COMPUTATIONAL BIOLOGY

Foundations of Systems Biology

Using Cell Illustrator and Pathway Databases

Masao Nagasaki
Ayumu Saito
Atsushi Doi
Hiroshi Matsuno
Satoru Miyano

Springer

CD-ROM
INCLUDED

Computational Biology

The *Computational Biology* series publishes the very latest, high-quality research devoted to specific issues in computer-assisted analysis of biological data. The main emphasis is on current scientific developments and innovative techniques in computational biology (bioinformatics), bringing to light methods from mathematics, statistics and computer science that directly address biological problems currently under investigation.

The series offers publications that present the state-of-the-art regarding the problems in question; show computational biology/bioinformatics methods at work; and finally discuss anticipated demands regarding developments in future methodology. Titles can range from focused monographs, to undergraduate and graduate textbooks, and professional text/reference works.

Author guidelines: springer.com > Authors > Author Guidelines

For other titles published in this series, go to http://www.springer.com/series/5769

Masao Nagasaki • Ayumu Saito • Atsushi Doi •
Hiroshi Matsuno • Satoru Miyano

Foundations of Systems Biology

Using Cell Illustrator® and Pathway Databases

Springer

Dr. Masao Nagasaki
Dr. Ayumu Saito
Prof. Satoru Miyano
University of Tokyo
Inst. Medical Science
Human Genome Center
4-6-1 Shirokanedai
Tokyo
Minato-ku
108-8639 Japan
miyano@ims.u-tokyo.ac.jp

Dr. Atsushi Doi
Institute of System LSI Design Industry
Fukuoka R & D Center
3-8-34 Momochihama
Fukuoka
Office 608, Sawara-ku
814-0001 Japan

Prof. Hiroshi Matsuno
Yamaguchi University
Graduate School of Science &
Engineering
Yamaguchi
753-8512 Japan

Computational Biology Series ISSN 1568-2684
ISBN: 978-1-84882-022-7 e-ISBN: 978-1-84882-023-4
DOI: 10.1007/978-1-84882-023-4

Translated by Satoru Miyano, Masao Nagasaki and Ayumu Saito

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

Library of Congress Control Number: 2009922124

Cover design: KünkelLopka GmbH, Heidelberg

Printed on acid-free paper

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

Foreword

Today, as hundreds of genomes have been sequenced and thousands of proteins and more than ten thousand metabolites have been identified, navigating safely through this wealth of information without getting completely lost has become crucial for research in, and teaching of, molecular biology.

Consequently, a considerable number of tools have been developed and put on the market in the last two decades that describe the multitude of potential/putative interactions between genes, proteins, metabolites, and other biologically relevant compounds in terms of metabolic, genetic, signaling, and other networks, their aim being to support all sorts of explorations through bio-data bases currently called *Systems Biology*.

As a result, navigating safely through this wealth of information-processing tools has become equally crucial for successful work in molecular biology.

To help perform such navigation tasks successfully, this book starts by providing an extremely useful overview of existing tools for finding (or designing) and investigating metabolic, genetic, signaling, and other network databases, addressing also user-relevant practical questions like

- Is the database viewable through a web browser?
- Is there a licensing fee?
- What is the data type (metabolic, gene regulatory, signaling, etc.)?
- Is the database developed/maintained by a curator or a computer?
- Is there any software for editing pathways?
- Is it possible to simulate the pathway?

It then goes on to introduce a specific such tool, that is, the fabulous “Cell Illustrator 3.0” tool developed by the authors. The book explains in great detail how this tool can be used for creating, analyzing, and simulating models explicating and testing our current understanding of basic biological processes. They pertain, for example, to

— the organization and control of metabolic networks and metabolic flux analysis,
— the regulation of gene transcription, processing, and translation, or

— the processing of information via signaling pathways.

The book deals with such topics by providing a fascinating array of detailed examples. Thus, it can serve as a perfect introduction to contemporary cell biology for anybody who wants to quickly gain insight into the most important and topical directions of research in this field. In particular, the book provides invaluable help for anybody who wants to learn more about why and how the current big bio-data bases can be used to develop and support Systems Biology research.

Therefore, any biology student can, and actually should, just work through these examples on his own screen to quickly gain important and solid expertise and become a valuable and well-informed member of the continuously growing Systems Biology research community.

The authors Masao Nagasaki, Ayumu Saito, Atsushi Doi, Hiroshi Matsuno, and Satoru Miyano have been working at the forefront of *in silico*-based biology for quite a few years, and are highly respected in the community.

I am therefore very happy to have their book appear in this series, and I congratulate the publishers for the very good work they have done in dealing with the challenging task of appropriately editing such a strongly digitally-oriented manuscript.

Prof. Dr. Andreas Dress
Director
Department of Combinatorics and Geometry (DCG)
CAS-MPG Partner Institute for Computational Biology (PICB)
Shanghai Institutes for Biological Sciences (SIBS)
Chinese Academy of Sciences (CAS)
June 2008

Preface

It has been said that “Systems Biology” is an important postgenomic challenge in biology to understand “life as systems”. That being said, what does it mean? What can be done with signaling pathways, metabolic pathways, and gene regulatory networks using computers? For those with similar concerns or questions, this should be the first book you consult for an understanding of *Systems Biology*.

The definition of *Systems Biology* varies from scientist to scientist. Some of you may have skimmed books or scientific papers with “Systems Biology” in the title and seen alien terms such as “robustness analysis”, “stochastic differential equations”, or “bifurcation analysis” fly by. Some may have felt that this is similar to lining up toy soldiers called differential equations and making them march. Those of you who have felt that way are the intended audience of this book.

Biological organisms consist of many molecules, such as proteins, which fulfill their functions and interact with others. One of the ways to understand this system is to construct the system in parts on a computer and analyze. Beneath the current attentions to Systems Biology is the compilation of large amounts of genomic data and biological knowledge on the parts that compose everything from bacteria to human beings. Since the basic mechanisms of these parts have been considerably well defined, it is now time to understand how the interactions between these parts create the high degree of complexity in biological systems.

On one hand, man-made systems such as electrical circuits and machinery can be made over and over once there are parts and blueprints, since the system is known from the beginning. On the other hand, organisms are made by nature and evolution, and there is a large gap between gathering the parts and understanding the system. Modeling and simulation are necessary technologies to close this gap. In order to understand this system, it needs to be modeled with a high-level language including mathematics and entered into a computer for computation. We should say a goodbye to messy (in Japanese, we say “Gochagocha”) printed diagrams with arrows and circles of various shapes with narrations. This is the point of entry of “Cell Illustrator”, which is a software tool for biological pathway modeling and simulation.

Reading the book and using Cell Illustrator bundled in the CD-ROM should make it possible to create highly complex pathways and simulations. There is no need for prior knowledge in differential equations or programming. The prerequisites are interest in biology, ability to operate a cell phone (or equivalent), and mathematical ability of a standard middle school student or better.

Using Cell Illustrator, reading the book, and finishing the exercises—answers are provided—should make you realize how easy this can be “ (^ o ^) v”. Although pathway drawing does not require any mathematical or programming skills, drawing pathways may require some artistic sense. In addition, just by drawing pathways using Cell Illustrator, pathway knowledge will become better organized, and the reader should feel a sense of accomplishment. The columns interspersed in the book are addendums and digressions; they can be skimmed at the reader’s discretion.

