7 and the input parameters are listed in Table 3. As shown in Fig. 7, the path loss fluctuates according to both operating frequency and volumetric water content. This fluctuation is due to the constructive or destructive interference of the second path based on the operating frequency and the VWC. This result suggests that dynamic

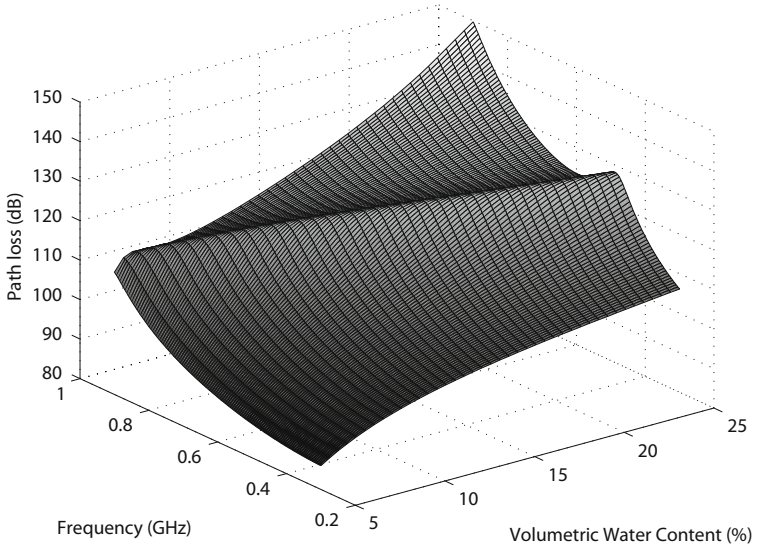


Fig. 7 Two-path channel model: path loss vs. operating frequencies and VWC

Table 3 Parameters used in the model evaluation in Fig. 7

Inter-node distance	Sand	Clay	Bulk density	Particle density	Depth ($H1 = H2$)
3 m	50%	15%	1.5 g/cm^3	2.66 g/cm^3	0.5 m

frequency operation may be necessary in WUSNs, where the operating frequency could be determined as a function of changes on the volumetric water content. Cognitive radio techniques [1] may be used to allow the adaptation of the WUSN to such changes normally related to changes on the environmental conditions.

4.2.2 Multi Path Fading and Bit Error Rate

In Sect. 4.2.1, it was presented an enhanced model for the wireless underground channel that considers the main propagation characteristics of EM waves and also the effects from the reflected waves. Moreover, the reflection is an aspect that depends on the burial depth of the nodes. However, the heterogeneity of soil can still interfere with the communication channel in two ways. First, the surface of the ground is not smooth and, hence, not only causes reflection, but also refraction. Second, the presence of rocks and roots of plants in soil is generally not homogeneous and results in non-deterministic behavior. Therefore, multi path fading should also be considered in addition to the basic two-path channel model.

The existence of fixed and mobile obstacles also results in fluctuation and random refraction of EM waves in air. Therefore, the amplitude and the phase of the received signal exhibit a random behavior with time. The multi path fading has been extensively investigated for the over-the-air wireless communication [29]. It was observed that, in general, this multi path channel obeys Rayleigh or log-normal probability distribution.

However, the underground environment for wireless communication does not present random refractions of the EM waves with time. As will be shown in Sect. 5.4, the wireless underground channel between two nodes is relatively stable. On the other hand, the roots of trees, rocks, clay particles and other objects in soil can still incur reflection and refraction for EM waves similar to the obstacles do in air. Consider the scenario where the transmitter node has the same inter-node distance related to multiple receivers that are placed at different locations, at the same burial depth. It is highly probable that different signal levels will be observed at the receivers. It occurs because the signal travels through different multi paths. Therefore, randomness in underground environment is due to the *locations* of the nodes rather than time, but it still obeys the Rayleigh probability distribution, where the variable is location instead of time. Accordingly, we consider that each path in the underground channel is Rayleigh distributed such that the envelope of the signal from each path is modeled as an independent Rayleigh distributed random variable, $\chi_i, i \in \{1, 2\}$. Consequently, for the one-path model, the received energy per bit per noise power spectral density is given by $r = \chi^2 E_b/No$, which has a distribution as $f(r) = 1/r_0 \exp(-r/r_0)$, where $r_0 = E[\chi^2]E_b/N_o$ and E_b/N_o can be directly found from the signal-to-noise ratio (SNR) of the channel.

For the two-path model case, in a similar way, it is assumed that the received signal is the sum of two independent Rayleigh fading signals. This model is denoted as *location dependent Rayleigh multi path channel*. Consequently, the composite attenuation constant, χ, in multi path Rayleigh channel is:

$$\chi^2 = \chi_1^2 + (\chi_2 \cdot \Gamma \cdot \exp(-\alpha \Delta(r)))^2 - 2\chi_1\chi_2\Gamma \exp(-\alpha\Delta(r)) \cos\left(\pi - \left(\phi - \frac{2\pi}{\lambda}\Delta(r)\right)\right), \quad (13)$$

where χ_1 and χ_2 are two independent Rayleigh distributed random variables of two paths, respectively. Γ and ϕ are the amplitude and phase angle of the reflection coefficient at the reflection point P, $\Delta(r) = r - d$, is the difference of the two paths and α is the attenuation constant.

Based on the above model, the goal of this section is to investigate the bit error rate (BER) characteristics of the underground channel. The results from these evaluations will supply guidelines to efficiently deal with the underlying challenges in the design of WUSNs.

It is well known that the BER of a communication system depends mainly on three factors: the channel model, the signal to noise ratio (SNR), and the modulation technique used by the system. Considering the channel model derived in Sect. 4.2.1, the signal to noise ratio (SNR) is given by $SNR = P_t - L_f - P_n$, where P_t is the transmit power, L_f is the total path loss given in (11), and P_n is the energy of noise. In order to determine the noise power P_n, field experiments were performed using the BVS YellowJacket wireless spectrum analyzer [41]. The average noise level is found to be -103 dBm at 30 cm depth, using the YellowJacket [19]. Although this noise P_n may change depending on the properties of the soil, this value is a representative value that can be used to represent the properties of underground BER.

For the evaluations, PSK modulation is considered. More specifically, adopting the 2PSK modulation scheme, the BER can be represented as a function of SNR as $BER = 0.5\text{erfc}(\sqrt{SNR})$, where erfc$(\cdot)$ is the error function and SNR is the signal to noise ratio.

In Fig. 8, the evaluation for single- and two-path models are shown as a function of operating frequency for different VWC levels. It can be observed that the VWC has an important impact on the BER compared to other parameters. As shown in Fig. 8, when the VWC increases from 5 to 15%, the BER also increases more than one order of magnitude. Moreover, the values of BER for the single path model are higher than the BER values for the two-path model. This shows the positive effects of the reflected path from the ground surface on BER. As already explained in Sect. 4.2.1, the two-path Rayleigh fading model is suitable for burial depths less than a threshold value. For the scenario considered in the evaluations in Sect. 4.2.1, this threshold value was found to be $H = 2$ m.

The effect of VWC in the two-path model is also shown in Fig. 8. One can verify that, even with a high VWC of 25%, it is possible to have a BER $<$20%, when a low depth deployment (dual-path model) is adopted with a low operating frequency.

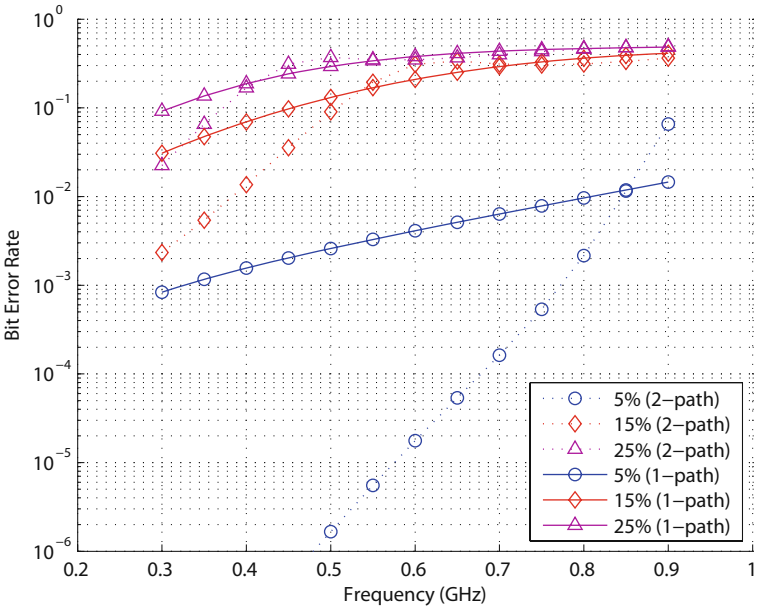


Fig. 8 BER vs. operating frequency and volumetric water content for one-path and two-path channel models

Finally, we are interested in knowing the maximum inter-node distance for the specific case of low depth deployment. We are assuming that the threshold burial depth is 2 m. In addition to BER, the inter-node distance is also important in the design of effective networking protocols for WUSNs. In Fig. 9, the maximum inter-node distance is shown for two-path model as a function of burial depth and the operating frequency. The BER target is assumed to be 10^{-3} and the maximum inter-node distance is found for this target. As shown in Fig. 9, when the frequency increases, the maximum distance decreases, as expected. Moreover, there are many sequences where we can observe the improvement of the inter-node distance, increasing the burial depth. This is true up to a certain burial depth; after this point, the inter-node distance decreases when the burial depth continues increasing. The behavior of this sequence is repeated many times. This result suggests that an optimal burial depth exists for a particular operation frequency, which is important for deployment of WUSNs.

4.3 Effects of Volumetric Water Content Variations in Soil

The underground channel model presented in Sect. 4.2.2 considers the main parameters that influence the communication performance. The volumetric water content is the most important soil parameter to be considered in WUSN design [15]. It is also possible to observe that each evaluation considers a *constant* value for VWC. For

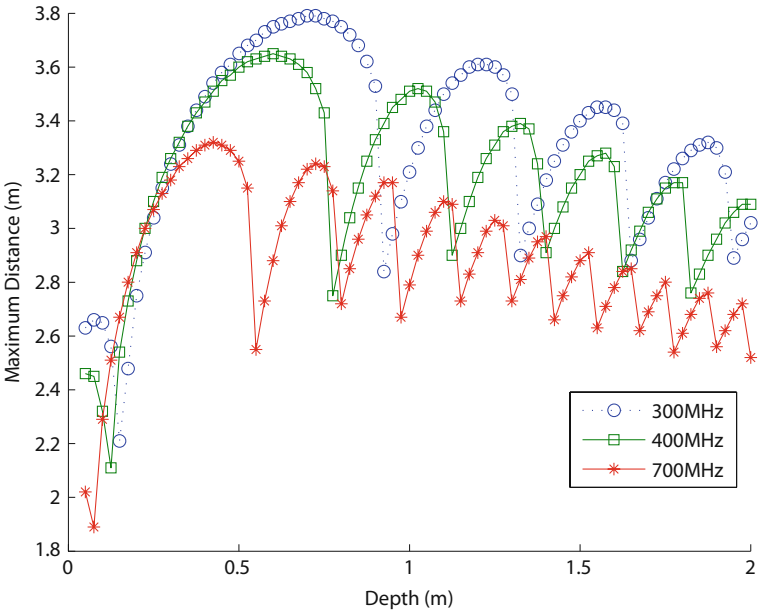


Fig. 9 Maximum inter-node distance vs. depth

instance, in Fig. 9, the simulations assumed that VWC is constant throughout the soil at different depths. However, experimental results reveal that VWC can also change with depth [14, 18, 42, 9]. Besides the variation of the VWC related to the soil depth, significant VWC variations are also observable as a function of weather, period of the year, and time of the day. Consequently, VWC changes for different depths and times and so the dielectric constant of soil, causing variations on path loss and BER. This section investigates the long-term and transient effects of VWC variations at different depths and times over the wireless underground communication.

4.3.1 Long-Term VWC Effects

Site measurements reveal that the volumetric water content does not linearly change with the depth [14, 18, 42, 9]. Generally, VWC increases with the depth first and then decreases with it. However, the variation characteristics as well as the specific depths where VWC starts to decrease also depend on, among other factors:

- the type of the soil;
- the type of the existent vegetation;
- the nature of the artificial watering process.

Therefore, it is important to know how these VWC variations change the original results from the model in Sect. 4.2.2.

To investigate the relation of wireless underground communication quality with the burial depth, experimental data in [42, 18] is used. Two different sites were

considered: *Set 1* and *Set 2*. The first site has soil with a quasi-homogenous distribution of sand, silt, and clay and the second site has a sandy soil. The VWC values for different times of the year as well as different depths are shown in Fig. 10. The properties of each experiment are described as follows:

- *Set 1:* Black soil with 22.75% sand and 28.1% clay [42].
- *Set 2:* Sandy soil with 50% sand and 15% clay [18].

The VWC values of these sets are shown in Fig. 10. Accordingly, the VWC values for Set 1 is significantly higher which implies worse communication through soil in Set 1 than in Set 2. Furthermore, VWC varies significantly with depth for Set 1 as well as season. These fluctuations for Set 1 would significantly affect the communication performance of WUSNs. However, the VWC measured in Set 2 is relatively constant with respect to both depth and season.

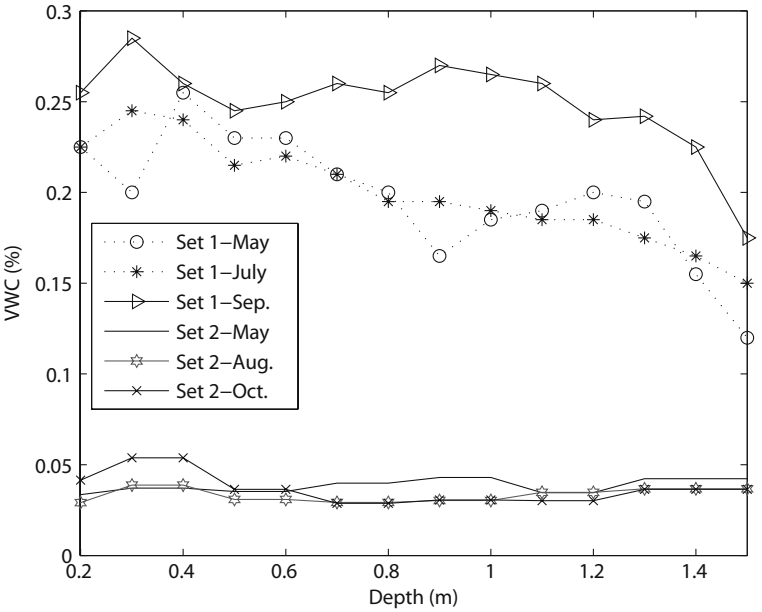


Fig. 10 Variation of VWC vs. burial depth. Two different types of soil were used. The values taken at different times of the year are also shown

The next step is to calculate the maximum inter-node distance, for $BER \leq 0.1\%$, plugging the values shown in Fig. 10 in the channel model presented in Sect. 4.2.2. We will compare these results with the case where the VWC is assumed to be uniform throughout the soil all time and at different depths. Consequently, the effects of variations of VWC on the communication performance are shown.

In Fig. 11, the maximum inter-node distance for BER target of 10^{-3} is shown as a function of depth for both data sets. The solid lines represent the cases where the VWC is considered constant throughout all depths for each data set. For Set 1,

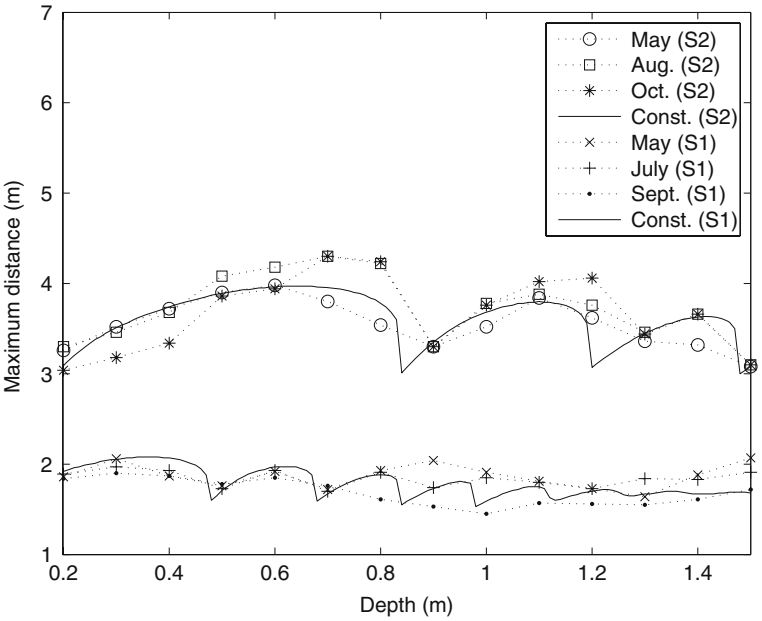


Fig. 11 The maximum inter-node distance vs. depth for the data Set 1 (S1) and Set 2 (S2) at different times of the year. Maximum distance calculated by considering uniform, that is, constant VWC at all depths is also shown

$VWC = 20\%$, which is the value measured at 0.3m depth in May according to [42] and for Set 2, $VWC = 3.7\%$.

First considering the results for the Set 1, it can be observed in Fig. 11 that the fluctuations estimated by the uniform VWC model are closely followed when depth is $d \leq 0.8$ m. However, at higher depths, longer hop distances are possible since the VWC decreases with increasing depth as shown in Fig. 10. The seasonal influence on the communication is also shown in Fig. 11. For burial depths higher than 0.8 m in Set 1, communication range is extended during May and decreased during September compared to the uniform VWC case. This can be explained by the fact that higher precipitation in the Set 1 region starts in July. Compared to the uniform VWC case, although there are differences in the maximum distance, using a single measurement of VWC it is possible to predict the communication characteristics accurately for burial depths less than 0.8 m. However, higher variations are observed when the depth is higher than 0.8 m. The maximum variation of VWC observed for Set 1 in Fig. 10 is larger than 60%, but verifying the Fig. 11, that variation actually causes a smaller one, almost 25%, on the maximum inter-node distance. Although, these results show the importance of the use of nodes equipped with soil moisture sensors and also environmental adaptive protocols, the differences between the uniform VWC and the real cases are not so critical as one is initially inclined to conclude when analyzing the Fig. 10.

The two sets present distinct types of soils. The black soil considered in Set 1 has more clay particles than the sandy soil considered in Set 2. As a result, the black soil holds more water compared to sandy soil, which negatively affects the communication. On the other hand, the sandy soil considered in Set 2, having lower volumetric water content, has the potential to improve the communication significantly. This fact is visible in Fig. 11. As shown in Fig. 10, the VWC for the data Set 2 does not vary significantly by depth or season. As a result, uniform VWC model closely matches the maximum distance achieved through actual VWC measurements, as shown in Fig. 11. Moreover, since the VWC of Set 2 is significantly lower than Set 1, the communication range is almost doubled. Intuitively, we can conclude that sandy soil is practically exempt from issues related to VWC variations. However, this assumption is not completely true as we will see later at the end of this section.

Considering the intense influence of VWC over the underground wireless communication, it is important to consider the relation between the variations in VWC and the corresponding changes in communication range in more details. This evaluation for the Set 1 is shown in Fig. 12, which relates a change in VWC with the maximum inter-node distance for BER target of 10^{-3}. The x-axis shows the change in percentage in VWC based on the measured data in Set 1 and the y-axis shows the corresponding change in maximum inter-node distance. The actual values as well as the fitted lines for three different months are given in Fig. 12. It is shown that an increase in VWC results in decrease in communication range. Although quadratic fitting is used, the relationship is linear. Moreover, the rate of change in distance with VWC (the line slope), depends on the season. For instance, a 20% decrease in VWC corresponds to a 10% increase in communication range in July, whereas, in September, this corresponds to a 4% increase in communication range.