This book is designed and structured to be used for a semester-long course text at the undergraduate level or can be used as a part of graduate courses. Chapter 1 describes a minimum biological knowledge and Chapters 2 and 3 explain some of the important pathway databases and software tools together with their related concepts. Chapter 4 describes the detailed first steps and elements for modeling pathways with Cell Illustrator. The reader may find that graphical pictures representing biological entities and processes help understanding the elements of pathways. Chapter 5 will guide the reader to model three kinds of pathways in a step-by-step manner as exercises. Chapter 6 discusses the computational functionalities required for Systems Biology. This book is an English translation of the original Japanese version published by Kyoritsu Shuppan Co., Ltd. With this edition, the data on software and database versions are updated and Chapter 6 is enhanced with some new topics.

We are grateful to many people. First and foremost, we would like to thank the current and former members of the Cell System Markup Language Project: Emi Ikeda, Euna Jeong, Kaname Kojima, Chen Li, Hiroko Nishihata, Kazuyuki Numata, Yayoi Sekiya, Yoshinori Tamada, Kazuko Ueno of Human Genme Center; Kanji Hioka, Yuto Ikegami, Hironori Kitakaze, Yoshimasa Miwa, Daichi Saihara, Tomoaki Yamamotoya of Yamaguchi University.

Andreas Dress should be specially acknowledged for the foreword of this book. For this English version, we were encouraged by Holger Karas and Edgar Wingender of BIOBASE and Wayne Wheeler of Springer U.K. as well as Koichi Nobusawa and Yumiko Kita of Kyoritsu Shuppan Co., Ltd. for the original Japanese version. Special thanks go to Jocelyne Bruand of UCSC and Tatsunori Hashimoto of Harvard University for helping this translation, and to Seiya Imoto, Rui Yamaguchi, Teppei Shimamura, André Fujita, Yosuke Hatanaka, Eric Perrier, Jin Hwan Do, and Takashi Yamamoto for their tremendous supports for Cell Illustrator.

Tokyo,
June 2008

Masao Nagasaki
Ayumu Saito
Atsushi Doi
Hiroshi Matsuno
Satoru Miyano

Contents

Chapter 1
Introduction

The primary aim of Systems Biology is “systems understanding of biology”. What does this phrase mean? What can be done with “signaling pathway”, “gene regulatory network”, and “metabolic pathway” using computers? This book is meant to be the first book for those people who have such questions and interests. Understanding the contents requires neither prior background knowledge/experiences in differential equations nor computer programming. Reading this book by using Cell Illustrator should enable the reader to make complex biological pathways for simulation. In this chapter we explain the basics which constitute these biological pathways.

1.1 Intracellular Events

A multitude of events occur within a cell. Inside, various molecules are fulfilling their functions, creating energy and proteins necessary for the cell’s survival and reproduction. On the surface of a cell, various molecules are receiving stimuli from the outside. This resembles a human society, with its diversity of specialists. There are proteins that transduce signals, and proteins that receive them. Some fulfill as critical a role as creating energy for the cell, while others help metabolize other molecules.

1.1.1 Transcription, Translation, and Regulation

The cell’s function, consisting of a variety of protein interactions, begins with the production of protein from DNA information. First, genetic information, which is coded as DNA in the nucleus, undergoes the process called *transcription* and produces mRNA. Ribosomes translate mRNA to protein. This process is called *translation*. The produced proteins have various functions. Some proteins move into the nucleus after synthesis and regulate the expression of certain genes by binding to specific sites of the DNA. This regulation is activation or repression. In the former case, the gene is up-regulated and so is expressed more; in the latter case, the gene is down-regulated and may not be expressed at all. Thus, not all genes are necessarily expressed at any given time. Even in the same person, depending on the cell type, there exist cells with different patterns of gene expression. In addition, miRNA, a type of RNA, has been recently discovered to influence expression regulation.

COLUMN 1

Small RNA

It is commonly known that “proteins form the bulk of cell function”. As mentioned above, according to the central dogma of molecular biology, proteins are produced by the sequence of transcription from DNA to mRNA and translation from mRNA to protein. However, some of the transcribed RNA have unknown function, unlike mRNA. This type of RNA was long thought to be garbage, and kept outside the scope of investigation.

However, in 1993, one such RNA sequence was found to control the expression of certain genes. Similar phenomena were discovered in the 21st century in other organisms, and these sequences became known as *microRNA* (miRNA). The miRNA sequences are very short, with only 20-25 base pairs length. They are thought to combine with protein and bind to a partially complementary mRNA, and prevent its translation, rather than moving to the cytoplasm like mRNA. In other words, the recently discovered miRNA is a type of molecule with the ability to block protein translation. In plants, an analogous type of RNA, *short interfering RNA* (siRNA), has been found to block viral RNA transcription. The roles of small RNA segments are being investigated. In fact, it is often said that the first functional molecules on the Earth resembled nucleic acids like RNA. Because nucleic acids carry information, it could be said that they are the basis of life. As sustaining any system is costly biologically, a sufficiently evolved organism has no reason to sustain any systems useless to survival.

In conclusion, the biological networks are complex, and one must not forget that there exist functional molecules other than proteins.

1.1.2 Signaling Pathways and Proteins

On the other hand, some proteins are secreted outside cell walls after being produced, and transmit messages to other cells. These proteins, called *ligands*, transmit messages, while others, called *receptors*, receive them. The three-dimensional structures of a ligand and receptor are complementary, resembling a molecular key and lock; therefore, a ligand only binds to the receptor that matches its shape. Upon receiving the ligand, the receptor is activated, and transduces the signal to another protein. This protein in turn activates another protein. The network of molecules transducing the signals is called a *signaling pathway* or *signal transduction pathway*. These signals reach the nucleus and lead to the aforementioned gene regulation.

1.1.3 Metabolism and Genes

The cell metabolizes the required compounds like ATP, amino acids, and sugars necessary through a variety of chemical reactions. For example, ethanol is metabolized to acetaldehyde which in turn becomes acetic acid. In addition to the proper reagents, these metabolic reactions require enzymes, which are produced from genes.

1.2 Intracellular Reactions and Pathways

A metabolic pathway is a network comprising many reactions. This is also the case for a signal transduction pathway and gene regulatory network. We generally call this network a *pathway*. Usually these pathways are visually represented as a network diagram of genes and their products in textbooks and pathway databases.

Figure 1.1 is an example showing gene regulatory relationships. The gene *Mdm2* inhibits the gene *p53*, which activates the gene *Bax*. The arrows that connect genes show the various relations between genes.

$$Mdm2 \;\dashv\; p53 \;\longrightarrow\; Bax$$

Fig. 1.1

Figure 1.2 is an example of a signaling pathway. The ligand FasL carries the apoptosis signal. The receptor Fas binds with FasL and transduces the signal by activating Caspase 8. In a signaling pathway diagram, the arrow represents chemical interaction such as the binding of protein to protein and phosphorylation.

The pathway for converting ethanol to acetic acid is usually represented as shown in Figure 1.3. The arrows connect the metabolic products in order. Each arrow repre-

$$\text{FasL (Ligand)} \longrightarrow \text{Fas (Receptor)} \longrightarrow \text{Caspase8 (Enzyme)}$$

Fig. 1.2

sents a certain metabolic reaction. Though omitted in this diagram, various enzymes necessary will usually be included as part of the diagram.

This book explores such pathways in order to understand biological systems *in silico*.

$$\text{Ethanol} \longrightarrow \text{Acetaldehyde} \longrightarrow \text{Acetic acid}$$

Fig. 1.3

Chapter 2
Pathway Databases

Pathway information is available through a large number of databases ranging from high-quality databases created by professional curators to massive databases, covering a vast number of putative pathways, created through natural language processing and text mining of abstracts. Because of the various differences in size, quality, and/or property, it is necessary to use the right database for the user's purpose, regardless of whether it is for commercial or for public use. In this chapter we introduce some of the major pathway databases. These databases can display pathway diagrams, which combine metabolic, genetic, and signal networks based on the literature. This chapter also covers some software applications for the production, editing, and analysis of such pathways.

2.1 Major Pathway Databases

Pathway databases are being created all around the world. Each database strongly reflects its builder's intent and purpose. There are databases with detailed metabolic pathways, while others have detailed signaling pathways. Most databases are created by curators who read papers and extract pathway information which will be organized together with pathway diagrams in the databases. Others are created using natural language processing and text mining, which extract from papers various biological relations such as gene regulatory relations and organize them into databases. This chapter covers those databases focused on metabolic and signaling pathways.

Pathway information is often described in the XML (eXtensible Markup Language) data format, which varies from database to database. This format can be easily read by both computers and humans. The following example shows the information "The lecture with Id "5" will be given on 4/1/2007 by a person named "masao nagasaki" in XML format:

```
<lecture id="5">
     <date>2007-04-01</date>
```

```
        <person>masao nagasaki</person>
  </lecture>
```

In the following chapters, we use acronyms ending with "... ML". This ending simply indicates that the pathway information is stored in some variant of XML. In this book, we do not go into the details of XML.

COLUMN 2

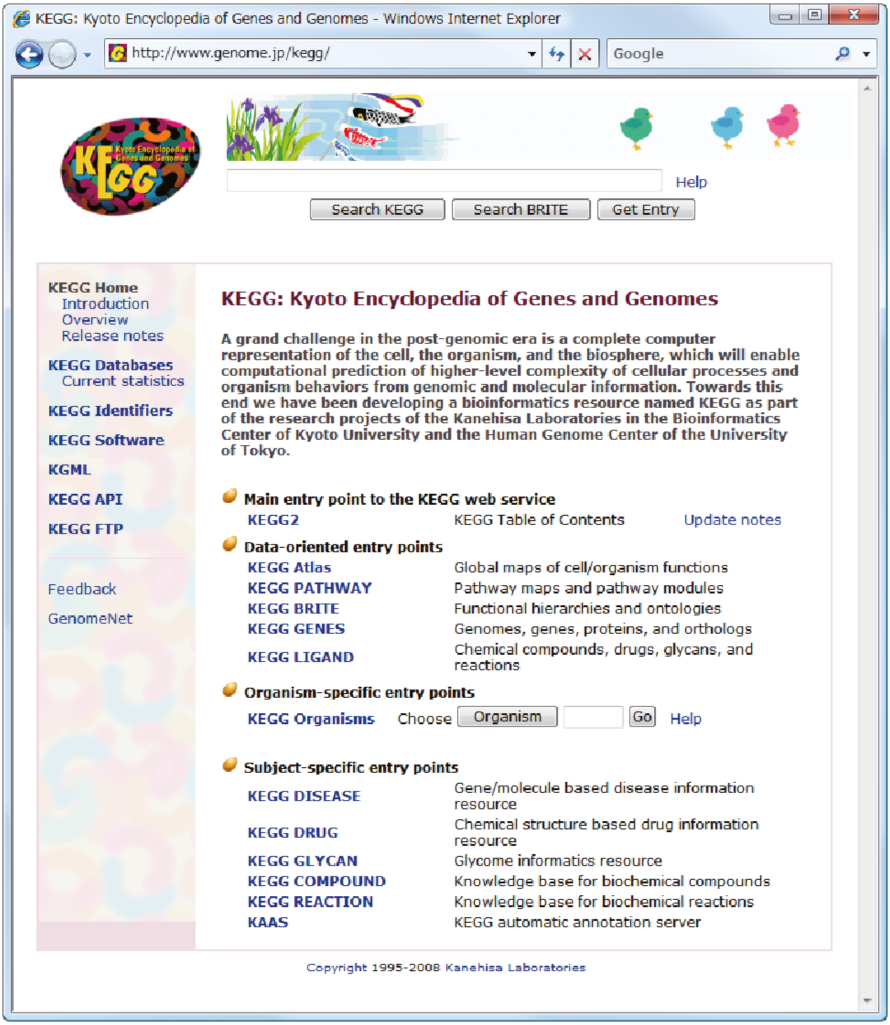
What's XML?

XML is one of many self-extensible markup languages. Its proper name is Extensible Markup Language. A markup language uses a sentence structure to list and categorize information. XML was developed in 1996 by the XML Working Group, part of the international standardization organization W3C. Because the creator can define and share a file format, a creator can use a standardized XML format for multiple applications, while allowing for a high degree of expression not constrained by the syntax.

2.1.1 KEGG

KEGG (Kyoto Encyclopedia of Genes and Genomes) (http://www.kegg.jp/) is a series of databases developed by both the Bioinformatics Center of Kyoto University and the Human Genome Center of the University of Tokyo. This database has been available for over 10 years. As the name encyclopedia suggests, the database includes information necessary for systems understanding of biology, such as genome sequences and chemical information (Figure 2.1). With its goal of collecting all knowledge relevant to biological systems, including the environmental information, KEGG will be a true encyclopedia. The "Pathway" section of KEGG consists mainly of metabolic pathways. For noncommercial uses, the license is free, while for commercial uses, the license is sold from Pathway Solutions Inc. (http://www.pathway.jp/).

KEGG is unique for its focus and coverage of yeast, mouse, and human metabolic pathways. Currently, signaling pathways for cell cycles and apoptosis are being expanded. New pathways are created by professionals (*curators*) who read and summarize the relevant literature. The information is displayed as a browser-viewable



KEGG: Kyoto Encyclopedia of Genes and Genomes

A grand challenge in the post-genomic era is a complete computer representation of the cell, the organism, and the biosphere, which will enable computational prediction of higher-level complexity of cellular processes and organism behaviors from genomic and molecular information. Towards this end we have been developing a bioinformatics resource named KEGG as part of the research projects of the Kanehisa Laboratories in the Bioinformatics Center of Kyoto University and the Human Genome Center of the University of Tokyo.

Main entry point to the KEGG web service

KEGG2	KEGG Table of Contents	Update notes

Data-oriented entry points

KEGG Atlas	Global maps of cell/organism functions
KEGG PATHWAY	Pathway maps and pathway modules
KEGG BRITE	Functional hierarchies and ontologies
KEGG GENES	Genomes, genes, proteins, and orthologs
KEGG LIGAND	Chemical compounds, drugs, glycans, and reactions

Organism-specific entry points

KEGG Organisms Choose Organism Go Help

Subject-specific entry points

KEGG DISEASE	Gene/molecule based disease information resource
KEGG DRUG	Chemical structure based drug information resource
KEGG GLYCAN	Glycome informatics resource
KEGG COMPOUND	Knowledge base for biochemical compounds
KEGG REACTION	Knowledge base for biochemical reactions
KAAS	KEGG automatic annotation server

Copyright 1995-2008 Kanehisa Laboratories

Fig. 2.1

pathway diagram. For example, one could search for the existence of a metabolic pathway from substance A to B, or the required enzymes for such a reaction. In addition, the database has links to relevant information such as genome sequences, positions, and conditions. The database is stored in a format called *KEGGML*. Since the pathways are then displayed as GIF files, the user cannot easily edit the pathway information.