4.3.2 Transient VWC Effects

The temporal analysis of the VWC presented so far considers seasonal changes and burial depths, i.e., long term changes in VWC. However, there are also *transient* scenarios that should be considered. For instance, the effects of a 5 min intensive rain on the communication are important for network operation. To understand how the dielectric constant of soil varies under these extreme scenarios, it is important to analyze the physical properties of soil in more detail.

Basically, if no watering process occurs frequently, the VWC of the soil is related to the capacity of the soil to *hold* the water, which depends on the surface area of its particles. Clay particles with a diameter smaller than 0.002 mm have a surface area of $8,000,000\,\text{cm}^2$ per gram. On the other hand, sand particles, with a diameter larger than 0.05 mm, have a surface area of only $227\,\text{cm}^2$ per gram [13]. Considering that water absorption is a function of the surface area, high values of VWC are expected for clay soils and low values for sandy soils.

The discussion in Sect. 4 shows that the water in the soil is usually classified into two based on its interaction with the soil particles. The *bound water* is tightly bounded to the surface of the soil particles. The *free water* is the remaining water in the soil, free of action of any particle [6]. Both kinds of water cause the elevation of

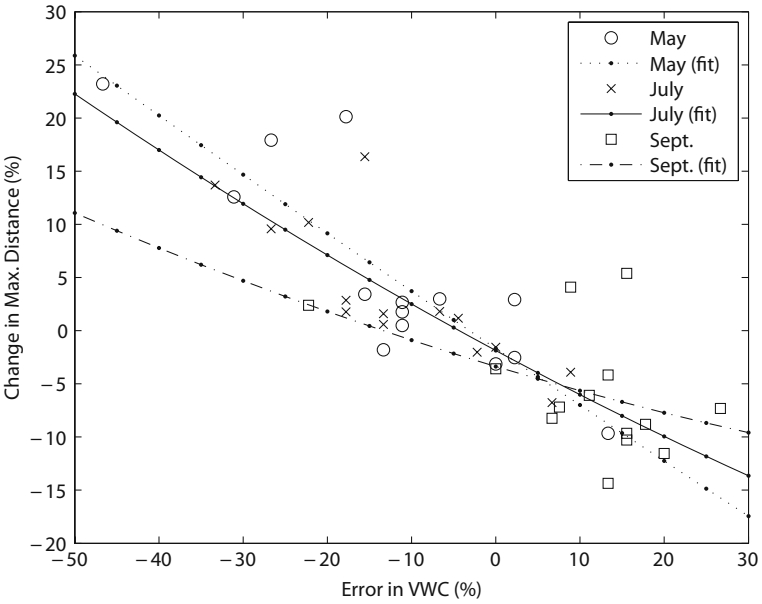


Fig. 12 Relationship between change in VWC and change in maximum inter-node distance

the soil dielectric constant, causing problems to the underground wireless communication. However, the dielectric constant of bounded water is between the dielectric constants of liquid water and ice. In other words, the effects of the bounded water are not positive for the EM wave propagation, but the effects of free water are even worse.

In general, for clay dominant soils, the majority of the VWC is related to the bound water. In a completely dry soil (VWC=0), the addition of water first causes the formation of bound water, followed by the formation of free water. From the point of view of the dielectric constant of the soil, as the VWC is increasing, one can see, first, a slow elevation of the dielectric constant due to the bounded water which is being formed. At a certain value of the VWC, called *transition level* [9], the slope of the dielectric constant drastically changes, and the dielectric constant assumes higher values more quickly. This second effect occurs because the bound water capacity of the soil is reached and the free water starts its formation, causing an abrupt increase in the dielectric constant. The transition point is smaller for sandy soils than for clay soils [6, 9]. Consequently, an abrupt elevation of the VWC can cause a higher variation of the dielectric constant of the sandy soil than the variation for a clay soil. Usually this scenario is temporary, for instance, for some minutes, due the fact that sandy soils will not absorb the same amount of water than clay soil and normally the majority of free water quickly infiltrates through the soil. This analysis also shows the importance for WUSN design to consider the use of some nodes equipped with soil moisture sensors.

As shown is this section, the model in Sect. 4.2.2 can be applied to real world even with the homogeneous VWC distribution assumption. Nonetheless, the spatio-temporal variability of the VWC must be controlled. For WUSN design, the adoption of some nodes equipped with soil moisture sensors is necessary. With these real-time measurements, cross-layer protocols could dynamically adjust the presented model considering both minor errors caused by the non-homogeneity of the VWC and also the temporal variation of VWC caused by environment factors.

5 Underground Experiments – The Practice

The characteristics of the underground channel in soil are modeled in Sect. 4 considering the effects of the reflection of surface and the multi path fading. Moreover, the theoretical analysis also considers different VWC levels and different frequencies at the range 300–900 MHz. As shown in Fig. 6, the model highlights the advantages of using the 300–400 MHz frequency range in terms of better values for path loss and smaller impact of the VWC over the communication. This section presents a set of field experiments that were realized to confirm such theoretical assumptions and also to provide a proof-of-concept of the feasibility of using commodity terrestrial WSN sensors in wireless underground applications. Finally, the experimental results also expose additional aspects to be carefully considered in a practical WUSN application.

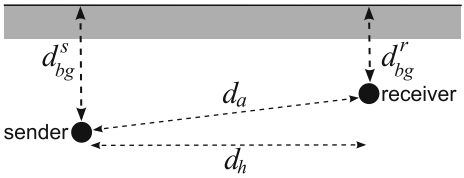
The underground experiments were carried out in Lincoln, NE, USA during August-November 2008 period [31]. The experiment site is shown in Fig. 13(a). The analysis of the soil texture is shown in Table 4 according to laboratory analysis [36]. MICA2 nodes operating at 433 MHz [10] were used in the experiments. As already discussed in Sect. 4, this operating frequency was chosen since it exhibits better propagation characteristics than higher frequencies typically adopted for terrestrial sensor nodes, e.g. 2.4 GHz. The experiment site was prepared as follows: 10 holes of 8 cm-diameter, with depths varying from 70 to 100 cm, were dug with an electric auger. A paper pipe with an attached MICA2 node is injected to each hole at different depths.

Table 4 Soil analysis report

Sample depth	Organic matter	Texture	%Sand	%Silt	%Clay
0–15 cm	6.4	Loam	27	45	28
15–30 cm	2.6	Clay Loam	31	40	29
30–45 cm	1.5	Clay Loam	35	35	30

For the experiments, a software suite called Small Grid Testbed for WSN Experiments (S-GriT) based on Java and TinyOS 1.1x [33] was developed. This application was developed to allow many number of the nodes acting as *receivers*. The S-GriT allows configuration of multiple experiments without physically accessing the buried nodes. More specifically, the following parameters are controlled: number of messages for each experiment, number of bytes per message, delay between the

(a) Out door environment of the experiments.



(b) Symbols used for distances in this Sect. 5.

Fig. 13 Symbols used for distances. Outdoor environment for the experiments [31]

transmission of each message, and the transmit power level. A MICA2 mote can assume one of the three roles in the S-GriT application:

- *Manager*: is located aboveground and connected to a laptop. It is used by the operator to configure and start the experiments and also to retrieve the results from the receivers.
- *Sender*: is buried underground and receives configuration information from the Manager, via wireless channel, and broadcasts test messages.
- *Receiver*: receives the test messages from the sender and prepares a summary containing the sequential number of each received message and the Received Signal Strength Indication (RSSI) level related to it.

The experiment setup and the terminology used in representing the results are illustrated in Fig. 13(b), where d_{bg} is the burial depth of the node, d_h is the horizontal inter-node distance, and d_a is the actual inter-node distance. The superscripts s and r are used to indicate the sender and the receiver. These values, as well as the transmit power, are varied to investigate the packet error rate (PER) and received signal strength (RSS) values of the underground communication.

The experiments are conducted for four values of transmit power, i.e., -3 dBm, 0 dBm, $+5$ dBm, and $+10$ dBm. The length of the packet is 30 bytes and the delay between the packets is 100 ms. Each experiment is consisted of 1000 packets. The number of packets correctly received by the receiver node is recorded along with the signal strength for each packet. Accordingly, the PER and the RSS level from each receiver are collected. To prevent the effects of hardware failures of each individual MICA2 node, qualification tests were performed before each experiment.

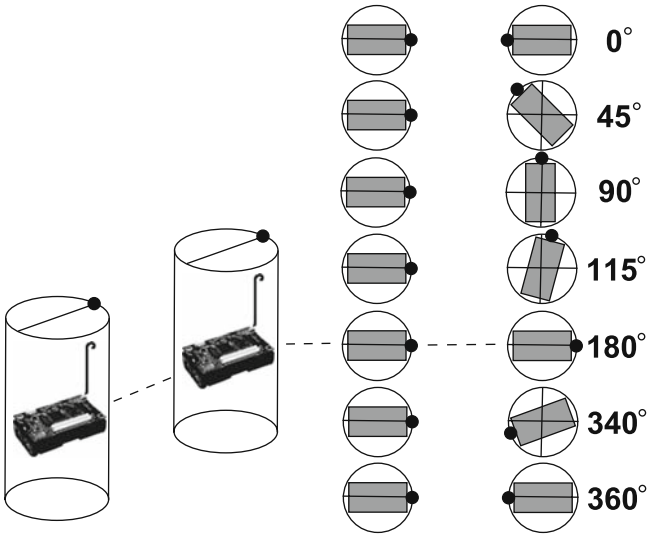
Accordingly, through-the-air tests, which consists of 200 packets of 30 bytes, were performed to (1) determine compliant nodes and (2) confirm that the battery level of a node is above a safe limit. A node is labeled compliant with a given set of nodes if (1) its PER varies within 10% of the average PER calculated for the set of nodes and (2) its RSS average varies, at maximum, ± 1 dBm from the average RSS for the set of nodes. The safe limit for the battery level has been determined as 2.5V for the MICA2 nodes. We observed that, in general, only 50% of the 11 nodes used were qualified for each experiment.

We present the results of these experiments to highlight important parameters that affect the wireless underground communication. In Sect. 5.1, the effects of antenna orientation, not previously considered in Sect. 4, are presented. The influence of the burial depth is shown in Sect. 5.2. In Sect. 5.3, the results related to the inter-node distance parameter are investigated, showing the actual limitations of using commodities WSN sensors in WUSN applications. The temporal characteristics of the wireless underground communication channel are analyzed in Sect. 5.4. Finally, the effects of VWC over the communication are presented in Sect. 5.5. The main goal of this section is to verify if the experimental results agree with the theoretical wireless underground communication models presented in the Sect. 4. Additionally, we will see how field experiments can reveal *hidden* aspects not considered in the theoretical study.

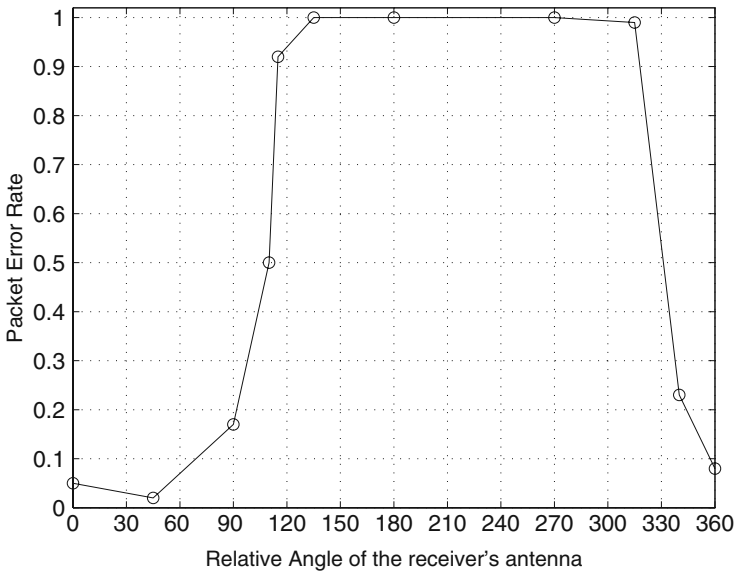
5.1 Antenna Orientation

The experiments with the antenna orientation were performed by placing a sender and a receiver at different angles as shown in Fig. 14(a). The 17 cm-length antenna of MICA2 is a standard one-quarter wavelength monopole antenna and this length reflect the wavelength of a 433 MHz wave. The MICA2 antenna has a radiation pattern that does not exhibit a perfect sphere and it matches the dipole antenna model presented in [27]. This type of antenna, in conjunction with the MMCX connector, allows a great flexibility in terms of antenna orientation. However, as shown in Fig. 14(a), the motes always have their antennas placed in the vertical polarization. The horizontal polarization was not considered because this implies larger holes for the MICA2 deployment (>19 cm). Also, the hybrid arrangement of antennae, mixing horizontal and vertical polarizations between the motes, is not even considered because it is well known that such arrangements are the worst cases in terms of signal loss. However, considering that all the motes have vertical polarization, this experiment aimed to answer the question if the way the motes are buried is critical for WUSNs. As will be shown next, while the antenna orientation is usually ignored for most WSN applications, it is a critical factor for wireless underground communication.

The experiments were performed at a depth of $d_{bg} = 40$ cm and at a distance of $d_a = d_h = 100$ cm between the sender and the receiver. In Fig. 14(b), the packet error rate (PER) is shown as a function of the node orientation. It can be observed



(a) Relative angles for the antenna.



(b) PER vs. relative angle for the antenna.

Fig. 14 **(a)** The schema used to test the effects of the antenna orientation for the underground-to-underground communication. **(b)** PER at the receiver mote [31]

that the best results (PER$<$0.1) were obtained in the configuration usually called *back-to-back* [27], which is where the angle equals to 0°. It can also be observed that when the relative angle varies from 90° to 340°, the PER drastically increases. If the antenna orientation is between 120° and 300°, virtually no communication is possible. This issue occurs because of two factors:

- The radiation pattern of the MICA2 antenna is not perfectly symmetric in all directions.
- When the receiver mote is rotated, besides the antenna orientation, the inter-node distance also varies. This variation is minimum ($<$8 cm), but for wireless underground communication, this small change in the inter-node distance may be critical.

The implications of this result are not trivial for the WUSN design and implementation. For instance, consider the case of three MICA2 nodes A, B, and C, linearly deployed at the same burial depth. Also, the inter-node distance between A and B is the same as the distance between B and C and the node B is broadcasting. Depending on the chosen inter-node distance, the quality of the signal from the sender B will not be the same from the viewpoints of the nodes A and C. Moreover, it is also possible to have the case where A can communicate with B, but C cannot. The antenna orientation is also an issue for the WUSN implementation because it implies that the deployment of the motes is not a trivial burial process. The mote must be buried with its antenna aligned to a specific direction with a minimum deviation. In addition to the challenges presented in chapter "Deployment Techniques for Sensor Networks", soil medium introduces additional challenges for WUSN deployment.

Although this antenna orientation problem is serious, it will only occur for some critical inter-node distances. Moreover, the kind of transceiver and antenna used by the nodes may also minimize the problem. However, the conclusion from this experiment is that the antenna orientation is an additional constraint that must be considered for the deployment of WUSNs, usually an aspect not very critical in traditional WSNs. This is especially true for multi-hop underground networks in conjunction with critical inter-node distances.

5.2 Effects of Burial Depth

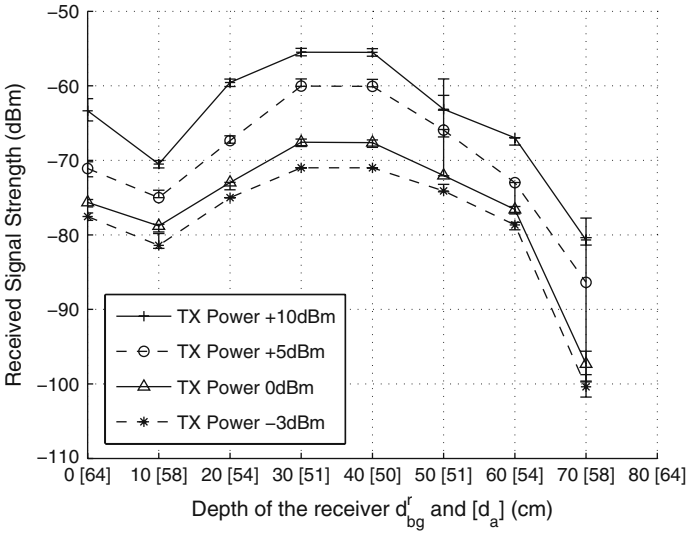
In this section, the effects of the burial depth on the signal strength and PER are shown. As shown in Table 4, the composition of the soil does not change significantly between the topsoil and subsoil regions. Therefore, the results of these specific experiments, where the burial depth of the receiver is varied, are directly related to the reflection effects of the ground surface. When the nodes are near the surface (smaller burial depth), a better quality for the communication is expected due the positive effects of the reflected waves as explained in Sect. 4.2.1. When the nodes have higher burial depth, the high attenuation of the soil will prevent that

some waves arrive at the surface and still have enough strength to be reflected and to arrive at the receiver.

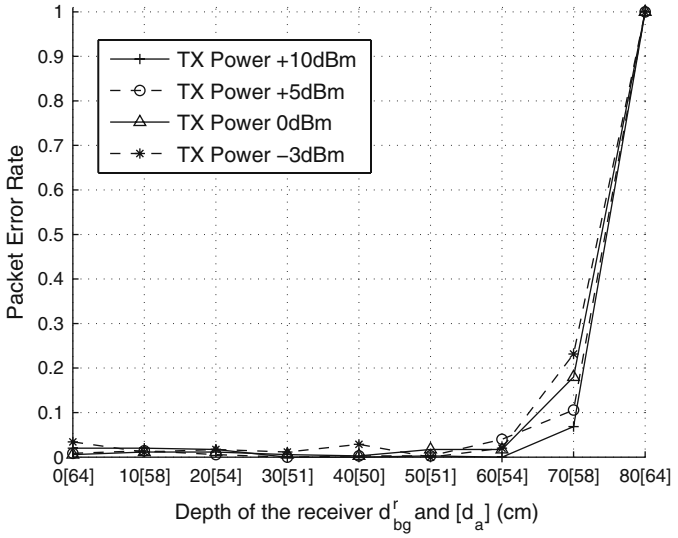
For these experiments, the horizontal inter-node distance between the sender and the receiver is fixed ($d_h = 50\,\text{cm}$), the burial depth of the sender is also fixed ($d^s_{bg} = 40\,\text{cm}$) and the depth of the receiver is varied from 10 to 80 cm using different transmit power levels. In Fig. 15(a) and (b), the RSS and PER values are shown, respectively, as a function of the receiver depth. The actual distance, d_a, between the sender and the receiver is also indicated in parenthesis on the x-axis. Each line in the figures shows the results for different transmit power levels. In Fig. 15(a), the variance of the RSS is also shown along with the average values for each point.

As shown in Fig.15(a), when we increase the actual inter-node distance, d_a, the signal strength decreases as expected. The highest signal strength corresponds to the receiver depth of 30–40 cm and the signal strength gradually decreases if the receiver burial depth is smaller than 30 cm or higher than 40 cm. One exception to this case is $d^r_{bg} = 0\,\text{cm}$, where the signal rays from above the ground impact the received signal strength positively and increase the RSS for each transmit power level. An important observation is the significant difference of RSS values at the same inter-node distance but at different burial depths. As commented before, it is not expected that this difference is being caused by the soil composition, which has only small variations for this specific experiment site. To see how the burial depth is affecting the RSS, consider the cases where d_a=58 cm: an additional attenuation of 20 dB is observed for the same inter-node distance of d_a=58 cm, when the receiver is buried at 70 cm compared to the burial depth of 10 cm. This behavior occurs mainly due to the reflection of the EM signals from the soil surface, which positively affects the RSS when the nodes are buried closer to the surface. This result validates the two-path channel model for the wireless underground channel studied in the Sect. 4.2.2.