2.1.2 BioCyc

BioCyc is a pathway database provided by SRI International (http://www.biocyc.org/). The database is a high-quality database focused on metabolic pathways originally formed by SRI International's bioinformatics research group. Related to BioCyc are the EcoCyc, MetaCyc, HumanCyc databases. Licenses are free for academic and nonprofit uses. Humans and *E. coli* are the major organisms listed with a variety of others. EcoCyc is mainly a database of *E. coli* metabolic pathways. These reactions are shown in the form of chemical equations. EcoCyc also contains a small number of signaling pathways. Curators extracted the pathway knowledge from the literature. Pathways are described with a proprietary format.

In addition, gene regulatory information upstream of the metabolic pathways is also listed. In other words, there is a link from a metabolic pathway to the genes coding enzymes and its regulators. The pathway map displays are separated in levels of detail. At the most detailed level, the metabolic products are shown in terms of the chemical equations.

2.1.3 Ingenuity Pathways Knowledge Base

Ingenuity Pathways Knowledge Base (IPKB) is the pathway database created by Ingenuity Systems Inc. (http://www.ingenuity.com/). All licenses, including academic and nonprofit, require a fee. The database consists of gene regulatory and signaling pathways. Curators extract knowledge from the literature for this database, which currently contains human, mouse, and rat genetic information. (As of May 2008, the website claims 13,600 human genes, 11,000 mouse genes, and 6,600 rat genes cataloged.) The database uses the Ingenuity Pathways Analysis (IPA) software mentioned later to view and analyze pathway data and thus IPKB is inaccessible through a web browser. Like KEGG and BioCyc, IPKB uses its own internal format for storage. However, unlike KEGG and BioCyc, IPKB allows for the editing of pathways through IPA. This edited data can later be exported as a graphic format such as SVG.

2.1.4 TRANSPATH

TRANSPATH is a gene regulatory and signaling pathway database created by BIOBASE (http://www.biobase-international.com/). The most recent version of the data requires a fee for both nonprofit and commercial uses. However, some parts of the old data are provided to academic users as a trial version (http://www.gene-regulation.com/). In addition to TRANSPATH, BIOBASE offers the TRANSFAC database of transcription factors and PROTEOME database of protein. It also provides a software ExPlain which combines and analyzes these databases.

TRANSPATH is formed similarly to those listed above through curators and therefore maintains high quality. Pathways are listed using a proprietary format. If the user has a license, the pathways are viewable from a web browser. In addition, it is possible to download the data stored as text file. For example, the phosphorylation of I-κB is shown below:

```
IkappaB-alpha, IkappaB-beta:p50:RelA +
ATP-IKK-alpha{p}:IKK-beta{p}:(IKK-gamma)2
-> IkappaB-alpha, IkappaB-beta{pS}:p50:RelA +
ADP (phosphorylation)
```

Each reaction has a link to the literature that confirms its existence. Therefore it is easy to understand what each biochemical reaction means. Figure 2.2 shows the IL-1 pathway displayed via a web browser, while Figure 2.3 displays the reaction information from TRANSPATH shown through a web browser. (As of May 2008, the website claims a total of 135,563 reactions mainly for human, mouse, and rat.)

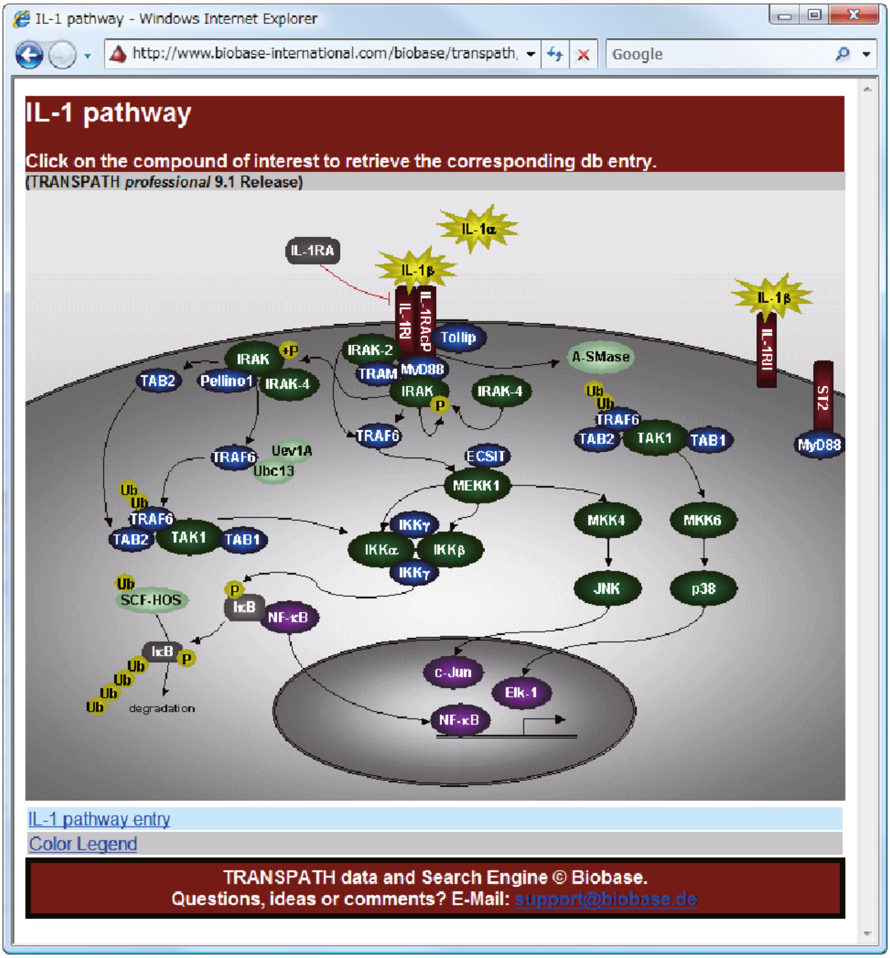
2.1.5 *ResNet*

ResNet (http://www.ariadnegenomics.com/) is the pathway database created by Ariadne Genomics. Academic and commercial licenses require a fee. The pathways of ResNet consist mainly of gene regulatory and signaling pathways. Unlike other databases, ResNet is constructed through computer analysis. In other words, the pathways and networks are created through natural language processing of relevant literature. MedScan is used for this natural language processing procedure. The database is constructed mainly from abstracts in PubMed, but some entries make use of the full text. In addition, there are a small number of entries created by curators.

The pathway data created by MedScan can be viewed through the viewing tool Pathway Studio. Similarly to other databases, MedScan uses its own proprietary format. ResNet employs arrows with various labels to show the relationships between molecules. '$+$' indicates activation, while '$-$' indicates suppression. Relationships which cannot be determined are indicated with '?'. In addition, comments are attached to the relation for nontrivial biological information. All such data are completely user editable.