It can be observed in Fig. 15(b), that for the receiver burial depth of 70 cm, the PER increases ($0.1 < \text{PER} < 0.2$) and an increase in burial depth from 70 cm to 80 cm immediately causes a communication loss. It is important to observe that this behavior occurs for all tested transmit power levels, highlighting the fact that the burial depth is an important parameter in a WUSN design. This conclusion is also in accordance with the classification proposed in the Fig. 1, where a distinction between Topsoil and Subsoil WUSNs is necessary specifically considering the burial depth of the nodes. It can also be observed in Fig. 15(a) that the RSS values have a very small variance for all depths and transmit power levels. Hence, for a given node deployment, the underground communication channel is very stable as long as the composition of the soil does not change, as will be analyzed in Sect. 5.4. However, the exception for the stability of underground communication is the effect of Volumetric Water Content (VWC) which will be explained in Sect. 5.5.



(a) Received Signal Strength vs. depth of the receiver (d^r_{bg}) and actual inter-node distance d_a.



(b) Packet Error Rate vs. depth of the receiver (d^r_{bg}) and actual inter-node distance (d_a).

Fig. 15 Effect of the reflected path from the ground surface. Sender buried with $d^s_{bg} = 40$ cm. Horizontal inter-node distance $d_h = 50$ cm. The depth of the receiver is varying from 0 to 80 cm [31]

5.3 Effects of Inter-Node Distance

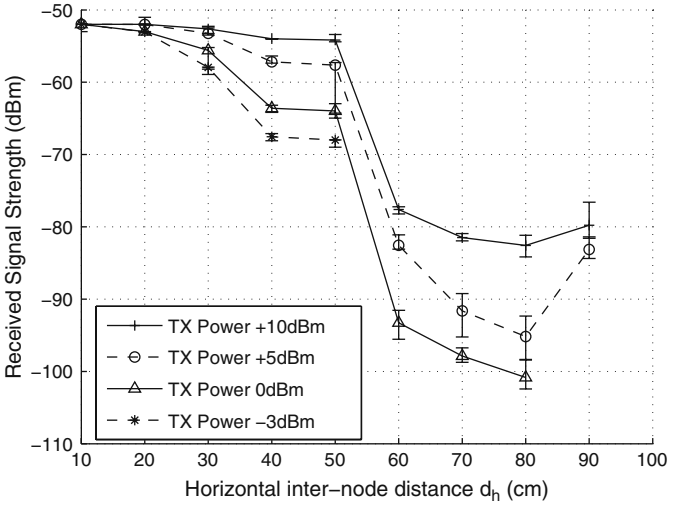
In this section, the effects of inter-node distance on WUSN design are analyzed. The results of the presented field experiments show the effects of the inter-node distance on the signal strength and PER. For our tests, the burial depth of the sender and the receiver is fixed ($d^s_{bg} = d^r_{bg} = 40\,\text{cm}$), and the inter-node distance is varied from 10 to 100 cm, again using different transmit power levels. In Fig. 16, the RSS and PER values are shown, respectively, as a function of the depth of the receiver for different transmit power levels and the variance of the RSS values are also shown.

As shown in Fig. 16(b), the maximum inter-node distance is found to be between 80 and 90 cm for transmit powers of +5 and +10 dBm, and 50 cm for −3 and 0 dBm. For transmit powers of −3 and 0 dBm, when the inter-node distance varies from 60 to 70 cm, the significant decrease of the signal strength can be observed in Fig. 16(a), which results in an abrupt PER increase as shown in Fig. 16(b). The first main conclusion of these results is that typical WSN nodes, such as the MICA2, are not adequate for Subsoil WUSNs, at least considering the use of a low power transceiver ($< +10\,\text{dBm}$), the typical antenna of the MICA2, and similar soil environment. In Sect. 4, it has been found that a path loss of about 30 dB corresponds to an inter-node distance of 100 cm, which is also observed in Fig. 16(a), where an attenuation of almost 30 dB for an inter-node distance of 90 cm is observed with transmit power of +10 dBm. However, more powerful transceivers can make possible the implementation of Subsoil WUSNs.

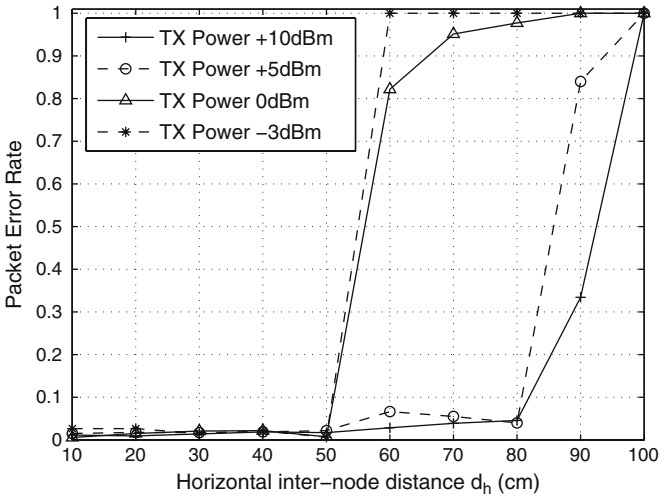
The second main conclusion from the burial depth experiments is that there are very good power management opportunities to be explored. Although more powerful transceivers may be necessary for some WUSN solutions, specially for Subsoil WUSN applications, it does not mean that a higher transmit power level always results in better communication quality. For instance, in Fig. 16(b), in the region where the signal is received with good quality (PER $< 10\%$), the communication behavior is very stable for all transmit power levels, including −3 dBm. Based on these results, for a given inter-node distance, there is a minimum transmit power for which the underground-to-underground communication has practically the same reliability when compared to a scenario which uses a higher transmit power level. This analysis highlights the possibility of efficient power consumption schemes for WUSNs, even expecting a more demanding energy solution caused by the use of more powerful transceivers.

5.4 Temporal Characteristics

In this section, the temporal characteristics of the wireless underground channel are investigated. Accordingly, a 24-hour experiment is performed by fixing the horizontal inter-node distance between the sender and the receiver ($d_h = 50\,\text{cm}$), the burial depth of the sender and the receiver ($d^s_{bg} = d^s_{bg} = 40\,\text{cm}$), and the transmit



(a) Received Signal Strength vs. horizontal internode distance (d_h).



(b) Packet Error Rate vs. horizontal inter-node distance (d_h).

Fig. 16 Maximum inter-node distance for underground-to-underground communication. Sender and receiver buried at depth = 40 cm ($d^s_{bg} = d^r_{bg} = 40$ cm). The inter-node distance d_h is varying from 10 to 100 cm [31]

power at +10 dBm. For comparison, the same experiment is repeated *over-the-air*, in an indoor environment with an inter-node distance of 5 m and a transmit power of +10 dBm. In Fig. 17, the RSS and PER values are shown, respectively, as a function of time. Each data point shows the average of 30 min of RSS or PER information, which corresponds to 150 packets. In Fig. 17(a), the confidence intervals of the RSS are also shown along with the average values for each point as well as the

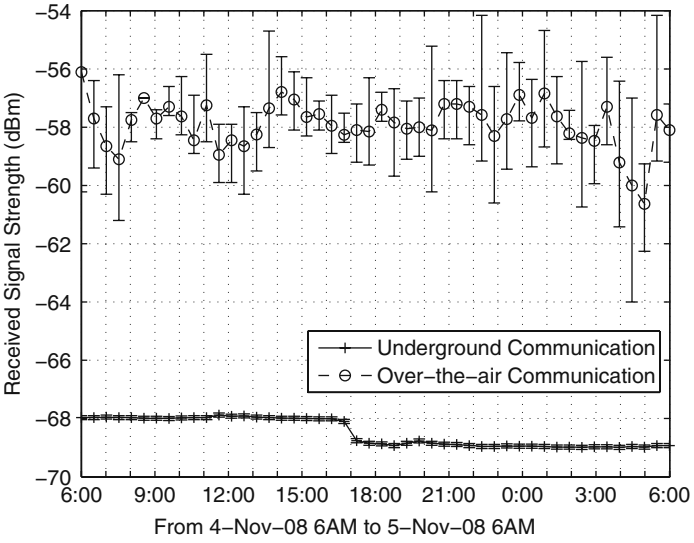
results of the over-the-air experiments. In Fig. 17(b), the temporal evolution of the cumulative PER is shown for underground communication.

As shown in Fig. 17(a), the maximum variation of the signal strength is only 1 dB. No precipitation event was registered during the period and only a 8°C variation is observed on the temperature during the experiment [23]. Compared to the over-the-air communication, where both the average and the variance of the RSS vary significantly with time, underground wireless channel exhibits a stable characteristic with time. As shown in Fig.17(b), during the same period of time, PER is always smaller than 0.5% with a small variance. This result agrees with the models for wireless underground channel proposed in Sect. 4, which also point out the high stability of the wireless underground channel. The temporal stability has important impacts in the design of routing and topology control protocols for WUSNs.

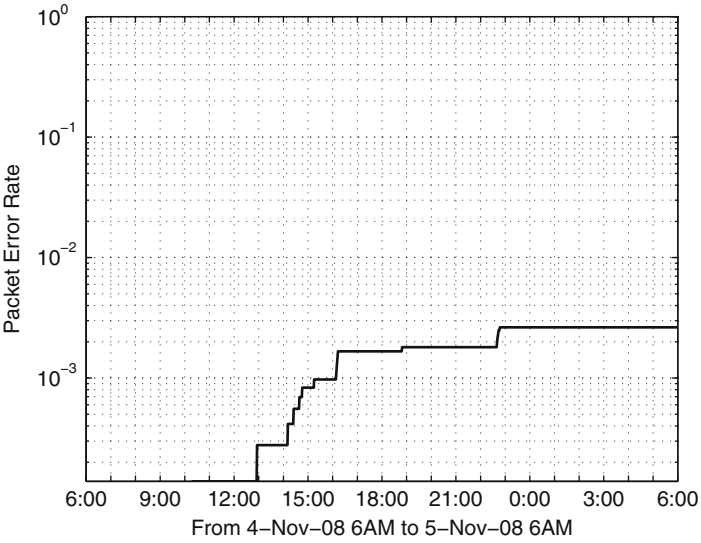
5.5 Effects of Soil Moisture

In this section, the effects of the volumetric water content (VWC) on the signal strength and PER are discussed. Accordingly, the burial depth of the sender and the receiver is fixed ($d_{bg}^{s} = d_{bg}^{r} = 40$ cm), two different inter-node distances ($d_h = 30$ and 40 cm) are used in conjunction with two different VWC levels (dry and wet soil), and the transmit power is varied. The *dry soil* experiments refer to tests realized on Oct 20, 2008, a sunny day, and the *wet soil* experiments were performed on Oct 22, 2008, a rainy day, when 2.5 ins. of precipitation was recorded [23]. Based on the *oven drying method* [39], the different VWCs are measured to be 11% for dry soil and 18% for wet soil, which corresponds to an increase of almost 60% in VWC. In Fig. 18(a) and (b), the RSS and PER values are shown, respectively, as a function of the transmit power level of the sender. Each line in the figures shows the results for different VWC and inter-node distances.

As shown in Fig. 18(a), for high VWC, i.e., wet soil, the attenuation increases by 12–20 dB compared to dry soil. The Fig. 18(b) also reveals that the increase of VWC implies higher PER. As shown in Fig. 18, the negative effect of the VWC over the quality of the communication is reduced when the transmit power is increased. This result agrees with the model for wireless underground channel in the Sect. 5.5, which showed that high levels of VWC have significant negative effects over the underground communication. Therefore, for a scenario where the natural or artificial irrigation is expected to occur, the design of WUSN protocols should carefully consider the variation of the VWC of the soil. For instance, the communication protocol may consider the volumetric water content measurements of a physical region to make informed routing decision or even consider to temporarily increase the transmit power of some of the nodes in order to decrease the adverse effects of VWC.

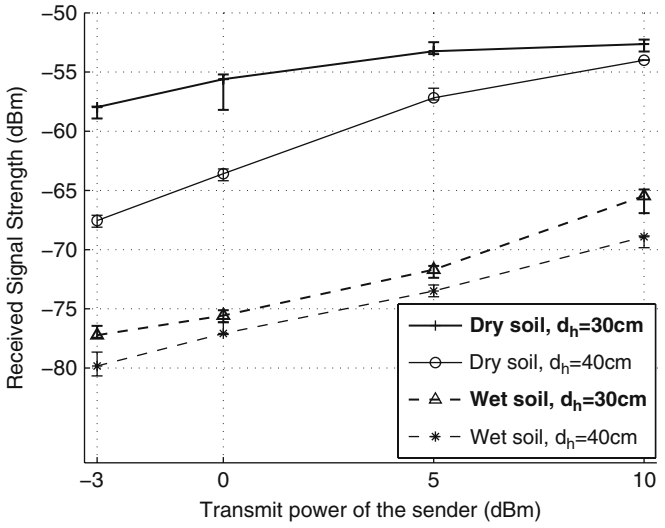


(a) Received Signal Strength vs. Time. Comparison between underground and over-the-airwire less communication.

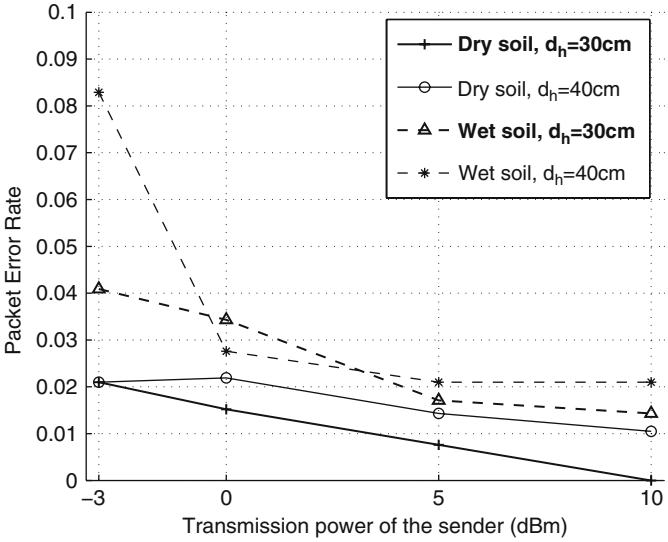


(b) Historical evolution of PER over the Time (PER $<1\%$ for all 24h period).

Fig. 17 Period of 24 h: RSS and historical evolution of PER. Sender and receiver buried with $d^s_{bg} = d^r_{bg} = 40$ cm. Horizontal inter-node distance $d_h = 50$ cm. Transmit power$= +10$ dBm [31]



(a) Comparison of the Received Signal Strength with dry and wet soil scenarios.



(b) Comparison of the PER with dry and wet soil scenarios.

Fig. 18 Effect of the VWC over the wireless underground-to-underground communication. Sender and receiver buried at $d^s_{bg} = d^r_{bg} = 40\,\text{cm}$, 2 inter-node distances: $d_h = 30\,\text{cm}$ and $d_h = 40\,\text{cm}$. Varying transmit power of the sender [31]

6 Open Research Issues

Considering the main aspects revealed by the theoretical and experimental results, the following challenges emerge for the design of WUSNs:

6.1 Energy Efficiency

The power consumption is one of the most important factors in the design of wireless sensor networks [3]. This is even more pronounced in WUSN because of the environmental effects. First, the higher attenuation in the underground communication requires higher transmit power levels to establish reasonable communication ranges. Second, changing batteries is even harder, if not impossible, in WUSN compared to its terrestrial counterpart. Consequently, energy efficient solutions are crucial for the design of WUSNs.

There are many possible methods to save energy in WUSN. Although the underground channel exhibits multi path effects with higher attenuation, it is observed that the wireless underground channel has been found to exhibit extreme temporal stability, which is important in the design of routing and topology control protocols. However, the temporal stability of the wireless underground channel can also be explored in terms of power saving schemes. As a result, simple modulation schemes and error control techniques may be sufficient for efficient underground communication. This in turn decreases the overhead incurred by these techniques and provides a potential way for saving energy.

It is also important to consider that most underground applications require less frequent traffic. Consequently, sleep schedules can be incorporated into the design of MAC schemes. Furthermore, it was shown that, for a given deployment and soil composition, there is a minimum transmit power for which the underground-to-underground communication has the same reliability compared to cases where higher transmit power levels are used. This feature is another potential way for decreasing the power consumption of the overall WUSN solution.

Compared with typical WSNs, the composition of the specific mentioned features of WUSNs reveals the need for the development of WUSNs protocols that exploit these features aiming reliable communication and energy efficiency.

Although the focus of this chapter is the communication of sensors buried underground, the use of hybrid architectures with aboveground nodes can also dramatically decrease the energy consumption of the overall WUSN solution. In fact, in a generic WUSN architecture, there may still be some devices, such as sink nodes, deployed above the ground. Hence, the communication between the underground sensors and the aboveground sinks should also be considered in the WUSN design.

6.2 Topology Design

The results for maximum attainable communication range illustrates that underground environment is much more limited compared to terrestrial WSNs. In particular, at the operating frequency of 300–400 MHz, the theoretical communication range can be extended up to 5 m. As described in Sect. 4.3, underground communication is also affected by the changes according to depth. The experiment results validate the theoretical model, also showing that the burial depth is very important for the WUSN design due to the effects of reflected rays from the underground-air interface at the surface. As a result, different ranges of communication distance can be attained at different depths. This requires a topology structure that is adaptive to the 3D effects of the channel. Optimum strategies to provide connectivity and coverage should be developed considering these peculiarities. This provides challenges in addition to the conventional deployment challenges discussed in chapter "Deployment Techniques for Sensor Networks".

Both mentioned theoretical and experimental results related to the inter-node distance suggest that multi-hop communication is essential in WUSNs. Consequently, in the design of a WUSN topology, this aspect should be emphasized. However, even with the use of multi-hop communication, the limitations of the commodity sensor nodes for WUSNs are also observed. For the specific Subsoil WSUN scenario used in the testbed (40 cm burial depth), the inter-node distance was smaller than 1 m. In terms of signal attenuation, this corresponds to roughly a 1:20 attenuation rate compared to *through-the-air* communication in an outdoor environment [5]. Consequently, a new generation of nodes with more powerful transceivers and/or more efficient antennas maybe required for the deployment of WUSN applications. We expect that the use of higher transmit power levels will provide higher communication ranges with still acceptable energy consumption, especially when the energy-efficient features of WUSNs mentioned before are effectively exploited. The experimental results also highlighted that the antenna orientation of the underground nodes plays an important role in the connectivity of WUSNs and must be considered in the WUSN topology design.

6.3 Operating Frequency

The channel model presented in this chapter clearly illustrates the fact that the attenuation increases with operating frequency, which motivates smaller frequency values considering the high attenuation. However, this results in a tradeoff between the frequency and the antenna size for the WUSN device. Our results show that 300–400 MHz band is suitable for WUSNs. The simulation results show that acceptable communication ranges are possible. Moreover, there are already sensor motes working in this band in the market; in fact, our 433 MHz MICA2 choice for the experiments was motivated by these observations.

As explained in Sect. 4.2, the communication performance at low depth reveals that using a fixed operating frequency may not be the best option for WUSNs.

Cognitive radio techniques [1] can be exploited to adapt to the changing environmental conditions. Furthermore, our analysis reveals that the optimal frequency to reach the maximum communication range varies by depth. Consequently, cognitive radio techniques can provide an adaptive operation for the WUSNs in this dynamic environment. Further experimental investigation in this area must be conducted to investigate the feasibility of this approach.

6.4 Cross-Layer and Environment-Aware Protocol Design

As shown in Sect. 4.3, external factors that directly influence the soil properties also have a great impact over the communication performance. The network topology should be designed to be robust to support drastic changes in the channel conditions. The most important soil property to be considered in a WUSN design is the VWC. Hence, the analysis of the spatio-temporal variation of the VWC in the region where a WUSN application will be deployed is very important. Furthermore, soil composition at a particular location should be carefully investigated to tailor the topology design according to specific characteristics of the underground channel at that location. For instance, different node densities and inter-node distances should be investigated for the WUSN deployment in a region that presents significant spatial soil composition heterogeneity.

Besides the effect of soil type, the seasonal changes result in variation of volumetric water content, which significantly affects the communication performance. This specific environment parameter was also investigated through experiments. The experimental results validated the theoretical results and clearly showed that the adverse effects of the VWC can be decreased using higher transmit power levels. Therefore, in the protocol design for WUSNs, the environment dynamics need to be considered. This implies an *environment-aware protocol* that can adjust the operation parameters according to its surrounding.

Finally, the feasibility of WUSNs depends on the investigation of multiple novel factors not considered for traditional terrestrial WSNs. For instance, WUSNs must be robust enough to environment parameters, such as VWC and soil composition. To this end, a new generation of sensors with more powerful transceivers may be necessary. However, there are many specific positive features of the underground environment, such as the temporal stability, that can be exploited to achieve reliable and energy-efficient communication. The use of hybrid architectures, cognitive radios, and cross-layer protocols that dynamically change the transmit power, packet size, and encode schema in function of the VWC are some of the options available for WUSN design.

Acknowledgements This work is supported by the UNL Research Council Maude Hammond Fling Faculty Research Fellowship. The authors would like to thank Dr. Suat Irmak for his valuable comments throughout the development of the WUSN testbed and Emily Casper and the UNL Landscaping Services staff for their valuable helps during the experiments.

References

1. I. F. Akyildiz, W. Y. Lee, M. C. Vuran, and S. Mohanty, "NeXt generation/dynamic spectrum access/cognitive radio wireless networks: a survey," *Computer Networks Journal (Elsevier)*, Vol. 50, pp. 2127–2159, September 2006.
2. I. F. Akyildiz and E. P. Stuntebeck, "Wireless underground sensor networks: research challenges," *Ad Hoc Networks Journal (Elsevier)*, Vol. 4, pp. 669–686, July 2006.
3. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks Journal (Elsevier)*, Vol. 38, No. 4, pp. 393–422, March 2002.
4. I. F. Akyildiz, Z. Sun, and M. C. Vuran, "Signal propagation techniques for wireless underground communication networks," *Physical Communication Journal (Elsevier)*, Vol. 2, No. 3, pp. 167–183, September 2009.
5. C. Alippi and G. Vanini, "Wireless sensor networks and radio localization: a metrological analysis of the MICA2 received signal strength indicator," in *Proceedings of IEEE Workshop on Embedded Networked Sensors*, Florida, USA, November 2004.
6. J. Behari, "Microwave Dielectric Behavior of Wet Soils", Springer, New York, 2005.
7. A. Chehri, P. Fortier, and P. M. Tardif, "Application of ad-hoc sensor networks for localization in underground mines," in *Proceedings of Wireless and Microwave Technology Conference 2006 (WAMICON '06)*, Florida, USA, December 2006.
8. A. Chehri, P. Fortier, and P. M. Tardif, "Security monitoring using wireless sensor networks," *Communication Networks and Services Research, 2007 – CNSR '07*, pp. 13–17, May 2007.
9. A. Chukhlantsev, "Microwave Radiometry of Vegetation Canopies", Springer, New York, 2006.
10. Crossbow Mica2 node, http://www.xbow.com.
11. D. J. Daniels, "Surface-penetrating radar," *Electronics & Communication Engineering Journal*, Vol. 8, No. 4, pp. 165–182, August 1996.
12. N. Elkmann, et al., "Development of fully automatic inspection systems for large underground concrete pipes partially filled with wastewater," *Robotics and Automation 2007*, pp. 130–135, April 2007.
13. H. D. Foth, "Fundamentals of Soil Science," Sixth Edition, John Wiley & Sons, New York, 1978.
14. J. R. Holdem, et al., "Estimation of the number of frequencies and bandwidth for the surface measurement of soil moisture as a function of depth," *IEEE Transactions on Instrumentation and Measurement*, Vol. 49, No. 5, pp. 964–970, October 2000.
15. T. R. H. Holmes, "Measuring surface soil parameters using passive microwave remote sensing. The ELBARA field campaign 2003," MSc Thesis, Vrije Universiteit Amsterdam, 71 pp., 2003.
16. G. A. Kennedy and P. J. Foster, "High resilience networks and microwave propagation in underground mines," *The 9th European Conference on Wireless Technology 2006*, pp. 193–196, September 2006.
17. Land and Water Development Division, FAO Organization of the United Nations, "Topsoil Characterization for Sustainable Land Management," in *Proceedings of Rome: Food and Agricultural Organization*, Rome, Italy, October 1998.
18. H. Li, Z. Dong, L. Wang, "Research on temporal and spatial variety of soil moistures of shifting sand dune and four main plant communities on Otindag sandy land," *Journal of Arid Land Resources and Environment*, Vol. 20, No. 3, May 2006.
19. L. Li, M. C. Vuran, and I. F. Akyildiz, "Characteristics of underground channel for wireless underground sensor networks," in *Proceedings Med-Hoc-Net '07*, Corfu, Greece, June 2007.
20. K. Martinez, R. Ong, and J. Hart, "Glacsweb: A sensor network for hostile environments," in IEEE SECON '04, pp. 81–87, 2004.
21. J. F. Mastarone and W. J. Chappell, "Urban sensor networking using thick slots in manhole covers," in *Proceedings of Antennas and Propagation Society International Symposium 2006*, New Mexico, USA, July 2006.

22. T. W. Miller, et al., "Effects of soil physical properties on GPR for landmine detection," in *Fifth International Symposium on Technology and the Mine Problem*, April 2002.
23. NOAA's National Weather Service Weather Forecast Office, http://www.crh.noaa.gov/gid/?n=neb rainfall observed#archive.
24. J. Paek, K. Chintalapudi, R. Govindan, and S. Masri, "A wireless sensor network for structural health monitoring: performance and experience," in *Proceedings of IEEE Workshop on Embedded Networked Sensors (EmNetS-II)*, May 2005.
25. C. Park, Q. Xie, P. Chou, and M. Shinozuka, "Duranode: wireless networked sensor for structural health monitoring," in *Proceedings of IEEE Sensors 2005*, pp. 277–280, November 2005.
26. N. Peplinski, F. Ulaby, and M. Dobson, "Dielectric properties of soils in the 0.3–1.3-GHz range," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 33, No. 3, pp. 803–807, May 1995.
27. T. Scott, "Mica mote antenna radiation pattern analysis", *Master's thesis – University of Victoria*, May 2004.
28. A. Sheth, et al., "Senslide: a sensor network based landslide prediction system, " in *Proceedings of SenSys'05*, pp. 280–281, 2005.
29. G. Stuber, "Principles of Mobile Communication," Kluwer Academic Publishers, Dordrecht, 1996, 2/e 2001.
30. E. Stuntebeck, D. Pompili, and T. Melodia, "Underground wireless sensor networks using commodity terrestrial motes," poster presentation at *IEEE SECON 2006*, September 2006.
31. A. R. Silva and M. C. Vuran, "Empirical evaluation of wireless underground-to-underground communication in wireless underground sensor networks," in *Proceedings IEEE Int. Conference on Distributed Computing in Sensor Systems (DCOSS '09)*, Marina Del Rey, CA, June 2009.
32. Z. Sun and I. F. Akyildiz, "Channel modeling of wireless networks in tunnels," in *Proceedings of IEEE Globecom '08*, New Orleans, USA, November 2008.
33. TinyOS web site, http://www.tinyos.net/.
34. M. C. Vuran and I. F. Akyildiz, "Packet size optimization for wireless terrestrial, underwater, and underground sensor networks," in *Proceedings of IEEE INFOCOM '08*, Phoeniz, AZ, April 2008.
35. J. Wait and J. Fuller, "On radio propagation through earth: antennas and propagation," *IEEE Transactions on Antennas and Propagation*, Vol. 19, No. 6, pp. 796–798, November 1971.
36. Ward Laboratories, http://wardlab.com.
37. T. P. Weldon and A. Y. Rathore, "Wave propagation model and simulations for landmine detection," *Tech. Rep.*, University of North Carolina at Charlotte, 1999.
38. G. Werner-Allen, et al., "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing* Vol. 10, No. 2, pp. 18–25, 2006.
39. C. O. Willits, "Methods for determination of moisture – oven drying," *Analytical Chemistry*, Vol. 23, No. 8, pp. 1058–1062, 1951.
40. N. Xu, et al., "A wireless sensor network for structural monitoring," in *Proceedings of ACM Conference on Embedded Networked Sensor Systems*, Baltimore, MD, November 2004.
41. YellowJacket wireless spectrum analyzer, Berkeley Varionics Systems, Inc., http://www.bvsystems.com.
42. K. Zhang, W. Peng, L. Wang, A. Fu, and X. Xu, "Variation of soil temperature and soil moisture on black soil profile in seasonal frozen area of northeast China," *Georgraphical Research*, Vol. 26, No. 2, March 2007.

Body Sensor Networks for Sport, Wellbeing and Health

Douglas McIlwraith and Guang-Zhong Yang

Abstract The last decade has witnessed rapid growth of high power, low cost mobile sensing platforms, finding successful application in a range of environments – from industrial process monitoring to structure management. To date, the most challenging deployment arena is the human body and extensive research is focusing on biocompatibility, signal propagation and power management to permit pervasive sensing of detailed physiological signals from implantable, wearable and ambient sensors. By involving users in the management of their own wellbeing, Body Sensor Networks (BSNs) aid in the delivery of preventative care by detecting the onset and systematic deterioration of "lifestyle" diseases. When used for sports performance monitoring, they can also record long-term progress whilst providing real-time training information that can be used to maximise the effectiveness of individual sessions. This chapter provides a detailed review of recent developments in BSNs, highlighting both the technical challenges and deployment issues related to autonomic sensing, context awareness and distributed inferencing.

1 Introduction

As the quality of healthcare in the western world continues to increase, previously leading causes of death that include influenza and tuberculosis have given way to an increase in chronic conditions – most notably cancers, heart disease and degenerative conditions such as Alzheimer's and Parkinson's [37, 18]. One of the main characteristics of many common chronic conditions is that their occurrence can be reduced through a change in lifestyle. For this reason, they are often called *lifestyle diseases*. The prevalence of these diseases is so significant that they are now the major contributors to mortality and morbidity in many countries. In 2006, 35% of all deaths in the UK were directly attributed to cardiovascular disease [3] and around one fifth of these could be linked to smoking. Within the same study, it was reported

D. McIlwraith (✉)
Royal Society/Wolfson MIC Laboratory, Department of Computing, Imperial College London, London UK
e-mail: dm05@doc.ic.ac.uk

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_13,

that 63% of all heart attacks within Western Europe were due to abdominal obesity. Clearly, the adoption of healthy and active lifestyles that include regular exercise and sport will reduce the probability of such complications.

Fortunately, many have now become aware of the benefits that accompany a health conscious lifestyle and are modifying their behaviour accordingly. The use of technology can facilitate this change by involving users in their own health and well-being – providing positive feedback to reinforce active lifestyles whilst presenting user specific targets to be reached. Such involvement allows for a more efficient shift in lifestyle as individuals have a more accurate picture of their behaviour as well as continuous feedback regarding their progress. This can help high risk patients, such as those who are morbidly obese, make a positive change to their lifestyle through achievable increases in energy expenditure and modifications to dietary intake. Such technologies are not restricted in their application and may be used to monitor the periodicity, duration and efficiency of all exercise – providing feedback to amateur and professional sports persons alike. This approach also has the potential to deliver detailed insights into sports technique and enable individual and tailored coaching to maximise athlete performance.

To this end, recent developments in Body Sensor Networks (BSNs) address many of the challenges raised. BSNs consist of a number of miniaturised, low power sensors capable of local processing and wireless communication for distributed, autonomic sensing in a variety of environments. One subject group that stands to benefit from advances in BSNs are the elderly, who represent a unique challenge to pervasive sensing. As general life expectancy has increased more than two – fold over the past two hundred years [105] and evidence suggests that this trend shall not cease or be reversed [88], a significant burden will be placed on existing health-care models requiring a reevaluation of care provision. An attractive solution is to provide effective health management in a community setting using BSNs, reducing the cost of care, freeing hospital beds and affording the patient both a higher level of independence and a better quality of life. This can be achieved through the monitoring of activity levels, analysis of physiological signals and determination of behavioural patterns – deviations in any of which may indicate the onset of a decline in health.

With a current shift towards early intervention and minimally invasive treatment in healthcare, BSNs plays a key part in post-operative assessment and rehabilitation. For laparoscopic surgery BSNs have been used to capture surrogate measures that are indicative of post-operative complication and infection [8, 9, 103]. For minimally invasive surgery, where patients are usually discharged shortly after the procedure and allowed to recover at home, the ability to continuously monitor patients' key physiological indices without impeding their activities (i.e., pervasively and ubiquitously) is crucial for future widespread clinical uptake. Where long-term recovery and rehabilitation is necessary, continuous sensing of gradual and subtle changes to key physiological and biomechanical indices are essential to tailor ongoing therapeutic regimes.

These are just some of the areas in which BSNs have already been utilised and, as devices become smaller, more powerful and more energy efficient, we should

Table 1 Summarising key application areas for BSNs

Area	Applications	Studies
Monitoring	Elderly and post-operative care, out-patient monitoring, skills assessment.	[7, 9, 54, 103]
Sport	Amateur and professional athletics, coaching, physiotherapy, occupational therapy.	[55, 85, 92, 111]
Lifestyle	Human computer interaction, leisure, gaming.	[12, 90, 131]
Healthcare solutions	Cardiac monitoring and regulation, prosthetic enhancement, therapeutic technologies.	[27, 101, 104]

expect to see development in many new and exciting avenues of research. As has been demonstrated to date, BSNs allow for real-time monitoring of physiological, biomechanical and environmental factors related to their subjects and as such, are of interest in many related areas of research. Sports scientists, occupational therapists, physiotherapists, as well as medical professionals and their patients, all stand to benefit from the exciting developments in BSNs which enable continuous monitoring of subjects within their own environment.

1.1 Body Sensor Networks

BSNs consist of a number of locally fixed, worn or implantable sensors with integrated processing and wireless communication capabilities [132]. In general, each processing device may have a number of sensors and actuators capable of user-centric monitoring and modification through the issue of drugs stored in micro-capsules, electrical stimulation or generation of sound and visual stimuli. In addition, an *expert* may mediate between acquired data from the network and actuator stimuli. This relationship has been defined in a purposefully general manner in order to encompass the greatest range of applications. We can observe this by considering that the traditional doctor-patient model also fits within this framework, where sensors are those utilized by the expert (doctor) and feedback is performed through consultation and medication. The purpose of BSNs is not to replace, but to augment this relationship through the design of intelligent networks – offering continuous feedback to subjects regarding their status. What is significant is that the traditional episodic measurement in clinical/laboratory settings is replaced by continuous sensing under normal living conditions. In this way, BSNs can be used for preventative health as well as for sport and training, allowing users to choose actions that impact their health positively.

1.1.1 Network Topology

BSN topology is an application specific parameter in practical deployment. Many early applications of BSN have focused on star topologies with limited in-network processing – where a centralised processing unit that may be worn or external to the subject is responsible for collecting and storing data [66]. Data can then be queried

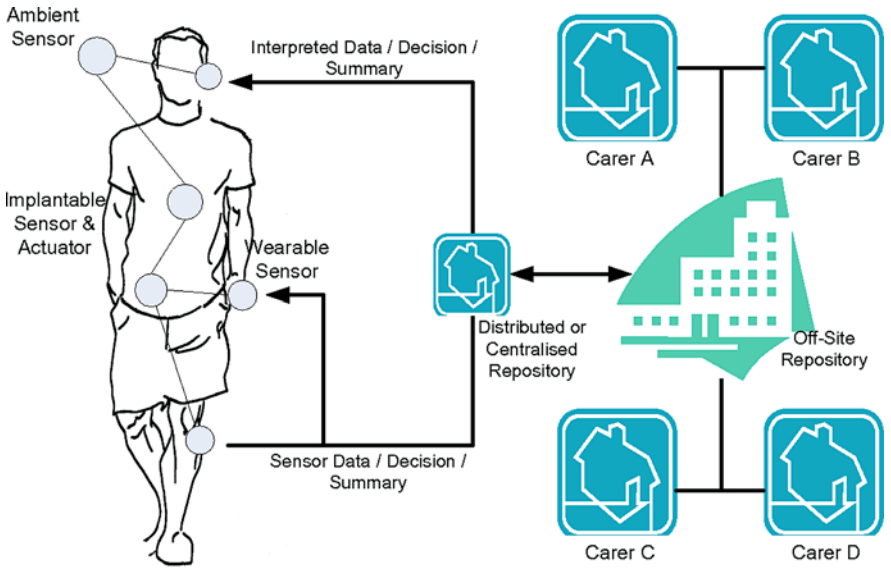


Fig. 1 BSNs consist of a number of ambient, wearable and implantable sensors/actuators with communication links in between. Information is either presented directly or the data is processed in a distributed or centralised manner before returning to the user. A centralised approach may be preferable where the computational requirement of the application exceeds the ability of the sensor nodes themselves. The network can also be further augmented with an additional tier – for example in the case of BSNs for healthcare where on-site hardware may pass data to a hospital repository which may operate upon this further providing storage capabilities and notification of status to formal and informal carers alike. "Standing Man" illustration courtesy of Philip Reed

by clinicians, doctors or trainers to form a complete picture of subject status from which feedback can be provided. For more advanced pervasive sensing, a remote interface can be provided through existing mobile devices [121], allowing medical and fitness monitoring to occur as the user performs routine activities. Not only does this provide a more convenient and unobtrusive monitoring platform but it allows for the acquisition of data less distorted by artificially controlled environments. Other projects, such as SAPHE [2, 78, 93], have utilised a two tier topology to facilitate both home-based and ambulatory sensing. When subjects are on the move, a mobile hub worn or carried by the user collects data from the subjects, sharing this with a fixed device once returned to the home monitoring environment.

Unfortunately, the adoption of a star topology has a number of drawbacks. Since the responsibility of aggregation lies with a centralised element, it must be in frequent communication with all other nodes. This can rapidly drain power sources and limit the period of time which a subject can remain untethered. Furthermore, centralised elements are both bottlenecks and a single point of failure within the network, impacting the throughput and reliability of BSNs. One possible solution lies in the use of intelligent and collaborative *mesh* networks where no single node is responsible for data aggregation and processing is performed in-network. For a more

detailed treatise of data aggregation for sensor networks in general, we refer the interested reader to the previous chapter on "Data Aggregation in Wireless Sensor Networks: A Multifacted Perspective", which provides a taxonomy of aggregation methods as well as a detailed discussion of the practicalities and considerations involved.

Where new sensors can be added to existing networks, how these are identified and distinguished from other network elements is crucial to ensure that nodes do not erroneously share sensitive data with third parties. A model of communication must also be adopted in order to balance power usage, load and frequency of update. Exact communication models will be application and context specific, but it is likely that sensor subnets will share and summarise sensor data before routing to nodes further in the network – to provide the advantages of both topologies as well as varied levels of service delivery.

This highlights the shift from a centralised model to one where intelligence has been distributed and moved into the network itself. To facilitate this, the development of high accuracy inference techniques capable of both distribution and *graceful* failure is required, so that BSNs may continue to provide safe and *reasonable* feedback where resource availability is reduced.

1.1.2 Requirements of Body Sensor Networks

BSNs provide a wealth of new challenges in the co-ordination of distributed resources as they must exhibit a degree of autonomy once deployed – for issues of transparency, accessibility and convenience. To address these issues systematically, the phrase *autonomic sensing* [126] has been used. Autonomic sensing is characterized by the *self*-* properties summarised in Table 2, and has been defined as:

Table 2 Summarising the *self*-* properties of autonomic BSNs [132]

self-* Property	Definition
Management	The ability of a system to know the *extent* of itself – its components, status and limitations.
Configuration	The ability of networks to respond to changes in operating environment without user interaction.
Optimisation	The action of continual optimisation dependent on dynamic context.
Healing	The property of a system which can detect and preempt problems and modify resource usage to provide uninterrupted services in light of these problems.
Protection	The ability of mission critical networks to preempt and respond to attacks from malicious parties.
Adaptation	Allowing the network to achieve long term goals through adaptation of operating techniques and procedures.
Integration	Providing methods for seamless interaction with other nodes and networks that are present within the operating environment.
Scaling	The ability of a system to prepare itself for new events and milestones within its life.