2.1.6 *Signal Transduction Knowledge Environment (STKE): Database of Cell Signaling*

The database of Cell Signaling, a part of Signal Transduction Knowledge Environment (STKE) (http://stke.sciencemag.org/), is an online service provided by Science. This is a high-quality signaling pathway database created and maintained by curators. The database can be accessed by subscribing to the online service of Sci-



IL-1 pathway

Click on the compound of interest to retrieve the corresponding db entry.

(TRANSPATH *professional* 9.1 Release)

IL-1 pathway entry

Color Legend

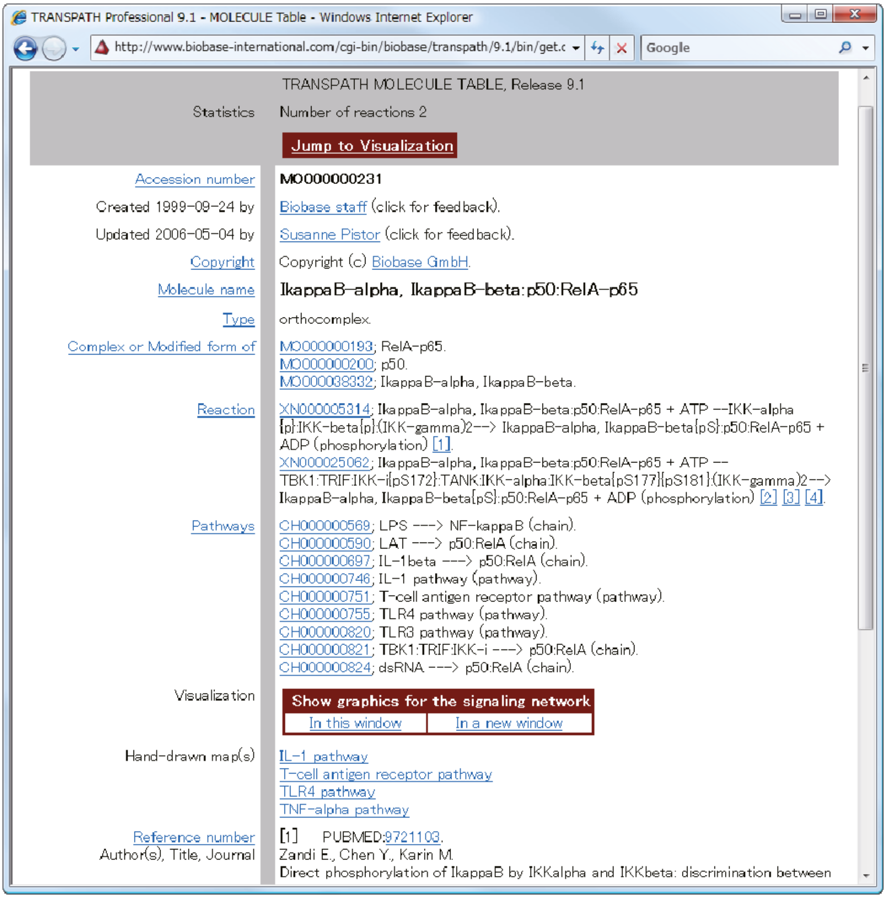
TRANSPATH data and Search Engine © Biobase.
Questions, ideas or comments? E-Mail: support@biobase.de

Fig. 2.2

ence although user registration does grant limited functionality such as pathway viewing. This database is accessible in GIF or SVG format through a web browser. Similarly to KEGG and BioCyc, this makes the pathway uneditable in browser. Similarly to ResNet, this database makes use of the labels ‘+’ for stimulatory relations, ‘−’ for inhibitory relations, ‘0’ for neutral relations, and ‘?’ for undefined relations. A feature of this database is the separation of pathways into “specific” and “canonical”. Specific pathways are those which are unique to an organism, while canonical pathways are those which are common. Unlike TRANSPATH or ResNet, however, the user cannot specify a list of genes (proteins) and create a network on that selection.

The following information is available in this database (as of March 2007):

- Cell Biology (46 pathways)


TRANSPATH Professional 9.1 - MOLECULE Table - Windows Internet Explorer
http://www.biobase-international.com/cgi-bin/biobase/transpath/9.1/bin/get.c
Google

TRANSPATH MOLECULE TABLE, Release 9.1

Statistics	Number of reactions 2 Jump to Visualization
Accession number	MO000000231
Created 1999-09-24 by	Biobase staff (click for feedback).
Updated 2006-05-04 by	Susanne Pistor (click for feedback).
Copyright	Copyright (c) Biobase GmbH.
Molecule name	IkappaB-alpha, IkappaB-beta:p50:RelA-p65
Type	orthocomplex.
Complex or Modified form of	MO000000193; RelA-p65. MO000000200; p50. MO000038332; IkappaB-alpha, IkappaB-beta.
Reaction	XN000005314; IkappaB-alpha, IkappaB-beta:p50:RelA-p65 + ATP --IKK-alpha {p}:IKK-beta{p}:(IKK-gamma)2--> IkappaB-alpha, IkappaB-beta{pS}:p50:RelA-p65 + ADP (phosphorylation) [1]. XN000025062; IkappaB-alpha, IkappaB-beta:p50:RelA-p65 + ATP -- TBK1:TRIF:IKK-i{pS172}:TANK:IKK-alpha:IKK-beta{pS177}{pS181}:(IKK-gamma)2--> IkappaB-alpha, IkappaB-beta{pS}:p50:RelA-p65 + ADP (phosphorylation) [2] [3] [4].
Pathways	CH000000569; LPS ---> NF-kappaB (chain). CH000000590; LAT ---> p50:RelA (chain). CH000000697; IL-1beta ---> p50:RelA (chain). CH000000746; IL-1 pathway (pathway). CH000000751; T-cell antigen receptor pathway (pathway). CH000000755; TLR4 pathway (pathway). CH000000820; TLR3 pathway (pathway). CH000000821; TBK1:TRIF:IKK-i ---> p50:RelA (chain). CH000000824; dsRNA ---> p50:RelA (chain).
Visualization	Show graphics for the signaling network In this window | In a new window
Hand-drawn map(s)	IL-1 pathway T-cell antigen receptor pathway TLR4 pathway TNF-alpha pathway
Reference number Author(s), Title, Journal	[1] PUBMED:9721103. Zandi E., Chen Y., Karin M. Direct phosphorylation of IkappaB by IKKalpha and IKKbeta: discrimination between

Fig. 2.3

- Developmental and Reproductive Biology (32 pathways)
- Immune, Inflammatory, and Defense Signaling (17 pathways)
- Microbiology (6 pathways)
- Neurobiology (5 pathways)
- Plant Biology (15 pathways)
- Stress, Death, and Survival Signaling (9 pathways)
- Pathways Implicated in Human Disease (11 pathways)

2.1.7 Reactome

Reactome is a pathway database containing cell metabolic and signaling pathways (http://www.reactome.org/). Cold Spring Harbor Laboratory, European Bioinformatics Institute, and Gene Ontology Consortium—which specifies Gene Ontology mentioned later—are the main developers of the project. Although humans are the main organism catalogued, it has data for 22 other species such as mouse and rat. Pathway knowledge is extracted by curators.

Reactome's pathways and reactions can be viewed but not edited through a web browser. Though the storage format is proprietary, a large number of pathways can be obtained in multiple formats. Human reactions are distributed through SBML format, human protein relations are given through TSV format, and cellular event information is given through the BioPAX format listed in Section 2.3.5. All data can easily be downloaded and edited.