> An approach to self-managed computing systems with a minimum of human interference. The term derives from the body's autonomic nervous system, which controls key functions without conscious awareness or involvement [49].

It is worth noting that these properties are not mutually exclusive and arguably have a great deal of overlap, however they are required to highlight the requirements for BSNs to work with minimal user interaction. For a system to be truly autonomic, it must survive in the presence of both errors and interference – adapting to sensing requirements whilst providing optimal or near optimal services – much like a biological entity. In the following sections, we shall discuss the operating modes of BSNs and highlight the importance of autonomic operation.

1.1.3 Operating Modes

As we have noted, BSNs are now applied to a variety of problems, from exercise monitoring and surgical skills assessment to elderly care. Through analysis of emerging trends we note that in general, applications may be classified into one of three categories: *restorative*, *enhancing* and *diagnostic* [39] – each with its own unique set of challenges.

Restorative

As suggested by the name, *restorative* BSNs return a level of function that has previously been lost. Consider the case of glucose monitoring and insulin delivery in the treatment of Type I diabetes mellitus. A closed loop BSN can emulate the islet beta cell activity of a healthy patient, providing a long term *cure* for the condition [104]. Cochlear implants [24] also provide an example since, through the collection of audio signals and direct stimulation of the auditory nerve, the BSN provides an "electrical ear," restoring a level of hearing to the deaf patient. Optical nerve stimulation [47] provides restorative treatment for blindness in some cases, creating visual cues through direct stimulation of the optic nerve or retina, for example where sight loss has been caused by outer retinal degeneration.

To date, many restorative technologies can provide only an approximation to original bodily function. In the case of hearing loss, cochlear implants have advanced only far enough to use a small number of electrodes to stimulate the auditory nerve and thus, when compared to the thousands of hair cells this attempts to mimic, the quality of sound can be poor. Optical stimulation is similarly approximate, with users only able "see" a series of luminescent dots they must learn how to use these new visual cues.

Enhancing

BSNs that provide additional capability to a user or improve the users existing functions are defined as *enhancing*. It is likely that the advancement of restorative technologies such as those stated above will eventually reach the level where the functionality they provide exceeds the natural ability of the patient – this would

raise important ethical questions for the subject and society as a whole. More immediate examples of *enhancing* BSNs augment the ability to perform particular tasks and as such have drawn significant military funding. Under the DARPA ASSIST (Advanced Solider Sensor Information Systems Technology) project [113], BSNs have been created to recognise and log the activities of soldiers in real time, providing a valuable tool for information sharing and debriefing [81]. To improve physical function such as strength, exoskeletons have been developed with which BSNs are an integral component. For example, the *Robot Suit HAL (Hybrid Assistive Limb)* [45] uses a collection of body worn sensors and actuators in order to enhance the effective strength of the wearer. Potentiometers at each joint measure angle and use the calculated centre of gravity to stabilise the wearer – this allows user to move unimpeded and lift objects that would be too heavy to move unassisted.

Diagnostic

Rather than restore or provide additional function, diagnostic BSNs have been proposed to augment the ability of medical professionals to determine the onset of disease. In the study of intermittent heart complications Insertable Loop Recorders (ILRs) [57] provide an implantable and untethered solution to cardiac monitoring. In other areas of diagnostic sensing, wireless devices have been used to assess the blood flow to a particular organ or region of tissue [32]. This can provide an early indication of infarction in by-pass or transplantation surgeries facilitating intervention before tissue death occurs. Such technologies may also allow for blood gases to be measured in vivo.

It should be noted that the categories above are by no means mutually exclusive. For example, exoskeletal systems can reroute force transmission to provide enhanced lifting capabilities however, under different circumstances, these devices could be considered *orthoses*, enabling those with muscular or skeletal deformation to walk by reducing the load placed upon their joints and limbs.

1.2 Sensors and Modalities of Sensing

Due to the complexity of the human body and the number of parameters available for monitoring, health and wellbeing is extremely difficult to assess. With respect to sensing modalities, *biomechanical* features are crucial for the analysis of movement as these provide an insight into the efficiency of movement for sport, recovery in post-operative patients, as well as the wellbeing of elderly patients at risk of neurological disease. To this end, *ambient* and *visual* sensors may provide a less obtrusive approximation whilst at the same time incorporating contextual information. Since biomechanical, ambient and visual sensors only provide data regarding the external manifestation of subject state, *physiological* sensors are required to capture a complete picture of the body. Candidate systems for physiological monitoring include the nervous, circulatory, respiratory, gastrointestinal, integumentary,

urinary, immune and endocrine [100] and these are discussed within the following sections where relevant to the deployment of BSNs.

1.2.1 Biomechanical

To monitor local motion, such as that of the limbs, angular displacement at each of the joints may be observed using Electric Goniometers (EGs). EGs return varying electrical voltage dependent upon angle of displacement and have been used successfully in many studies [58]. Such devices are cheap to manufacture and, provided they are placed suitably, can provide an accurate insight into the motion of the body during activity. When coupled with wireless networking, monitoring becomes untethered allowing for in-place sports monitoring, however such devices may not prove suitable for contact sport where they may be unintentionally removed from the subject.

Accelerometers have been extensively investigated in order to determine motion as they provide an inexpensive solution and result in intuitive data. Through the placement of accelerometers on the limbs it is possible to determine their motion along with the impulse on external objects. When worn on the body, individual signals can provide important information regarding subject sway and displaced energy. Unfortunately, data obtained can often be difficult to analyse as accelerometer signals are composed of static and dynamic acceleration which cannot be separated without additional technology. The former is the acceleration due to the earth's gravity, and is therefore dependent upon the orientation of the sensor. The second is the induced acceleration due to motion. Whilst in theory the integration of accelerometer signal should provide velocity, then displacement once integrated again – the composition of the signal and the inclusion of noise leads to a high level of drift that renders the calculated displacement inaccurate. Because of this, the additional use of MEMS gyroscopes has been suggested. Although such devices typically suffer from similar issues of drift, it has been shown possible to correct for this using accelerometer data, by frequently resetting the integration baseline [55].

To detect the transmission of forces through the foot when running and walking, the use of force plates has been adopted. This allows for gait cycle to be analysed in detail, such that the point of initial contact with the floor can be determined along with the subsequent distribution of weight over the foot. This has proved beneficial for many different applications, from analysis of walking aids, to rehabilitation and the analysis of running style.

To provide a general sensing platform for biomechanical signals, researchers at Imperial College London have proposed the e-AR (ear worn, Activity Recognition) sensor, illustrated in Fig. 2, based upon successful experimentation with the ear mounted BSN platform [65]. Inspired by the location of the body's own orientation sensors within the inner ear, the e-AR sensor is ideally placed to capture the orientation of the head and acceleration of the body as well as the transmission of force throughout the wearer.

Fig. 2 The e-AR sensor developed at Imperial College London comprises a Nordic nRF24E1 chipset, built-in 2.4GHz RF transceiver, memory, and an analog-digital converter for data sensed using a 3-axis accelerometer. This is based upon a miniaturised design of the BSN node [65]

1.2.2 Ambient and Visual Sensors

Accurate motion capture can be performed with optical technology, using one of many commercial products available from Northern Digital Incorporated, the Motion Analysis Corporation, Charnwood Dynamics (CodaMotion) and many other companies. Such systems are excellent for detailed analysis of movement since they are both accurate and can capture data at a high frequency. With suitable placement of markers, imaging systems can allow both linear and angular displacement to be calculated and, when combined with models of muscle motion and exerted forces, force transmission and joint strain – important factors for sports analysis, rehabilitation and patient assessment. Unfortunately, such systems are of limited use for highly mobile applications as they require multiple cameras, many markers to be worn and typically only have a limited area of coverage within the range of several metres.

For situations where detailed motion analysis is not required, Passive Infra Red (PIR) can be employed to give an indicator of movement within a room. Whilst such limited information may be useful when a high level of privacy must be maintained, the type of motion performed by the subject cannot be captured rendering these devices of limited use for activity recognition. One potential solution, offered by Lo et al. [67] is to process the scene using a statistical model of the background [62] and represent subjects as binary blobs. This has the effect that detailed subject information, such as appearance and identifying features, are removed before further processing occurs – negating the need for potentially sensitive video information to be stored and transmitted. Whilst background segmentation allows overall subject pose to be extracted, crucial for activity recognition in healthcare environments, self occlusion can provide misleading blob output and remove important movement information such as that of the limbs. For this reason the use of optical flow, to

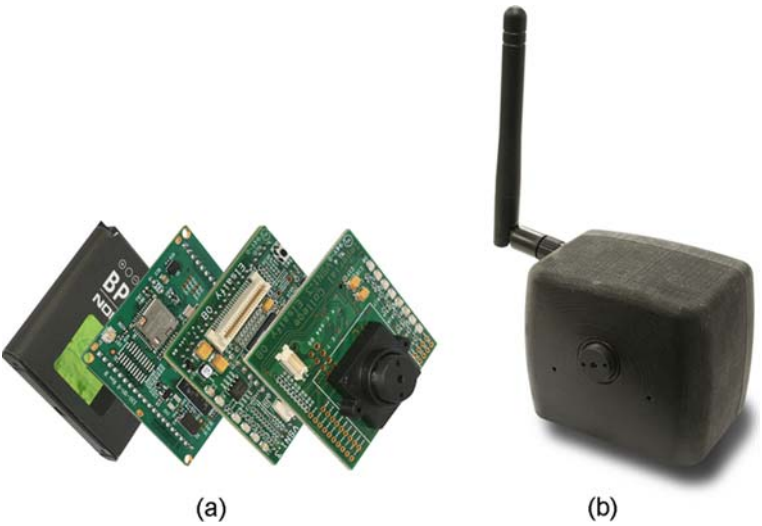


Fig. 3 The ambient blob sensor comprising an image sensor, Blackfin on-board processor, wireless transceiver and battery. An exploded view of the device can be seen in (**a**). Once packaged, (**b**), the blob sensor can be integrated within the home using a wall mounted bracket, similar to the installation of a PIR sensor. Image data is processed on-board and in real-time, communicating derived metrics only to other devices in the vicinity. This provides a significant advantage in terms of power consumption and ensures potentially sensitive video data is not disseminated

capture motion within the segmented blob, has been proposed [93]. This device, known as the *blob sensor* is illustrated in Fig. 3.

1.2.3 Respiratory and Circulatory Monitoring

The interrelated respiratory and circulatory systems offer a rich source of data when monitoring for sport, healthcare and wellbeing. Both individual performance and overall fitness can be determined through the calculation of energy expenditure – and for healthcare potential respiratory and cardiac problems can be screened for before symptoms become manifest.

In the care of cardiac disease, bedside monitoring of heart rate parameters have typically involved the use of electrocardiographs (ECG) to measure the electrical activity of the heart through chest mounted electrodes. ECG output can be utilised to rule out damage and disease of the heart muscle as well as measure cardiac indices. Where symptoms of disease are episodic, wearable or implantable devices [95] can capture abnormal events as part of an outpatient procedure. Several wearable commercial systems aimed at providing remote monitoring for high performance, high risk groups, such as the emergency services and military [83], have also been proposed.

In the study of respiration, spirometry is commonly used to measure both the volume and rate of air transferred through the lungs. This can be effective for the

diagnosis and assessment of diseases impacting the respiratory function, such as Chronic Obstructive Pulmonary Disorder (COPD), emphysema, asthma and cystic fibrosis. Due to the nature of this test, spirometry parameters cannot be obtained continually, since the procedure is episodic by nature. Mobile devices have been designed however, which attach to a mobile phone [35] and could prove valuable in providing context to continuously acquired variables as part of a larger BSN.

The piezoresistive effect, which imposes a change in resistance due to mechanical stress [94], is used to determine breath rate, whilst other systems proposed have used Polypyrrole (PPy) coated foam sensors [17] and ultra-wide-band (UWB) radar [114]. Impedance pneumography has also been used successfully [16] in which an alternating current, transmitted between two chest mounted electrodes, is used to determine thoracic change during respiration. This is performed by monitoring the voltage change across the electrodes which is subject to variance between inhalation and expiration. Whilst a relatively unobtrusive monitoring method, this does not allow for the volume transferred to be calculated and can be subject to motion artefacts due to changes in body position and orientation. To address these issues, inductive plethysmography was proposed [128] to monitor the volume of the thorax during respiration – which has a direct relationship to the volume of air transferred. This is performed using wearable bands, producing a magnetic field that induces voltage dependent on the magnitude of chest expansion.

Photoplethysmography (PPG) has seen significant interest for both the monitoring of respiration and cardiac rhythm [51]. PPG operates by observing either the transmissive or reflective wavelength absorption as light is transmitted through, or reflected off, the skin – typically on the finger or earlobe. In the case of transmissive PPG, the light from two LEDs (of different frequencies) passes through the extremity and the wavelengths of the resulting light is analysed. Since the difference in wavelength absorption will depend on both the blood engorgement of the extremity, and the composition of the contained blood, both pulse rate and oxygen saturation can be extracted. Recent research proposes reflective PPG for the e-AR sensor [68, 123, 124, 99], however dealing with motion artefacts remains the most challenging research question.

Near Infra Red Spectroscopy (NIRS) utilises the near infra-red region of the optical spectrum which exhibits relatively low absorption by the tissues of the body – allowing for much deeper tissue penetration than with PPG. Dependent upon the absorption characteristics of the light it is possible to build detailed maps of oxygenated sites. Consequently, NIRS has seen significant interest as a method for brain imaging [122] and as a tool for sports monitoring to measure blood flow and oxygen consumption by the muscles [29, 28, 41, 63].

1.2.4 Neural, Biological and Chemical Analysis

In the field of neural sensing, several previously mentioned technologies are finding important clinical application. NIRS has been used to map brain function indirectly by detecting oxygenated regions within the brain, whilst electroencephalograms (EEG) perform the same function through the capture of electrical activity. Although

used extensively in clinical scenarios, for example to characterize seizures [82] and to monitor brain function in comatose or anaesthetized patients, EEG is beginning to find application for the control of personal computers [131], prosthetics [42] and even bridge the gap between brain and appendage, allowing those who have lost motor function due to nerve or spinal damage to regain a level of function through Functional Electro Stimulation (FES).

Blood gas and pH sensing is routinely carried out on blood taken during in-patient procedures. With this technique, the concentration of oxygen and carbon dioxide is determined which can be important in the care of pre-term babies and the critically ill. By measuring pH, general information regarding the metabolism can be garnered, allowing for indirect analysis of kidney function. Currently, blood gas analysis is performed periodically – requiring blood to be drawn from the patient and processed either in a laboratory setting or using one of many emerging point-of-care systems. Unfortunately, whilst bedside systems reduce the turnaround time in obtaining results, such systems still only offer a snapshot of that person's status and, since blood gases can change rapidly, it would be advantageous to obtain continuous, real-time data. To this end, in vivo blood gas monitors have been proposed [44, 80, 118] and trialled [127]. Such technologies, fuelled by advances in micro-electronics, have ignited interest into chip designs that integrate several laboratory functions (lab-on-a-chip) to screen for many health related issues at a single point of a care. For example we may be able to deliver virus screening to developing countries at a fraction of the current cost, allowing for early detection and treatment that slows the progress of infection [43].

For chemical excretions, the use of smart clothing [71] plays an important role for many practical applications. Coyle et al. [26] have proposed an approach that uses dyes optically sensitive to the pH of the absorbed fluid. Consequently, dependent upon the reflective properties of the clothing, sweat pH can be obtained. As well as detecting properties of the wearer, such technologies may also be used to sense parameters of the environment providing invaluable piece-of-mind for workers in hazardous environments [110].

1.2.5 Implants, Ingests, Actuation and Feedback

A variety of implants have been discussed within the research community, leading to many clinical trials and commercially available products. ECGs [57], pacemakers [48] and cardioverter-defibrillators [101] are all now implanted during routine procedures performed around the world for a variety of conditions. Cochlear implants have continued to improve since their inception with recent studies targeting the modiolus [24], *enhancing* residual hearing rather than destroying it as with previous systems.

Retinal implants, that restore vision to sufferers of diseases such as macular degeneration and retinitis pigmentosa, are a relatively new advancement in implantable technology. Where nerve pathways linking the brain and the eye remain healthy, the retina may be artificially stimulated to simulate vision. To date, several such devices, which process images captured from an externally mounted camera

and transmit these to electrodes mounted on or underneath the retina, have reached clinical trials [47]. This technology is still in its infancy and, as the number of electrodes used for stimulation is typically several orders of magnitude smaller than the number of ganglion cells stimulated by light entering the eye, users report that they can detect features such as edges and light points but the level of detail is low. With further development we may expect to see restorative vision which is a much better approximation to that lost through degenerative conditions. Where the optical nerve has become diseased or the eye has been damaged or lost, a form of vision can still be provided through the use of cortical implants [72].

As discussed above, functional electro-stimulation (FES) can restore motion to incapacitated patients in a variety of scenarios. FES has been used both internally and externally to stimulate muscular contraction for sufferers of drop foot [70], characterized by a difficulty in turning both the ankle and toes upwards, strokes [27], who may have partial or total loss of function in their body, and incontinence [73]. FES systems are typically separated into a sensor and stimulator – either of which may be implanted. Sensors attempt to determine the users intention and provide the ability to act upon this through muscle stimulus. In the case of drop foot, systems use inertial sensors to determine which part of the gait cycle the user is currently engaged in, communicating with the stimulator in order to clear the toes from the ground before impact. For the treatment of stroke and spinal damage, implantable sensors can bridge the gap between intention and action through sensors implanted within the brain or spinal column. Ultimately, fully implantable, closed loop FES could be used to restore full movement to mobility impaired candidates by providing artificially implanted nerve pathways [69].

Solutions to insulin delivery have been sought to improve the convenience of glucose testing and ensure the accuracy of insulin delivery in the treatment of Type I diabetes. Traditional therapy involves the use of long acting insulin taken at regular intervals to regulate blood glucose level during normal routine. Once determined, sufferers are required to calculate and administer the amount of insulin required to break down carbohydrate introduced at meal times. Such an approach is time consuming as it requires frequent testing and delivery of insulin. Several advances have been made in order to automate this process, most notably by using insulin pumps attached to a subcutaneous delivery system operating at a *basal rate*; to replace long acting insulin. Short acting insulin still needs to be introduced after meal times although this can be performed through the existing delivery system – negating the requirement for frequent injections. Significant research is ongoing in order to close the loop for insulin delivery such that the patient no longer need consider their condition [104]. In this way an artificial pancreas that simulates healthy function could be provided using implantable sensors and delivery devices.

In the field of imaging, many researchers are turning to ingestible devices to perform imaging of the internal structure of the body, for example the small intestine, which is unreachable in its entirety through endoscopy. Such procedures are painless and allow the upper gastrointestinal (GI) tract to be screened for polyps, which if left untreated can become cancerous [129]. Whilst unreported, we may imagine the combination of imaging devices with *lab-in-a-pill* technology to create a detailed

overview of GI health, sensing parameters such as temperature [74] and pH [50] while also recording images during transit through the body. Figure 4 provides a graphical overview of aforementioned technologies, indicating the typical locations within the body that sensors would be deployed.