2.1.8 Metabolome.jp

Metabolome.jp (http://metabolome.jp/) is a metabolic pathway-focused database created by some research labs led by the University of Tokyo Graduate School of Frontier Sciences. Using an applet called ARM, pathways can be viewed and edited through a browser. Pathways are created by curators. Each metabolic product is shown with an atomic structural formula and it is possible to display a pathway which considers atom movements. Unlike KEGG, it is possible to track the movement of atoms in metabolic reactions. Pathway storage uses a proprietary format.

2.1.9 Summary and Conclusion

As described above, a variety of databases are available. The databases vary in the types of information offered; there are metabolic pathway databases and signaling pathway databases. In addition, there are differences in the organisms covered by the databases. However, a common problem is that these databases do not have enough information to permit simulating the pathways.

Pathway databases are constructed by curators or through the use of natural language processing and text mining tools via computer. This difference affects the characteristics of the databases significantly. Through methods such as natural language processing, one has the advantage of a large breadth of literature which curators are unable to cover. In addition to the quality problem, however, there is usually the problem of lacking specific biological or experimental facts listed in the database. Although it is likely that this technology will be improved in the future, such databases are currently ancillary to those created by curators (such as IPKB or TRANSPATH). Databases created by curators are on the whole more reliable and detailed. Each pathway database has its own proprietary format. Although there are formats such as SBML and BioPAX (mentioned later) which aim at standardizing these formats, the current situation is not satisfactory in practice.

In addition to the databases introduced here, there are many other good pathway databases. Some of them are:

- BioCarta: Signaling pathways (http://www.biocarta.com/)
- INOH: Signaling pathways (http://www.inoh.org/)
- iPath: Signaling pathways (http://www.invitrogen.com/content.cfm?pageid=10878)
- Molecular Interaction Map: Signaling pathways as well as gene regulatory networks (http://discover.nci.nih.gov/mim/index.jsp)

There are a myriad of databases which are not listed here. It is likely that databases—whether or not they are listed here—will develop or disappear for a variety of reasons: "Research fund is terminated."; "The government fully supports the database."; "The database is commercialized." When using a database, the following items will be a useful guideline for assessment.

- Is the database viewable through a web browser?
- Is there a licensing fee?
- What is the data type (metabolic, gene regulatory, signaling, etc.)?
- Is the database developed through computer or curator?
- Is there any software for editing pathways?
- Is it possible to simulate the pathway?

2.2 Software for Pathway Display

Pathway information must somehow be displayed. In this section, we introduce software applications that help visualize pathways.

2.2.1 Ingenuity Pathway Analysis (IPA)

Ingenuity Pathway Analysis (IPA) is the software used to display pathway data from the Ingenuity Pathway Knowledge Base (IPKB) by Ingenuity Systems Inc. For a given gene set, IPA automatically generates the pathways that are related to those genes. This means that, for example, if one finds a set of genes with large gene expression variance as a result of microarray analysis, IPA automatically generates the pathway which involves those genes. The pathway is generated with a mixture of human, mouse, and rat data. Therefore, it should be cautioned that there can be no pathway in the real organism of the user's interest even if IPA generates some pathway.

2.2.2 Pathway Builder

Pathway Builder is a viewer that automatically generates pathways from the TRANSPATH database (http://www.biobase-international.com/). Pathway Builder can find the pathways related to a set of genes and connect them to display as one pathway. This allows to search and display genes upstream and downstream of the genes in the set. Using this feature, one can find the genes whose transcriptions are activated by a gene (downstream search) or find the genes which regulate a particular gene (upstream search).

2.2.3 Pathway Studio

Pathway Studio is the viewer for Ariadne Genomics' ResNet. Pathway Studio has a function to add new molecules and user's information into the pathway. The automatic layout feature is one of the unique parts of this viewer. Like IPA and Pathway Builder, Pathway Studio can search with gene names and create a pathway of genes related to any given gene (or protein).

2.2.4 Connections Maps

Connections Maps is a viewer for Signal Transduction Knowledge Environment (STKE): Database of Cell Signaling. This program creates the GIFs and SVGs of the pathways according to the data created by curators called "Pathway Authorities". Genes and proteins have specific set symbols and colors, and the relations are indicated with '$+$' (activation), '$-$' (repression), and '?' (undefined). In addition, the graphics have embedded links, which make it simple to get more detailed information. Because of the SVG format, the user is free to magnify any level of the pathway. However, Connections Maps is unable to generate custom pathways from a list of genes, unlike IPA and Pathway Builder.

2.2.5 Cytoscape

Cytoscape is a software tool designed to visualize the molecular interactions as a network diagram (http://www.cytoscape.org/). It was developed mainly by the Institute for Systems Biology and University of California San Diego as well as some other institutions such as the Pasteur Institute, MSKCC, Agilent, and UCSF as an open source project. The program is free to download and it requires the use of Java; the current version (as of April 2008) is 2.6.0.

The software can use protein–protein binding information, protein–DNA binding information, and microarray data to provide a network view. The network visualization is proprietary. Proteins and genes are shown as circles, triangles, and squares (called nodes), while relationships are shown as lines (called edges). In addition to nodes and edges, various attributes such as Gene Ontology or wet lab measurements of expression can be added. Cytoscape has a filtering feature to show only the network of interest. By using Gene Ontology in combination with filtering, it is possible to show all the genes with a certain function.

In addition, analysis functionality can be provided as plugin. A number of plugins have been developed for a variety of purposes. For example, a plugin provided by Agilent allows Cytoscape to extract protein and genome information from textual abstracts and display the results as a network.

A variety of storage formats can be imported, such as Simple Interaction File (SIF), Graph Markup Language (GML), Extensible Graph Markup and Modeling Language (XGMML), SBML, BioPAX, and PSI MI. Of these, GML and XGMML are standard XML formats for graph (a set of vertices connected with edges) formation. SBML, BioPAX, and PSI MI will be mentioned later. The SIF format is, as the name states, a simple format for showing interactions. For example, if protein A and protein B act upon each other, one would simply put the interaction type between the names and write in the following way:

`A pp B` (`pp` stands for protein-protein interaction)

2.3 File Formats for Pathways

2.3.1 Gene Ontology

Gene Ontology (GO) defines a common framework to organize biological concepts (http://www.geneontology.org/). Ontology was originally studied in Artificial Intelligence and is defined as "a hierarchical taxonomy of terms for a certain area of knowledge". The GO project began in the 1990s, and seeks to record genetic and functional information in the same syntax to simplify database comparison. The terms defined by GO are called GO terms and can be divided into the following three categories:

- Biological processes
- Cellular components
- Molecular functions

These categories have terms such as "nuclear chromosome", "chromosome", "nucleus", and "cell". Between these terms are relationships such as "*is_a*" as in "nuclear chromosome *is_a* chromosome" or "*part_of*" as in "nucleus *part_of* cell". These relationships are called *ontologies*. The relationships between such terms are listed in a directed acyclic graph (DAG). A consortium has been formed adopting GO and there are a large number of databases contributing to the project.

2.3.2 PSI MI

Proteomics Standards Initiative (PSI) began around 2002 and attempts to standardize data from mass spectrometry and protein–protein interaction experiments, in order to facilitate data comparison and transfer (http://psidev.sourceforge.net/). PSI MI is defined to handle information on protein-protein interactions.