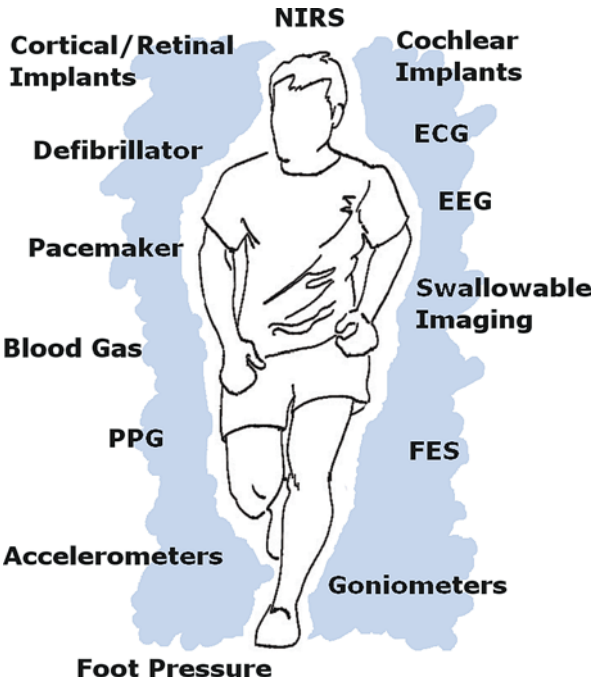


Fig. 4 Illustration of sensing modalities and their physical location within BSNs. "Running Man" illustration courtesy of Philip Reed

1.3 Directions and Challenges

The use of in vivo sensors poses two unique challenges not encountered in other areas of remote sensing. Firstly, the development of suitable biosensors, which respond to the parameter of interest and act as an interface to digital processing and communication devices, is required. Secondly, issues of biocompatibility are raised with respect to both the sensor itself and the supporting circuitry so that hardware does not create adverse effects with the body's immune system when left in situ.

For extended deployment, issues of power management are important since implantable devices are not readily accessible for battery replacement or charging. Possible solutions include the use of an inductive coil, however, to remain power efficient these are only feasible for implants residing close to the skin. More attractively, the development of energy harvesting techniques would provide an indefinite solution without the need for supporting hardware external to the body. The use of

biofuel cells that can draw sufficient power from biological sources such as blood glucose is an active area of research with promising initial results [46]. Furthermore, depending on the site of implantation both tissue type and cardiac-respiratory motion will have an effect on both attenuation and propagation characteristics of wireless signals. Suitable solutions for successful in-body communication are likely to involve collaboration in the fields of sensor-placement, antennae design and wireless communications.

There are further challenges for practical deployment, since the data obtained is user-centric and must be treated as potentially sensitive – this highlights the need for effective but lightweight transmission security which will comply with hardware restrictions and energy expenditure constraints. Also, as data can be readily moved, copied and applied, there should be strict control over who has access to the data and what purposes it can be used for. For successful adoption of BSNs, where deployed in a specific capacity with patient agreement, monitoring outside of this scope should not be allowed to occur.

Perhaps the most important challenge is to create autonomic BSNs that operate with minimal interaction from users and are able to respond to changes in both the environment and the subject. To date, research in this area has focused on the realisation of the *self-** properties outlined in Table 2, for example self-managing and self-healing BSNs, and has typically been addressed through research into the use of context. To create truly autonomous networks, BSNs must acquire large volumes of data which may require intensive computation for context inference, creating a significant challenge in the provision of resources. In Sect. 2, we shall discuss the issues of context and autonomicity further, reviewing current research in these fields and discussing their impact on future developments in BSNs.

2 Data Modelling and Pattern Recognition

In the previous section, we have provided an overview of BSNs and their structure along with a review of current an emerging technologies in wearable, ambient and in vivo technology for user-centric data capture. In this section, we provide a view of BSNs and emerging issues from an algorithmic standpoint – Fig. 5 provides a road map of research challenges in data analysis from initial knowledge of the phenomena, through to decision output. Where topics appear in parallel this denotes that issues are inter-related. Importantly, for the development of useful pervasive systems, the correct set of procedures must be applied together and with the application in mind. To this end, the pervasive sensing paradigm must be considered holistically. The techniques applied at each step towards inference and their subsequent effects should be considered carefully and result from domain specific knowledge held by pervasive system designers.

In the field of BSNs, data analysis issues fall within one of three key categories: *signal processing and reconditioning*, *sensor placement and feature selection* and *data modelling and inference*. The former describes low level issues regarding the phenomena under investigation and the processing of data to extract relevant infor-

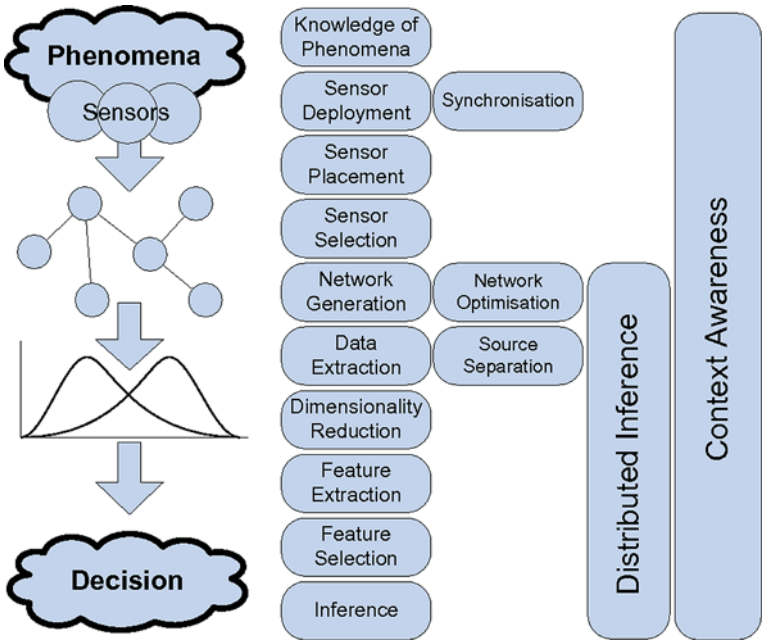


Fig. 5 Illustrating the key research areas for data processing within BSNs – from initial knowledge of the phenomenon through to decision level output. Where topics appear in parallel, this denotes that they are interrelated

mation. For example, within this category, data may be processed in order to separate contributions from multiple sources, or features extracted which succinctly summarise the data. Within the second category, the magnitude of data may be further reduced through the selection of the most relevant features or feature sets from particular sensors for the decision task at hand. In the final category, the underlying patterns within the data are extracted using model based or statistical techniques so that patterns may be learned and decisions drawn from previously unseen data.

2.1 Signal Processing and Reconditioning

Where phenomena cannot be extracted cleanly regardless of sensor type and placement, *source separation* may be used [14] to identify the elements of a compound signal. A practical example has been proposed by researchers of home-healthcare systems where multiple occupants may be housed [78]. In this work, a multi-sensor scenario is considered, which has been shown to increase the accuracy of activity recognition [93]. Data corresponding to the same subject is matched from different sources using a feature selection technique that identifies sets with discriminatory overlap – maximising the likelihood that activity classification will agree when sensors are used independently.

Where multiple sensors are used, synchronisation must be given careful consideration. For clinical applications, the correlation of data is important for practitioners searching for causal relationships between patient symptoms as they rely on data timestamps to provide an insight into patient deterioration. Within sport, victory can often be forfeit in a fraction of a second, therefore trainers and athletes alike require high rate, high accuracy synchronisation to leverage the most from their training regimes.

Post data acquisition, features are extracted to capture specific characteristics of the data whilst providing a compact representation. Although automatic *feature selection* algorithms exist [64], detailed domain specific knowledge of the data set and the potential trends contained within will help in the generation of candidates before the application of such algorithms. A further possibility is the application of dimensionality reduction techniques such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Multidimensional Scaling (MDS) or Locally Linear Embedding (LLE). Such methods reduce the storage and transmission requirement of the data whilst retaining its discriminatory power.

2.2 Sensor Placement and Feature Selection

With knowledge of the observation domain, BSN designers may address issues regarding sensor placement within the environment and on the body, [96, 15] as well as the type of sensors to be deployed. Various research within the literature has addressed the problem of sensor selection in pervasive sensing generally [86] and more recently for BSNs [53]. Whilst sensor selection may occur as an investigative step in application design, before sensors are deployed permanently, it may also be performed with the sensors in situ and dependent upon context. Consider the selection of sensors to determine subject activity. During the investigation stage sensor selection ensures that sensors are placed to capture the widest range of activities however, during the data acquisition stage, a subset of these may be activated to identify individual activities.

2.3 Data Modelling and Inference

For successful data modelling, the phenomena under surveillance must be well understood. Without detailed knowledge of the underlying problem, application designers will fail to collect basic data relating to the process under observation and may misunderstand the relevance of the data that is acquired. Whilst the necessity for detailed requirements analysis is true of any large project, this can be especially important within BSNs due to the complexity of the subject and the large number of parameters available for collection. Although tempting to collect as many parameters as possible and display these back to the user, this can create sensory overload, overly confuse the problem and create difficulties in automation for closed loop systems.

Sensor data or data derivatives can be passed to clustering or classification algorithms for inference, with the requirement that such algorithms should extract the underlying patterns behind the observed phenomena. Such algorithms can assume prior knowledge of the domain, for example in the case of Bayesian Networks, or may be completely statistical in nature. Of specific importance for healthcare and sport are temporal models such as Conditional Random Fields (CRFs) [60] and Hidden Markov Model (HMM) variants [34] as these can model multi-stage activities or behavioural traits that evolve over time. Algorithm choice is dependent on many factors including data type, data processing techniques and the magnitude of data available.

Traditionally, classification algorithms deployed for inferencing in sensor networks have been centralised and applied after data has been aggregated from all sources however, as the computational power of nodes within wireless sensor networks increases, so does the possibility for the distribution of such algorithms. Not only does this remove central nodes as a potential bottleneck and a single point of failure but it provides a more scalable solution – allowing for the development of modular networks which can accept requests from heterogeneous sensors to share the computational burden of inference.

Typically, the distribution problem is considered from one of three standpoints, taken from the machine learning and sensor fusion communities. *Data level*, *algorithm level*, or *decision level.* Within the first level, data is shared amongst sensors so that elements of the network have access to data from many disparate sources, *before* the decision analysis occurs. In the most naïve implementation, data would flood the network with the effect that each sensor has access to all data – sensor nodes are then free to reach their own decisions independently of all others. Unfortunately, such an approach does not leverage the heterogeneity of the network, creates a high level of redundancy and requires frequent communication between nodes. A refinement of this approach defines aggregator nodes to compactly represent regions of data, as discussed in chapter "Data Aggregation in Wireless Sensor Networks: A Multifaceted Perspective", however nodes may become overspecialised leading to partitioning of the network during sensor failure.

In order to provide a solution to this problem, significant research has occurred at the *algorithm level*. This requires the decision analysis stage of processing to be pushed within the network such that each element is responsible for some stage of a decision that is reached. This entails a high level of coordination from each network component, ideally with awareness of network delay, power usage, processing, memory requirements and quality of service guarantees. To date, research has investigated distributed pre-processing [30], statistical classification, [87, 10] and Gaussian mixtures [40]. Supervised learning of decision boundaries has also seen some interest in the last few years and we refer the interested reader to the following review [102]. For probabilistic graphical models using prior knowledge, various methods have been proposed to learn structure in a distributed fashion [61] and which leverage the similarity of probabilistic inference over tree structures to the distribution of the sensor nodes in physical space [98, 97]. More recently, distributed

methods have been proposed for approximate probabilistic inference in dynamic systems where transient sensors exist [36].

Whilst a promising area for autonomic sensor networks, due to the complexity involved in the deployment of existing algorithms there have typically been few practical implementations. Consequently, the future of BSNs may be dominated by *decision level* distribution, especially as the individual power of sensor nodes continues to increase. Recent work [125] has proposed an ASIC comprising digital control with Neural Network core implementing a Self Organising Map (SOM) [106]. Neural weights within the SOM are learned off-line before being imported to an ASIC which is exposed to data gathered from eleven subjects wearing an e-AR sensor and performing routine tasks [124]. The results presented show high accuracy for the tasks performed, illustrating the potential for on-node activity recognition. This may impact the direction of distributed sensing, as inference is performed on-node by independent sensors and the results are shared over a communications protocol to enable decision level fusion at each sensor. Since data is shared only at the decision level, such an approach fails to leverage inter-sensor data interaction for classification. Consequently, where patterns are present and interact over several sensors, naïve classification will almost certainly fail. This highlights the importance of *preparation* when designing BSNs – as detailed understanding of the problem would preclude decision level fusion for such patterns, requiring the need for *algorithm* or *data* level fusion to occur. In reality, BSNs of the future are likely to provide distribution on one or more of the aforementioned levels dependent upon the application at hand.

2.4 Context Awareness

The use of multiple sensors creates a rich dataset which can be leveraged using machine learning techniques to provide accurate decision level output, however it may be beneficial or necessary to consider the underlying relationship between data from different sources. Often it is the case that, when data is considered in light of other sources, important trends can be revealed and erroneous results removed: this is known as *context awareness*. Context-aware computing is a model where environment, or context, is a considered factor in the design and use of BSNs. An exact definition remains elusive, and is still subject to debate in the research community [112, 22, 108, 31]. Dey and Abowd [1] overview the development of the semantics of context in computing and what it means for an application to be context-aware.

Since BSNs are user-centric, context is defined in terms of the subject under observation and can consist of many factors. For health and wellbeing, temporal factors such as the time of day, week and year can all provide explanations for behavioural changes whilst environmental factors such as the temperature can explain sudden deviations in activity levels, for example, people are less likely to venture outside on a cold day. In the observation of recovery, past medical complications, age, gender, lifestyle and background may all affect the “normal” baseline

recovery rate and so may be considered contextual in this sense. Within sport, what is defined as context will depend on the level of the athlete in question and the sport undertaken. Training regime, how recently training was conducted and the wind direction and strength would all affect the performance of a sprinter for example, but it is likely that changes within many of these parameters will produce only a small difference for elite athletes and possibly negligible effects for amateurs.

It is worth noting that *context* in general is difficult to define succinctly and can vary across applications – thus it is the responsibility of application designers to build appropriate models that factor variables which may not be explicitly defined or may be assumed by practitioners at the application level. If we return to Fig. 5, we see that *context awareness* has been included at every stage of the data processing pipeline, this is because context may require consideration at every stage of BSN design and execution. In extreme cases, a change in context may challenge designers knowledge of the phenomena requiring the deployment of extra sensors, or the relocation of existing ones. Operating under the assumption that all relevant data is captured, context changes can affect data selection techniques, network generation and inference models. This highlights the importance of a holistic approach to BSN design with each step of the data processing pipeline considered in terms of all others as well as the target application.

Current research in this area [56] has concentrated on modelling context at the *inference* stage of the processing pipeline, with context represented explicitly as a parameter in the decision making process. In [119] the concept of a Spatio-Temporal Self Organising Map (STSOM) is introduced, augmenting Self Organising Maps (SOMs) with temporal pattern recognition. STSOMs track neuron activation patterns in SOMs in order to perform activity recognition that could influence sensed physiological parameters such as heart rate. Temporal models such as HMMs have also seen significant treatment in the literature since, by encoding the progression of output symbols through the use of hidden states, they provide a representation of temporal context. Conditional Random Fields (CRFs) can be seen as a generalization of HMMs that can encode context from much further in the past, as well as the future, and have therefore seen interest for use in context aware applications [120].

3 Emerging Applications

Due to the ability of BSNs to provide a detailed picture of subject status, we have seen their successful deployment for a variety of applications. In this section we provide a review of BSNs within sport, healthcare and wellbeing discussing emerging work as well as future directions for user-centric sensing.

3.1 Sport

As the research surrounding sports science has begun to mature, we are seeing a significant impact on the playing fields – narrowing the margins between winning

and losing at the highest levels of competition. Consequently small modifications to technique and training patterns can have a substantial impact on competition outcome. Furthermore, as the popularity of professional sporting events has grown, so has the size of their supporting industries. Professional athletes and their sponsors now have greater financial motivation to continue to push the boundaries of skill and physical capability – to be faster, fitter, stronger and more technically able than the opposition. In this section, we identify three key areas in which BSNs can contribute to the performance of athletes and maximise the effectiveness of their training.

Firstly, in order to provide the necessary energy for intense training schedules, the nutritional composition of food intake is an important factor to ensure athletes have the required energy to perform along with the necessary conditions to recover quickly and maximise the overall positive affect on their physiology. Secondly, an understanding of training techniques and their nature, periodicity and level of exertion is required to ensure maximal physical development without injury and prepare athletes for competition. Finally, the refinement of technique within each individual sport is crucial in order to promote efficient movement delivering expended energy in such a way that it has the greatest affect on the outcome of the competition. For professional athletes to perform at their highest level each of these areas need to be considered in light of all others.

All aerobic regimes should include easier and more intensive workouts to provide a solid basis of aerobic fitness and build up stamina within the core body. The latter helps build overall muscle strength while lighter sessions allow for the recovery of overworked muscles, developing both tone and strength. To this end many devices to assist training schedule are currently on the market, ranging from the Nike + iPod training assistant, offering audio feedback on distance through a foot mounted accelerometer and wireless link, to the Polar S-725X Heart Rate monitor which is user programmable with a multitude of training regimes. These devices have been developed as an aid to training however the loop remains open – the effectiveness of their training schedule is assessed off-line and it is up to the wearer to effect any changes themselves. In the near future, we may expect to see devices to self assess the efficacy of training schedules and provide recommended changes where appropriate. To this end, some basic systems exist, [107, 90], however, the metrics used to assess success for a given sport are not simple, and for more complex training regimes a wealth of knowledge is used by coaches to assess physiological impact. Such information would have to be distilled into the intelligence of BSNs in order to provide the most effective training aids.

Until now, methods for perfecting the technique of athletes have relied upon the experience of coaches to determine the most effective way to perform a particular exercise. For example, a rowing coach will analyse the motion of a rower from the riverside and suggest actions that will not only improve technique, but also help the athlete understand *why* such actions should be undertaken. Because such techniques are subjective, it can be difficult to evaluate the effectiveness of individual coaches until the season or competition that is being trained for is complete and the results have been collated. Furthermore, different athletes are likely to respond in different ways to the same coach and thus their effectiveness is highly subjective. What may

Fig. 6 Monitoring the biomechanical and physical indices of a runner using three modalities. An e-AR sensor worn by the athlete captures sway and general motion of the head and body whilst the Parotec [133] foot pressure sensor details the specifics of subject gait, for example – which part of the foot strikes the ground first and the force distribution as the next stride is taken (*top right*). The foot sensor can be seen in detail in the *bottom right* of the picture. Additional information captured includes the volume of inhalation/exhalation and the gas composition which aids in the calculation of energy expenditure [25]

be required is a more scientific approach to training which evaluates athlete efficiency allowing for the modification of technique based upon tangible performance metrics. Such devices, based upon biomechanical theory, would have the advantage that they could be used by coaches as a training tool, or individually – by amateur and recreational athletes to maximise the efficiency of individual training sessions. In the literature, analysis of technique has been performed for a variety of activities. Swimming [89, 116], skiing, [79, 19], martial arts [23] and climbing [92] have all seen the use of technology to provide quantitative analysis of technique as opposed to the qualitative approach offered by one-on-one coaching.

Fig. 7 Typical rowing technique for one complete "stroke". Athlete starts at the "back stops" position (**a**) with the back straight, legs locked and hands below the pectoral muscles. The oar has been removed from the water at this point. During the next part of the stroke, the oar is pushed away smoothly by the hands (**b**) before the back and legs are engaged, (**c**) and (**d**). When the rower reaches their most forward position, toward the end of phase (**d**), the oar is place in the water with a small upward movement of the hands and the legs are engaged to drive the oar through the water. As the rower reaches the rear of their slider, the back and arms are engaged fluidly, and in that order ((**e**) and (**f**)) before the oar is removed from the water with a small push of the oar handle downward. "Back stops" is then reached again, completing the cycle (**f**)

As mentioned above, one application for sports monitoring using BSNs is that of rowing, illustrated in Fig. 7. Due to the distances covered by crews, traditional coaching techniques have involved either bank-side parties cycling with the crew or carried by low-wake motorized boats, who then offer coaching via mega-phone. Unfortunately, both of these techniques require both physical and mental effort on behalf of the coaches, placing them under undue stress and redirecting attention from their crew. Furthermore, the debrief offered to individual crew members at the end of an outing is not substantiated with evidence, based only upon the subjective feel the coach had regarding performance – and where boats contain a crew of VIII the burden of recall placed on coaches becomes substantial. An ideal solution would be to provide video during outings however, due to practical considerations, this is not always possible. Alternatively we may deploy BSNs on each oarsman to measure the efficiency of each stroke. This has several advantages. Firstly, the equipment required is extremely light representing minimal ballast. Secondly, the BSN can be used with or without the coach present, thus allowing crews to effectively debrief themselves when coaches cannot be found. Whilst studies have thus far been limited to a laboratory setting [117, 77, 76], we expect advances in BSNs to enable real-time

on-water sensing, providing immediate feedback to oarsmen and their coxswain – maximizing performance and minimizing the risk of injury due to poor technique [55].