2.3.3 CellML

CellML is the first Systems Biology XML format to integrate cellular level molecular dynamics as a part of its format. Over 300 models have already been submitted and displayed at the CellML Repository (http://www.cellml.org/). It is a format developed by the University of Auckland in New Zealand under the auspices of the International Physiome Project. CellML 1.0 was published in 2000 and CellML1.1 is currently proposed. CellML is structured to include model structures, differential equations-based dynamics information, and additional comments. To store all these, CellML utilizes MathML, a math typesetting format for XML. The format seeks to describe everything from the cellular to organ level by combining with FieldML (http://www.physiome.org.nz/xml_languages/fieldml/).

2.3.4 SBML

SBML (Systems Biology Markup Language) is one of the XML formats designed to model biological reactions (http://www.sbml.org/). In 2001, SBML level 1 was released, and in 2003, SBML level 2 was released. Like CellML, SBML was expanded to include MathML support, spatial position and physical size information.

As of May 2008, SBML 2.3 is the current version. Currently, this format is actively heading towards level 3 release. An open source application called *SBW* (Systems Biology Workbench) has been developed to combine with other simulation and analysis software for use with SBML. In addition, a database called *BioModels* (http://www.ebi.ac.uk/biomodels/) based upon SBML, though small, has been under development.

2.3.5 BioPAX

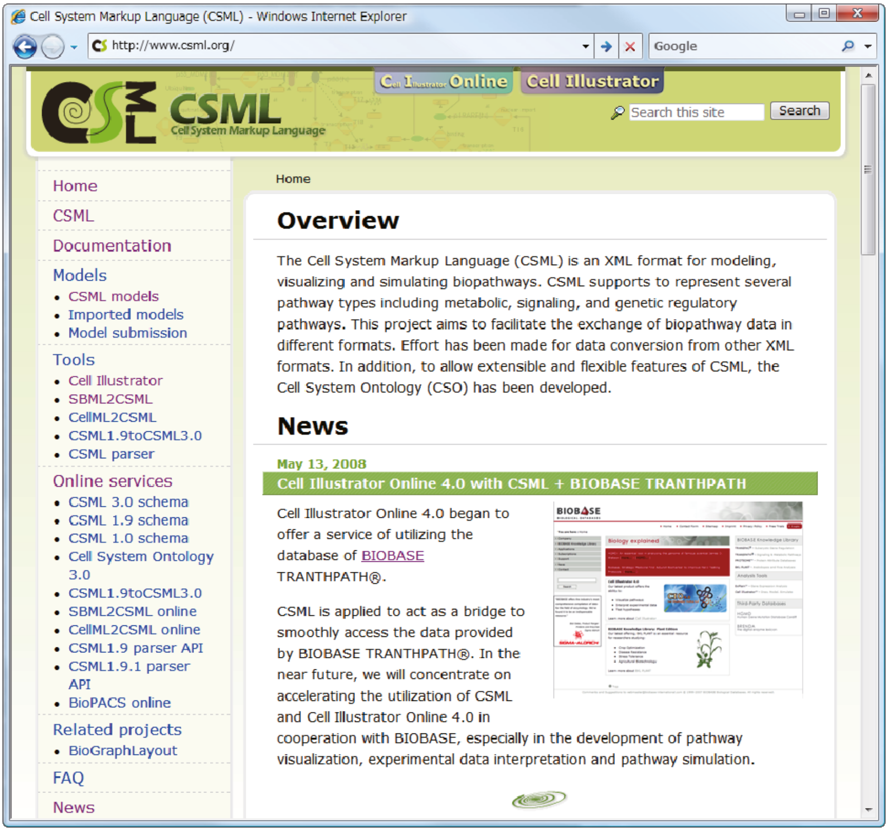
BioPAX was started in 2002 in order to encourage open source formats for pathway information (http://www.biopax.org/). The format is defined by using OWL (an XML type language used to define ontologies). BioPAX level 1 targets information regarding compounds and metabolism. BioPAX level 2 targets molecular relationships and includes information on molecular bindings, phosphorylation sites, posttranslation modifications, as well as experimental data and pathway structures. Discussions on BioPAX level 3 are developing so that it will include gene regulatory and signaling pathways.

2.3.6 CSML/CSO

CSML (Cell System Markup Language) is an XML format designed to define gene regulatory, metabolic, and signaling pathways with regard to system dynamics (http://www.csml.org/). It has been developed at the Human Genome Center of the University of Tokyo.

As of May 2008, CSML 3.0 is the newest version. In addition, CSML is widely extensible and can import the CellML and SBML formats introduced in Sections 2.3.3 and 2.3.4.

Furthermore, in order to achieve a high level of compatibility with other data formats, CSML defines and uses its own ontology format, Cell System Ontology (CSO). CSO is an ontology which effectively describes dynamics and signal pathways not expressible by BioPAX introduced in Section 2.3.5. In addition, CSO defines a large number of standardized icons (over 350) to be used for defining necessary terms and relations (see Chapter 4, Figure 4.37). CSML pathways are displayed in Cell Illustrator—software which will be described in Chapter 3—which uses these icons. CSML models can be downloaded from the above URL (Figure 2.4).



Overview

The Cell System Markup Language (CSML) is an XML format for modeling, visualizing and simulating biopathways. CSML supports to represent several pathway types including metabolic, signaling, and genetic regulatory pathways. This project aims to facilitate the exchange of biopathway data in different formats. Effort has been made for data conversion from other XML formats. In addition, to allow extensible and flexible features of CSML, the Cell System Ontology (CSO) has been developed.

News

May 13, 2008

Cell Illustrator Online 4.0 with CSML + BIOBASE TRANTHPATH

Cell Illustrator Online 4.0 began to offer a service of utilizing the database of BIOBASE TRANTHPATH®.

CSML is applied to act as a bridge to smoothly access the data provided by BIOBASE TRANTHPATH®. In the near future, we will concentrate on accelerating the utilization of CSML and Cell Illustrator Online 4.0 in cooperation with BIOBASE, especially in the development of pathway visualization, experimental data interpretation and pathway simulation.

Fig. 2.4

Chapter 3
Pathway Simulation Software

In Chapter 2, we surveyed some pathway databases that are currently available. In this chapter, we will present some of the pathway simulation software tools. While pathway databases provide the information on a pathway with biological facts mapped on a pathway illustration, most simulation software tools assume models to be described with differential equations and programs, where variables represent the concentrations of molecules and events which are hard to represent with differential equations are described as a program in a general programming language, e.g., C++. This may be one of the reasons why the simulation approach was not fully accepted by the communities of molecular biology and medical sciences—these descriptions are far from intuitive biological understanding. Generally speaking, a *model* is a system represented with some kind of dynamical systems such as differential equations, and the process of creating such a model is called *modeling*. Recent advances in graphical user interfaces (GUIs) of software applications for biological pathway modeling and simulation have made it possible to create and simulate pathways in a way just like “drawing”.

3.1 Simulation Software Backend

There are two key factors to keep in mind when evaluating simulation software applications. The first is the method in simulation engine which we will call the *architecture*. The second is the GUI used for modeling pathways. There are also some additional matters to consider, e.g., license fees, OS (Windows, Linux, Mac OS X), compatibility between models.

3.1.1 Architecture: Deterministic, Probabilistic, or Hybrid?

This section is for readers who are interested in the architectures employed in software application's. Some technical terms are used without any detailed explanations. Other readers can safely skip Section 3.1.1 since the rest of this book is not dependent on this section.