3.2 Wellbeing

The aim of BSNs for wellbeing is to provide preventative care through target-based lifestyle management and the promotion of healthy pastimes; to maintain and improve *quality of life* through a tangible improvement in health. By improving general fitness, subjects can partake in medium exertion activities for longer, rendering them more active and less susceptible to disease. By leading an active lifestyle subjects also report reduced time to sleep and are therefore able to obtain a better nights rest [21]. To date, advances in wellbeing have focused mainly on providing motivation [75] for usually sedent subjects, either through feedback and positive reinforcement, or by improving the accessibility of new pass-times through the use of technology.

Shakra [12, 13, 85] has been developed to provide a metric of distance covered using mobile phone positioning. When walking or jogging users specify on their personal mobile phone that exercise has been started then as the carrier of that device moves around, the varying intensity of the GSM signal as the phone is handed off between cells is used to determine the overall activity of the owner. These metrics can be presented back to the user to bring about a change in behaviour when users do not meet self imposed targets. Interestingly, it has been noted in this study that one of the most important contributing factors in motivating a continued increase in activity is social in nature [12], and thus perhaps the most successful applications of BSNs for wellbeing will consider this basic human need. This may explain the popularity of the WiiFit balance board and fitness suite, for use with the Nintendo Wii, which uses weight distribution on the board in order to determine posture and movement. Typically, games involving this device have a strong multi-player focus, hinting towards social gaming as the contributing factor in its success. In this vein, Muller et al. present their work to facilitate social jogging [85], where participants may be geographically disparate. The advantage of this work is twofold – since joggers do not need to live in proximity or have similar running paces in order to communicate and encourage each other. As mentioned above, the Nike + iPod Sports Kit is commercially available to report on individual progress however, a more a advanced system that automatically selects particular music tracks based upon their motivating features has been proposed [90].

BSNs have been able to improve the accessibility of activities for wellbeing through tentative advances in automating expert knowledge. For those who cannot afford, or are too busy to commit to regular coaching sessions and fitness classes this provides a low-cost alternative and allows users to maximise the benefit of their workouts without being constrained to a particular schedule. To date, many commercial products exist within this category, such as the Polar and Garmin families for running and cycling. Within the literature low impact, core stability exercises

such as Yoga [33] and Tai-Chi [59] have also been addressed which can have beneficial impact in terms of overall improvement in flexibility and mobility, leading to a decreased probability of falls within the elderly [130].

3.3 Healthcare

BSNs were originally envisioned to provide supportive technologies for the elderly and infirm as well as for sufferers of both acute and chronic conditions. Within home monitoring environments this can be achieved through the study of behaviour profiles [5, 11], where deviations from what is considered "normal" can be detected. Such information can offer a welcome window into the life of an elder, from both the perspective of the subject and informal carers, such as immediate family and loved ones – and can be provided through two tier systems similar to that illustrated in Fig. 1. Recently, the fusion of ambient and wearable sensors has provided increased accuracy activity recognition for home healthcare environments [93], and is more robust to failure than using wearable or ambient sensors alone. BSNs may not only find application for general monitoring and have been used in continuous studies of gait for suffers of Parkinson's disease [109, 84]. Such technology allows for better disease management and provides a quantitative measure of the efficacy of prescribed medication.

Where surgery may have affected patient motion, as in the case of hip and knee surgery after injury, or after a stroke [134, 52], sensors can be used to evaluate recovery by charting the difference between healthy and current gait. Data can then be presented to rehabilitation specialists and to the users themselves to effect positive changes in behaviour that facilitate recovery. When combined with traditional physiotherapy, this ensures that incorrect or damaging habits are not adopted by the patient between visits.

One of the key applications of BSNs for elderly care management is in the detection of falls. To date, current research has provided mixed results, since it can be difficult to distinguish between a relatively rapid motion, such as sitting down, and an episodic fall. When this is coupled with the requirement for a high sensitivity in fall detection, and the variety of falls which can occur within the home, the complexity of the problem becomes apparent. Clearly, prevention is better than detection, and we have seen BSNs deployed to determine both the factors that cause falls [91] and develop exercises that can provide resilience to falls within the home [38]. For fall detection, many systems have been investigated with many different sensing modalities. In [4], vibration sensors have been used with the reported capacity to detect the vibration signature generated by the fall of a human body within a 15ft radius. Other approaches have utilised ambient sensors [115, 20] such as the SINBAD (Smart Inactivity Monitor using Array-Based Detectors) system, which leverages a wall mounted thermal imaging system for inactivity recognition. Where subjects move around their home, an array of infra-red sensors track target size, location and velocity. For fall detection, target motion is analysed for large variation in vertical velocity before target inactivity is measured and compared to a map of

Fig. 8 An elderly COPD sufferer participating in trials with an e-AR sensor. The ear mounted sensor monitors the level of exercise performed and provides feedback, via computer, to both patient and carer. This provides an insight into daily routine and helps subjects set achievable goals, ensuring they remain active. This can significantly slow down the deterioration of health and improve quality of life

acceptable inactivity periods in each segment of the camera's field of view. This has the effect of reducing the occurrence of false positives triggered by everyday events, for example, when a person is sitting down or getting into bed.

The use of BSNs in the treatment of Chronic Obstructive Pulmonary Disorder (COPD) – a non-reversible chronic condition characterized by a shortness of breath – has also recently been proposed [6]. COPD is caused by an inflammation of the airways and, whilst chronic in nature, moderate exercise is a recommended treatment which can slow deterioration, leading to a better quality of life. Exercise can be detected by body worn sensors, such as the e-AR, which then provide feedback regarding exertion levels to users – creating a target based system for patients who find self-motivation difficult. This can be beneficial in many ways. Firstly, it forces sufferers to breathe more deeply, increasing their intake of oxygen. Secondly, it increases the efficiency with which the body can utilise oxygen, thirdly, it encourages motion which ensures that other regions of the body do not atrophy or become prone to poor circulation and finally, it promotes a sense of wellbeing and reduces the probability of co-morbidity with other conditions such as obesity and diabetes.

4 Conclusions

In this chapter, we have provided an overview of the current and emerging technologies for BSNs which are set to revolutionize the way in which people of all ages manage their health. We have seen advances in sensing modalities which shall afford the next generation of practitioners and users a much deeper insight into the workings of the human body. This will pave the way for evidence based medicine and truly preventative healthcare – where we may be able to identity key mechanisms of disease before they cause irreversible damage. With such potential, considerations for the designers of future BSNs are significant, and it has become clear that for clinical adoption the responsibility for creating secure BSNs is a key concern.

Whilst research in BSNs has achieved a great deal in a short period of time, many challenges still lie ahead. As computational devices continue to shrink, we must ensure that our community strives to push the boundaries of biosensing, biocompatibility and intelligent data processing to keep pace – only in this way will we achieve our vision of transparent and continuous monitoring for all. Furthermore, the requirement for autonomous, *self-managing* BSNs is likely to become more important as the number of devices ubiquitously included within the environment continues to grow. This is an exciting time for BSNs, and we believe that advances in the field will highlight the need for future generations to take care of their most precious possession – their body.

References

1. Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing, pp. 304–307 (1999)
2. Ali, R., ElHelw, M., Atallah, L., Lo, B., Yang, G.Z.: Pattern mining for routine behaviour discovery in pervasive healthcare environments. In: Proceedings of the 5th International Conference on Information Technology and Application in Biomedicine (ITAB) (2008)
3. Allender, S., Peto, V., Scarborough, P., Kaur, A., Rayner, M.: Coronary Heart Disease Statistics. BHF: London (2008)
4. Alwan, M., Rajendran, P.J., Kell, S., Mack, D., Dalal, S., Wolfe, M., Felder, R.: A smart and passive floor-vibration based fall detector for elderly. In: Proceedings of the 2nd International Conference on Information and Communication Technologies, pp. 1003–1007 (2006)
5. Atallah, L., ElHelw, M., Pansiot, J., Stoyanov, D., Wang, L., Lo, B., Yang, G.Z.: Behaviour profiling with ambient and wearable sensing. In: IFMBE Proceedings of the 4th International Workshop on Wearable and Implantable Body Sensor Networks, pp. 133–138 (2007)
6. Atallah, L., Elsaify, A., Lo, B., Hopkinson, N., Yang, G.Z.: Gaussian process prediction for cross channel consensus in body sensor networks. In: the IEEE Proceedings of the 5th International Workshop on Wearable and Implantable Body Sensor Networks (2008)
7. Atallah, L., Lo, B., Yang, G.Z., Siegesmund, F.: Wirelessly accessible sensor populations (WASP) for elderly care monitoring. In: the workshop on Ambient Technologies for Diagnosing and Monitoring Chronic Patients, Part of Pervasive Health 2008 (2008)
8. Aziz, O., Atallah, L., Lo, B., ElHelw, M.,Wang, L., Yang, G.Z., Darzi, A.: A pervasive body sensor network for measuring post-operative recovery at home. Surgical Innovation 14(2), 83–90 (2007)
9. Aziz, O., Lo, B., Yang, G.Z., King, R., Darzi, A.: Pervasive body sensor network: An approach to monitoring the post-operative surgical patient. In: IEEE Proceedings of the 3rd International Workshop on Wearable and Implantable Body Sensor Networks, pp. 13–18 (2006)
10. Bandyopadhyay, S., Gianella, C., Maulik, U., Kargupta, H., Liu, K., Datta, S.: Clustering distributed data streams in peer-to-peer environments. Information Science 176(14) (2004)
11. Barger, T.S., Brown, D.E., Alwan, M.: Health-status monitoring through analysis of behavioral patterns. IEEE Transactions on Systems, Man, and Cybernetics 35, 22–27 (2005)
12. Barkhuus, L.: Designing ubiquitous computing technologies to motivate fitness and health. In: Grace Hopper Celebration of Women in Computing '06 (2006)
13. Barkhuus, L., Maitland, J., Anderson, I., Sherwood, S., Hall, M., Chalmers, M.: Shakra: Sharing and motivating awareness of everyday activity. In: Proceedings of Ubicomp 2006 (2006)
14. Benaroya, L., Bimbot, F., Gribonval, R.: Audio source separation with a single sensor. IEEE Transactions on Audio, Speech and Language Processing 1(14), 191–199 (2006)

15. Biagioni, E.S., Sasaki, G.: Wireless sensor placement for reliable and efficient data collection. In: Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS) (2003)
16. Blunt, J.: Respiration (Biophysical Measurement Series), Chap. Impedance Pneumography, pp. 107–126. Spacelabs Inc: Redmond, WA (1992)
17. Brady, S., Dunne, L.E., Tynan, R., Diamond, D., Smyth, B., O'Hare, G.M.P.: Garment-based monitoring of respiration rate using a foam pressure sensor. In: Proceedings of the Ninth IEEE International Symposium on Wearable Computers, pp. 214–215 (2005)
18. Brock, A., Griffiths, C.: Twentieth century mortality trends in England and Wales: Changes in mortality from all causes of death combined and the changes by broad chapter of the International Classification of Diseases (ICD). Health Statistics Quarterly 18 (2003)
19. Brodie, M., Walmsley, A., Page, W.: Fusion motion capture: a prototype system using inertial measurement units and GPS for the biomechanical analysis of ski racing. Sports Technology 1(1), 17–28 (2008)
20. Bromiley, P.A., Courtney, P.P., Thacker, N.A.: Design of a visual system for detecting natural events by the use of an independent visual estimate: a human fall detector. Series on Machine Perception and Artificial Intelligence 50 (2002)
21. Browman, C.P.: Sleep following sustained exercise. Psychophysiology 17(6), 577–580 (1980)
22. Brown, P.J., Bovey, J.D., Chen, X.: Context-aware applications: from the laboratory to the marketplace. IEEE Personal Communications 4(5), 58–64 (1997)
23. Chi, E.: Introducing wearable force sensors in martial arts. IEEE Pervasive 4, 47–53 (2005)
24. Cohen, L.T., Saunders, E., Clark, G.M.: Psychophysics of a prototype peri-modiolar cochlear implant electrode array. Hearing Research 155(1–2), 63–81 (2001)
25. Cosmed: K4b2 V02 sensor. http://www.cosmed.it/
26. Coyle, S., Wu, Y., Lau, K., Kim, J., Brady, S., Wallace, G., Diamond, D.: Design of a wearable sensing platform for sweat analysis. Proceedings of pHealth (2007)
27. Cozean, C., Pease, W., Hubbell, S.: Biofeedback and functional electric stimulation in stroke rehabilitation. Archives of Physical Medicine and Rehabilitation 69(6), 401–405 (1988)
28. Davis, J.L., Clarke, M.S., Bush, J.A., Jackson, T.: The use of NIRS to measure patterns of blood flow in lower limb during rest and incremental cycle exercise. Medicine and Science in Sports and Exercise 38(5) (2006)
29. DeLorey, D.S., Kowalchuk, J.M., Paterson, D.H.: The relationship between pulmonary O2 uptake kinetics and muscle deoxygenation during moderate-intensity exercise. Journal of Applied Physiology 95(1), 113–120 (2003)
30. Delouille, V., Neelamani, R., Baraniuk, R.: Robust distributed estimation in sensor networks using the embedded polygons algorithm. In: Proceedings of IPSN, pp. 405–413 (2004). DOI 10.1109/IPSN.2004.1307362
31. Dey, A.: Context-aware computing: The cyberdesk project. In: Proceedings of the AAAI Spring Symposium on Intelligent Environments, pp. 51–54 (1998)
32. Dixon, B., Ibey, B.L., Ericson, M.N., Wilson, M.A., Cote, G.L.: Monte Carlo modeling for perfusion monitoring. Optical Diagnostics and Sensing in Biomedicine III 4965(1), 42–48 (2003). DOI 10.1117/12.479264. URL http://link.aip.org/link/?PSI/4965/42/1
33. Fels, S., Gauthier, J., Smith, P.: Responses in light, sound and scent: a therapeutic interactive yoga system. In: Proceedings of IWEC, pp. 355–362 (2002)
34. Fraser, A.M.: Hidden Markov Models and Dynamical Systems. Society for Industrial and Applied Mathematics: Philadelphia, PA (2009)
35. Friedrich, P., Scholz, A., Clauss, J., Gruber, H., Wolf, B.: The mobile doctor: Ambient medicine through sensor aided personalized telemedical devices. In: Proceedings of MEDETEL (2007)
36. Funiak, S., Guestrin, C., Paskin, M., Sukthankar, R.: Distributed inference in dynamical systems. Advances in Neural Information Processing Systems 19 (2006)
37. Gorina, Y., Hoyert, D., Lentzner, H., Goulding, M.: Trends in causes of death among older persons in the United States. Aging Trends (6), 1–12 (2005)

38. Grabiner, M., Donovan, S., Bareither, M., Marone, J., Hamstra-Wright, K., Gatts, S., Troy, K.: Efficacy of task specific training to reduce fall risk: Trunk kinematics and fall risk of older adults: translating biomechanical results to the clinic. Journal of Electromyography and Kinesiology 18(2), 197–204 (2008)
39. Gray, C.: The Cyborg Handbook, 1st edn. Routledge, New York (2005)
40. Gu, D.: Distributed EM algorithm for Gaussian mixtures in sensor networks. IEEE Transactions on Neural Networks 19(7) (2008)
41. Guenette, J.A., Vogiatzis, I., Zakynthinos, S., Athanasopoulos, D., Kosklou, M., Golemati, S., Vasilopoulou, M., Wagner, H.E., Roussos, C., Wagner, P.D., Boushel, R.: Human respiratory muscle blood flow measured by near-infrared spectroscopy and indocyanine green. Journal of Applied Physiology 104(4), 1202–1210 (2008)
42. Guger, C., Harkam, W., Hertnaes, C., Pfurtscheller, G.: Prosthetic control by an EEG-based brain-computer interface (BCI). In: Proceedings of the 5th European Conference for the Advancement of Assistive Technology (AAATE) (1999)
43. Guo, W.P., Ma, X.M., Zeng, Y.: Clinical laboratories on a chip for human immunodeficiency virus assay. In: Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference (2005)
44. Harsanyi, G.: Sensors in Biomedical Applications, Fundamentals, Technology and Applications. CRC Press, Boca Raton, FL (2000)
45. Hayashi, T., Kawamoto, H., Sankai, Y.: Control method of robot suit HAL working as operator's muscle using biological and dynamical information. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2005)
46. Heller, A.: Miniature biofuel cells. Physical Chemistry and Chemical Physics 6, 209 – 216 (2004). DOI 10.1039/b313149a
47. Humayun, M.S., Weilanda, J.D., Fujiia, G.Y., Greenbergb, R., Williamsonb, R., Littleb, J., Mechb, B., Cimmarustib, V., Boemela, G.V., Dagneliec, G., de Juan Jr., E.: Visual perception in a blind subject with a chronic microelectronic retinal prosthesis. Vision Research 43(24), 2573–2581 (2003)
48. Humen, D.P., Kostuk, W.J., Klein, G.J.: Activity-sensing, rate-responsive pacing: Improvement in myocardial performance with exercise. Pacing and Clinical Electrophysiology 8(1), 52–59 (2006)
49. IBM: Autonomic Computing Manifesto. http://www.research.ibm.com/autonomic
50. Johannessen, E., Wyse, C., Cumming, D., Cooper, J.: Biocompatibility of a lab-on-a-pill sensor in artificial gastrointestinal environments. IEEE Transactions on Biomedical Engineering 53(11), 2333–2340 (2006)
51. Johansson, A., Nilsson, L., Kalman, S., Oberg, P.A.: Respiratory monitoring using photoplesmography: Evaluation in the postoperative care unit. In: Proceedings of the 20th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (1998)
52. Jovanov, E., Milenkovic, A., Otto, C., de Groen, P.C.: A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation. Journal of NeuroEngineering and Rehabilitation 2(1), 6 (2005)
53. King, R.C., Atallah, L., Darzi, A., Yang, G.Z.: A HMM framework for optimal sensor selection with applications to BSN sensor glove design. In: EmNets '07: Proceedings of the 4th Workshop on Embedded Networked Sensors, pp. 58–62. ACM, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1278972.1278987
54. King, R., Atallah, L., Lo, B., Yang, G.Z.: Development of a wireless sensor glove for surgical skills assessment. IEEE Transactions of Information Technology in Biomedicine 13(5), 673–9 (2009)
55. King, R.C., McIlwraith, D.G., Lo, B., Pansiot, J.: Body sensor networks for monitoring rowing technique. In: Proceedings of the 6th International Workshop on Wearable and Implantable Body Sensor Networks (2009)
56. Korel, B.T., Koo, S.G.: Addressing context awareness techniques in body sensor networks. In: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW '07) (2007)

57. Krahn, A.D., Klein, G.J., Skanes, A.C., Yee, R.: Insertable loop recorder use for detection of intermittent arrhythmias. Pacing and Clinical Electrophysiology 27(5), 657–664 (2004)
58. Kuiken, T., Amir, H., Scheidt, R.: Computerized biofeedback knee goniometer: acceptance and effect on exercise behavior in post-total knee arthroplasty rehabilitation. Archives of Physical Medicine and Rehabilitation 85(6), 1026 – 1030 (2004)
59. Kunze, K., Barry, M., Heinz, E.A., Lukowicz, P., Majoe, D., Gutknecht, J.: Towards recognizing T'ai Chi an initial experiment using wearable sensors. In: Proceedings IFAWC '05 (2005)
60. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289. Morgan Kaufmann Publishers Inc.: San Francisco, CA (2001). URL http://www.cis.upenn.edu/ pereira/papers/crf.pdf
61. Lam, W., Segre, A.: A distributed learning algorithm for Bayesian inference networks. IEEE Transactions of Knowledge and Data Engineering 14(1) (2002)
62. Lee, D.S.: Effective Gaussian mixture learning for video background subtraction. IEEE Transactions on Pattern Analysis 27(5), 827–832 (2005)
63. Legrand, R., Marles, A., Prieur, F., Lazzari, S., Blondel, N., Mucci, P.: Related trends in locomotor and respiratory muscle oxygenation during exercise. Medicine and Science in Sports and Exercise 39(1), 91–100 (2007)
64. Liu, H., Motoda, H.: Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic Publishers, Dordrecht (1998)
65. Lo, B., Atallah, L., Aziz, O., ElHelw, M., Darzi, A., Yang, G.Z.: Real-time pervasive monitoring for post-operative care. In: IFMBE Proceedings of the 4th International Workshop on Wearable and Implantable Body Sensor Networks, pp. 122–127 (2007)
66. Lo, B., Thiemjarus, S., King, R., Yang, G.Z.: Body sensor network a wireless sensor platform for pervasive healthcare monitoring. In: Adjunct Proceedings of the 3rd International Conference on Pervasive Computing (PERVASIVE 2005), pp. 77–80 (2005)
67. Lo, B., Wang, L., Yang, G.: From imaging networks to behavior profiling: Ubiquitous sensing for managed homecare of the elderly. In: Proceedings of the 3rd International Conference on Pervasive Computing, pp. 101–104 (2005)
68. Lo, B., Yang, G.Z.: Body sensor networks – research challenges and opportunities. In: IET Seminar on Antennas and Propagation for Body-Centric Wireless Communications (2007)
69. Loeb, G., Richmond, F., Singh, J., Peck, R., Tan, W., Zou, Q., Sachs, N.: RF-powered BIONs for stimulation and sensing. In: Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 4182–4185 (2004)
70. Lyons, G., Sinkjaer, T., Burridge, J., Wilcox, D.: A review of portable FES-based neural orthoses for the correction of drop foot. IEEE Transactions on Neural Systems and Rehabilitation Engineering 10(4), 260–279 (2002)
71. Mann, S.: Smart clothing: the shift to wearable computing. Communications of the ACM 39(8), 23–24 (1996). DOI http://doi.acm.org/10.1145/232014.232021
72. Margalit, E, Maia, M, Weiland, J., Greenberg, R., Fujii, G., Torres, G., Piyathaisere, D., O'Hearn, T., Liu, W., Lazzi, G., Dagnelie, G., Scribner, D., de Juan Jr, E., Humayun, M.: Retinal prosthesis for the blind. Survey of Ophthalmology 47(4), 335–356 (2002)
73. Matzel, K., Stadelmaier, U., Hohenfellner, N., Gall, F.: Electrical stimulation of sacral spinal nerves for treatment of faecal incontinence. Lancet 346, 1124–1127 (1995)
74. McCaffrey, C., Chevalerias, O., O'Mathuna, C., Twomey, K.: Swallowable-capsule technology. IEEE Pervasive Computing 7(1), 23–29 (2008)
75. McElroy, M.: Resistance to Exercise: A Social Analysis of Inactivity. Human Kinetics Publishers: Champaign, IL (2002)
76. McGregor, A., Anderton, L., Gedroyc, W.: The trunk muscles of elite oarsmen. British Journal of Sports Medicine 36(3), 214–217 (2002)
77. McGregor, A.H., Patankar, Z.S., Bull, A.M.: Longitudinal changes in the spinal kinematics of oarswomen during step testing. Journal of Sports Science and Medicine 6, 29–35 (2007)

78. McIlwraith, D.G., Pansiot, J., Thiemjarus, S., Lo, B., Yang, G.Z.: Probabilistic decision level fusion for real-time correlation of ambient and wearable sensors. In: EEE Proceedings of the 5th International Workshop on Wearable and Implantable Body Sensor Networks (2008)
79. Michahelles, F., Schiele, B.: Sensing and monitoring professional skiers. IEEE Pervasive 4, 40–45 (2005)
80. Miller, W., Yafuso, M., Yan, C., Hui, H., Arick, S.: Performance of an in-vivo, continuous blood-gas monitor with disposable probe. Clinical Chemistry 33, 1538–1542 (1987)
81. Minnen, D., Westeyn, T., Ashborrk, D., Presti, P., Starner, T.: Recognising solider activities in the field. In: Proceedings of the 4th International Workshop on Wearable and Implantable Body Sensor Networks, pp. 236–241 (2007)
82. Mizrahi, E.M., Kellaway, P.: Characterization and classification of neonatal seizures. Neurology 37(12), 1837– (1987). URL http://www.neurology.org/cgi/content/abstract/37/12/1837
83. Montgomery, K., Mundt, C., Thonier, G., Tellier, A., Udoh, U., Barker, V., Ricks, R., Giovangrandi, L., Davies, P., Cagle, Y., Swain, J., Hines, J., Kovacs, G.: Lifeguard – a personal physiological monitor for extreme environments. In: Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEMBS), pp. 2192–2195 (2004). DOI 10.1109/IEMBS.2004.1403640
84. Moore, S.T., et al.: Long-term monitoring of gait in Parkinson's disease. Gait and Posture 26(2), 200–207 (2007)
85. Mueller, F.F., O'Brien, S., Thorogood, A.: Jogging over a distance: supporting a "jogging together" experience although being apart. In: Proceedings of CHI '07, pp. 2579–2584 (2007)
86. Nakamura, Y., Tei, K., Fukazawa, Y., Honiden, S.: Region-based sensor selection for wireless sensor networks. In: SUTC '08: Proceedings of the 2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2008), pp. 326–331. IEEE Computer Society, Washington, DC, USA (2008). DOI http://dx.doi.org/10.1109/SUTC.2008.47
87. Nowak, R.D.: Distributed EM algorithms for density estimation and clustering in sensor networks. IEEE Transactions on Signal Processing 51(8) (2003)
88. Oeppen, J., Vaupel, J.: DEMOGRAPHY: Enhanced: Broken limits to life expectancy. Science 296(5570), 1029–1031 (2002)
89. Ohgi, Y.: Microcomputer-based acceleration sensor device for sports biomechanics -stroke evaluation by using swimmer's wrist acceleration. IEEE Sensors 1 (2002)
90. Oliver, N., Flores-Mangas, F.: MPTrain: a mobile, music and physiology-based personal trainer. In: Proceedings of the 8th Conference on Human–Computer Interaction with Mobile Devices and Services, pp. 21–28 (2006)
91. Owings, T.M., Pavol, M.J., Grabiner, M.D.: Mechanisms of failed recovery following postural perturbations on a motorized treadmill mimic those associated with an actual forward trip. Clinical Biomechanics 16(9), 813–819 (2001)
92. Pansiot, J., King, R., McIlwraith, D., Lo, B., Yang, G.Z.: ClimBSN: Climber performance monitoring with BSN. In: IEEE Proceedings of the 5th International Workshop on Wearable and Implantable Body Sensor Networks (2008)
93. Pansiot, J., Stoyanov, D., McIlwraith, D., Lo, B.P., Yang, G.Z.: Ambient and wearable sensor fusion for activity recognition in healthcare monitoring systems. In: Proceedings of Body Sensor Networks '07 (2007)
94. Paradiso, R.: Wearable health care system for vital signs monitoring. In: Proceedings of the 4th International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine (2003)
95. Park, C., Chou, P.H., Bai, Y., Matthews, R., Hibbs, A.: An ultra-wearable, wireless, low power ECG monitoring system. In: Proceedings of IEEE BioCAS (2006)
96. Park, J.H., Friedman, G., Jones, M.: Geographical feature sensitive sensor placement. Journal of Parallel and Distributed Computing 64(7), 815–825 (2004)
97. Paskin, M., Guestrin, C.: Robust probabilistic inference in distributed systems. In: Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence (UAI) (2004)

98. Paskin, M., Guestrin, C., McFadden, J.: A robust architecture for distributed inference in sensor networks. In: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN) (2005)
99. Patterson, J.A.C., McIlwraith, D.G., Yang, G.Z.: A flexible, low noise reflective PPG sensor platform for ear-worn heart rate monitoring. In: Proceedings of the 6th International Workshop on Wearable and Implantable Body Sensor Networks (2009)
100. Pocock, G., Richards, C.: Human Physiology: The Basis of Medicine. Oxford University Press, Oxford (2006)
101. Powell, C., Fuchs, T., Finkelstein, D., Garan, H., Cannom, D., McGovern, B., Kelly, E., Vlahakes, G., Torchiana, D., Ruskin, J.: Influence of implantable cardioverter-defibrillators on the long-term prognosis of survivors of out-of-hospital cardiac arrest. Circulation 88, 1083–1092 (1993)
102. Predd, J., Kulkarn, S., Poor, H.: Distributed learning in wireless sensor networks. IEEE Signal Processing Magazine 56 (2006)
103. Rajasekaran, M.P., Radhakrishnan, S., Subbaraj, P.: Remote monitoring of post-operative patients using wireless sensor networks. International Journal of Healthcare Technology and Management 9(3), 247–257 (2008)
104. Renard, E.: Implantable closed-loop glucose-sensing and insulin delivery: The future for insulin pump therapy. Current Opinion in Pharmacology 2(6), 708–716 (2002). DOI 10.1016/S1471–4892(02)00216–3
105. Riley, J.: Rising Life Expectancy: A Global History. Cambridge University Press: New York (2001)
106. Ripley, B.: Pattern Recognition and Neural Networks. Cambridge University Press: Cambridge (1996)
107. Ruttkay, Z., Zwiers, J., van Welbergen, H., Reidsma, D.: Towards a reactive virtual trainer. In: Proceedings of the 6th International Conference Intelligent Virtual Agents, pp. 292–303 (2006)
108. Ryan, N.S., Pascoe, J., Morse, D.R.: Enhanced Reality Fieldwork: The Context-aware Archaeological Assistant. British Archaeological Reports. Tempus Reparatum: Oxford (1998). URL http://www.cs.kent.ac.uk/pubs/1998/616
109. Salarian, A., Russmann, H., Vingerhoets, F., Dehollain, C., Blanc, Y., Burkhard, P., Aminian, K.: Gait assessment in Parkinson's disease: Toward an ambulatory system for long-term monitoring. IEEE Transactions on Biomedical Engineering 5(8), 1434–1443 (2004)
110. Sandersa, C.A., Rodriguez, M., Greenbaum, E.: Stand-off tissue-based biosensors for the detection of chemical warfare agents using photosynthetic fluorescence induction. Biosensors and Bioelectronics 16(7), 439–446 (2001)
111. Sarakoglou, I., Tsagarakis, N., Caldwell, D.: Occupational and physical therapy using a hand exoskeleton based exerciser. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2004)
112. Schilit, B., Theimer, M.: Disseminating active map information to mobile hosts. IEEE Network 8(5), 22–32 (1994)
113. Schlenoff, C., Weiss, B., Steves, M., Virts, A., Shneier, M., Linegang, M.: Overview of the first advanced technology evaluations for ASSIST. In: Proceedings of the 2006 Performance Metrics for Intelligent Systems Workshop (2006)
114. Scilingo, E.P., Lanat, A., Zito, D., Pepe, D., Mincica, M., Zito, F., Rossi, D.D.: Wearable monitoring of cardiopulmonary activity through radiant sensing. In: Proceedings of the 5th International Workshop on Wearable Micro and Nanosystems for Personalised Health (2008)
115. Sixsmith, A., Johnson, N.: A smart sensor to detect the falls of the elderly. IEEE Pervasive Computing 3(2), 42–47 (2004)
116. Slawson, S.E., Justham, L.M., West, A.A., Conway, P.P., Caine, M.P., Harrison, R.: The Engineering of Sport 7, Chap. Accelerometer Profile Recognition of Swimming Strokes, pp. 81–87. Springer, Paris (2008)
117. Steer, R.R., McGregor, A.H., Bull, A.M.: A comparison of kinematics and performance measures of two rowing ergometers. Journal of Sports Science and Medicine 5, 52–59 (2005)

118. Szaflarski, N.: Emerging technology in critical care: continuous intra-arterial blood gas monitoring. American Journal of Critical Care 5(1), 55–65 (1996)
119. Thiemjarus, S., Lo, B.P.L., Yang, G.Z.: A spatio-temporal architecture for context aware sensing. In: Proceedings of Body Sensor Networks '06, pp. 191–194 (2006)
120. Vail, D.L., Veloso, M.M., Lafferty, J.D.: Conditional random fields for activity recognition. In: Proceedings of AAMAS '07 (2007)
121. Vallespín, B., Alonso, A., de Arana, M., Ehrenberg, S., Pastor, X., Roca, J.: Continuous mobile services for healthcare (HealthService24 project). Technology and Health Care 13(5), 441–443 (2005)
122. Villringer, A., Planck, J., Hock, C., Schleinkofer, L., Dirnagl, U.: Near infrared spectroscopy (NIRS): A new tool to study hemodynamic changes during activation of brain function in human adults. Neuroscience Letters 154(1–2), 101–104 (1993)
123. Wang, L., Lo, B., Yang, G.: Reflective photoplethysmograph earpiece sensor for ubiquitous heart rate monitoring. In: Body Sensor Networks (BSN07), pp. 179–183 (2007). DOI 10.1007/978–3-540–70994–731(31)
124. Wang, L., Lo, B., Yang, G.Z.: Multichannel reflective PPG earpiece sensor with passive motion cancellation. IEEE Transactions on Biomedical Circuits and Systems 1(4), 235–241 (2007)
125. Wang, L., Thiemjarus, S., Lo, B., Yang, G.Z.: Toward a mixed-signal reconfigurable ASIC for real-time activity recognition. In: IEEE Proceedings of the 5th International Workshop on Wearable and Implantable Body Sensor Networks, pp. 227–230 (2008)
126. Want, R., Pering, T., Tennenhouse, D.: Comparing autonomic and proactive computing. IBM Systems Journal 42(1), 129–135 (2003)
127. Weiss, I.K., Fink, S., Harrison, R., Feldman, J.D., Brill, J.E.: Clinical use of continuous arterial blood gas monitoring in the pediatric intensive care unit. Pediatrics 103(2), 440–445 (1999)
128. Wilhelm, F.H.: The lifeshirt: An advanced system for ambulatory measurement of respiratory and cardiac function. Behavior Modification 27(5), 671–691 (2003). DOI 10.1177/0145445503256321
129. Winawer, S., Zauber, A., Ho, M., O'Brien, M., Gottlieb, L., Sternberg, S., Waye, J., Schapiro, M., Bond, J., et al., J.P.: Prevention of colorectal cancer by colonoscopic polypectomy. The national polyp study workgroup. New England Journal of Medicine 329(27), 2028–2029 (1993)
130. Wolf, S., Sattin, R., Kutner, M.: Intense T'ai Chi exercise training and fall occurrences in older, transitionally frail adults: a randomized, controlled trial. Journal of the American Geriatrics Society 1, 188–189 (2003)
131. Wolpaw, J., McFarland, D., Neat, G., Forneris, C.: An EEG-based brain-computer interface for cursor control. Electroencephalography and Clinical Neurophysiology 78(3), 252–259 (1991)
132. Yang, G.Z. (ed.): Body Sensor Networks. Springer-Verlag, New York (2006)
133. Zequera, M., Stephan, S., Paul, J.: The "PAROTEC" foot pressure measurement system and its calibration procedures. In: E. in Medicine, B.S. (EMBS) (eds.) 28th Annual International Conference of the IEEE, pp. 4135–4139 (2006). DOI 10.1109/IEMBS.2006.259624
134. Zheng, H.: Position-sensing technologies for movement analysis in stroke rehabilitation. Medical and Biological Engineering and Computing 43(4), 413–420 (2005)

Index

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6, © Springer-Verlag Berlin Heidelberg 2010

SIGNALS AND COMMUNICATION TECHNOLOGY

(continued from page ii)

EM Modeling of Antennas and RF Components for Wireless Communication Systems
F. Gustrau, D. Manteuffel
ISBN 3-540-28614-4

Interactive Video Methods and Applications
R. I Hammoud (Ed.)
ISBN 3-540-33214-6

ContinuousTime Signals
Y. Shmaliy
ISBN 1-4020-4817-3

Voice and Speech Quality Perception
Assessment and Evaluation
U. Jekosch
ISBN 3-540-24095-0

Advanced ManMachine Interaction
Fundamentals and Implementation
K.-F. Kraiss
ISBN 3-540-30618-8

Orthogonal Frequency Division Multiplexing for Wireless Communications
Y. (Geoffrey) Li and G.L. Stüber (Eds.)
ISBN 0-387-29095-8

Circuits and Systems Based on Delta Modulation
Linear, Nonlinear and Mixed Mode Processing
D.G. Zrilic
ISBN 3-540-23751-8

Functional Structures in Networks
AMLn—A Language for Model Driven Development of Telecom Systems
T. Muth
ISBN 3-540-22545-5

RadioWave Propagation for Telecommunication Applications
H. Sizun
ISBN 3-540-40758-8

Electronic Noise and Interfering Signals
Principles and Applications
G. Vasilescu
ISBN 3-540-40741-3

DVB
The Family of International Standards for Digital Video Broadcasting, 2nd ed.
U. Reimers
ISBN 3-540-43545-X

Digital Interactive TV and Metadata
Future Broadcast Multimedia
A. Lugmayr, S. Niiranen, and S. Kalli
ISBN 3-387-20843-7

Adaptive Antenna Arrays
Trends and Applications
S. Chandran (Ed.)
ISBN 3-540-20199-8

Digital Signal Processing with Field Programmable Gate Arrays
U. Meyer-Baese
ISBN 3-540-21119-5

Neuro-Fuzzy and Fuzzy Neural Applications in Telecommunications
P. Stavroulakis (Ed.)
ISBN 3-540-40759-6

SDMA for Multipath Wireless Channels
Limiting Characteristics and Stochastic Models
I.P. Kovalyov
ISBN 3-540-40225-X

Digital Television
A Practical Guide for Engineers
W. Fischer
ISBN 3-540-01155-2

Speech Enhancement
J. Benesty (Ed.)
ISBN 3-540-24039-X

Multimedia Communication Technology
Representation, Transmission and Identification of Multimedia Signals
J.R. Ohm
ISBN 3-540-01249-4

Information Measures
Information and its Description in Science and Engineering
C. Arndt
ISBN 3-540-40855-X

Processing of SAR Data
Fundamentals, Signal Processing, Interferometry
A. Hein
ISBN 3-540-05043-4

Chaos-Based Digital Communication Systems
Operating Principles, Analysis Methods, and Performance Evalutation
F.C.M. Lau and C.K. Tse
ISBN 3-540-00602-8

Adaptive Signal Processing
Application to Real-World Problems
J. Benesty and Y. Huang (Eds.)
ISBN 3-540-00051-8

Multimedia Information Retrieval and Management Technological
Fundamentals and Applications
D. Feng, W.C. Siu, and H.J. Zhang (Eds.)
ISBN 3-540-00244-8

Structured Cable Systems
A.B. Semenov, S.K. Strizhakov, and I.R. Suncheley
ISBN 3-540-43000-8

UMTS
The Physical Layer of the Universal Mobile Telecommunications System
A. Springer and R. Weigel
ISBN 3-540-42162-9

Advanced Theory of Signal Detection
Weak Signal Detection in Generalized Obeservations
I. Song, J. Bae, and S.Y. Kim
ISBN 3-540-43064-4

Wireless Internet Access over GSM and UMTS
M. Taferner and E. Bonek
ISBN 3-540-42551-9