Architecture, as used here, is the method used to describe a model. If the behavior of an event (system) is deterministic and continuous, Ordinary Differential Equations (ODEs) and Partial Differential Equations (PDEs) are generally used for the model. For example, enzyme reactions are often described with ODEs. In the case of ODEs and PDEs, models can be simulated relatively fast. Furthermore, if a model can be described with an established framework with ODEs, we can efficiently build the model by selecting appropriate coefficients. In the case of a small system, this enables rigorous mathematical analyses. If the event for modeling should include probabilistic behaviors, we need to add other features. For the events involving discrete behaviors such as switches, they can be approximated with special ODEs.

For such probabilistic events, Gillespie's Direct method (GD), Gibson-Bruck next reaction method (GB), Firth-Bray multistate stochastic method (FB), Gillespie Tau-Leap method (TL), and Stochastic Petri Net method (SPN) are generally used. The details of each architecture are outside the scope of this book. One feature of these methods is that they can simplify modeling. At the same time, however, these methods require more simulation time and the analysis of behaviors is usually more complicated. Therefore these architectures are not a panacea and it may not be appropriate to use these probabilistic methods if the details of the reactions are not well-known. Of course, some events essentially require probabilistic features, and it is important to consider the degree to which probabilistic action affects the model.

In addition to those two categories mentioned before, there are hybrid architectures such as Vasudeva-Bhalla method (VB), Haseltine-Rawlings method (HR), and Hybrid Functional Petri Net. These architectures allow for a higher degree of flexibility than either deterministic or probabilistic models.

3.1.2 Methods of Pathway Modeling

As introduced in Chapter 2, there are many pathway model formats. Creating a pathway model is to describe it with one of the formats which should include dynamics in the pathway. The following can be considered to do this:

1) Utilize scripting and programming languages directly, such as C++, Java.
2) Write directly in a text format such as XML.
3) Use a GUI embedded in the simulation software. There are a variety of types in GUIs that range from simple spreadsheets to highly human friendly drawing canvases.

3.2 Major Simulation Software Tools

Although there are many pathway simulation tools, some of them require some programming skills. The software tools described here are those which are relatively easy to use and install.

3.2.1 Gepasi/COPASI

Gepasi is one of the first programs developed to simulate intracellular reactions (http://www.gepasi.org/). It runs on Windows. COPASI (http://www.copasi.org/) is the successor to Gepasi, and it also runs on UNIX and Mac. These programs are provided free of charge for noncommercial use. Intracellular reactions are modeled using equations called "reactions"; these reactions are entered through a basic GUI. The simulation results are displayed as a graph, and the architecture is based on ordinary differential equations.

3.2.2 Virtual Cell

Virtual Cell (http://www.vcell.org/) is a cell modeling environment provided and developed by the University of Connecticut Health Center. The program is freely available to any researcher and distributed through an online environment called Java Web Start. The program has a GUI with which the user can construct a pathway by placing and connecting parts inside a cell diagram. A unique feature of this program is that it can create 3D models. Models created using Virtual Cell are shown on the website. The architecture is based on ordinary and partial differential equations.

3.2.3 Systems Biology Workbench (SBW), Cell Designer, JDesigner

Systems Biology Workbench (SBW) is a simulation environment for the SBML format. It is freely available on the project website (http://sbw.sourceforge.net/). SBW, like some old programs, is a bare-bones simulation engine. Therefore, it must be combined with a frontend such as Cell Designer (http://www.systems-biology.org/cd/) or JDesigner (http://www.sys-bio.org/) which does not have the simulation capability. Cell Designer and JDesigner have GUIs with which the user can create models through parts selection and placement. SBW imports and simulates these models. The architecture is a hybrid one (GD, GB like).

3.2.4 Dizzy

Dizzy is a Java simulator developed by the Institute for Systems Biology to simulate probabilistic and deterministic biochemical reactions. As of May 2008, the latest version is 1.11.3. Simulation results are shown as a graph. The architecture uses ordinary differential equations for deterministic reactions and Gilespie, Gibson-Bruck, and Tau-Leap for probabilistic ones. Although it is possible to select either of these two methods, it is not possible to mix both probabilistic and deterministic reactions in one model. The program operates under Windows, Linux, and OS X.

3.2.5 E-Cell

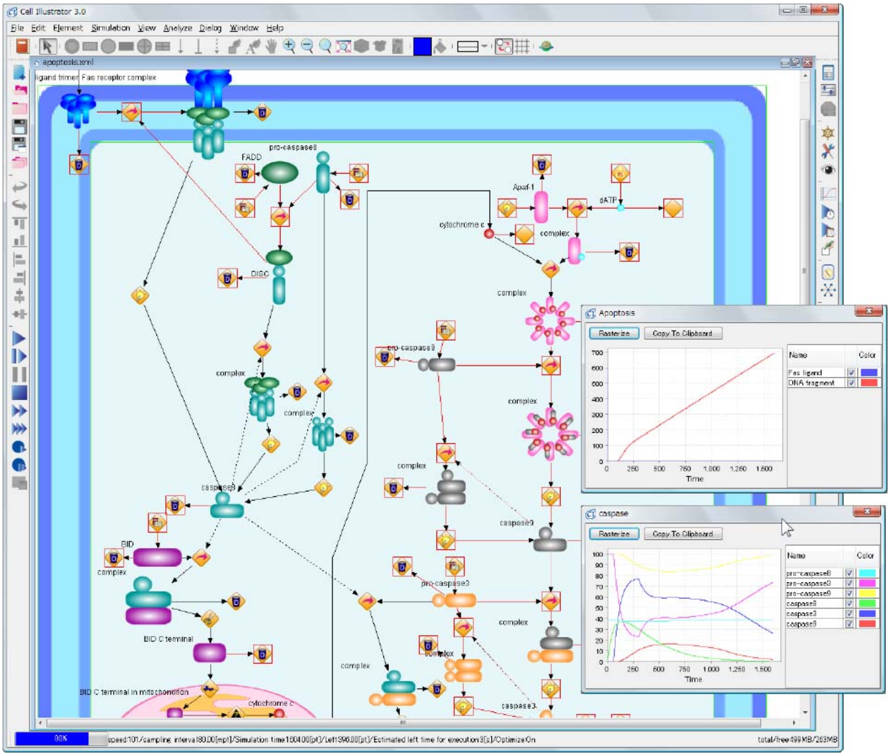
E-Cell is a pioneering program developed by Keio University for modeling metabolic reactions (http://www.e-cell.org/). The GUI can display the results but cannot display the pathway. As of February 2008, version 3.1.106 has been released under GPL for Windows, Linux, and Mac OS X. Although very few models have been created with E-Cell, it takes significant skills to quickly create a model. The architecture is a hybrid type.

3.2.6 Cell Illustrator

Cell Illustrator is a program developed by the Human Genome Center of the University of Tokyo. As of May 2008, the latest version is 3.0. The online version is also released as Cell Illustrator Online 4.0 (see Section 6.3). It is currently distributed from http://www.cellillustrator.com/. Unlike SBW mentioned in Section 3.2.3, both drawing and simulation of pathways can be performed on a single tool (Figure 3.1). Furthermore, the program automatically creates an underlying ontology through modeling (drawing). The architecture is Hybrid Functional Petri Net with extension (HFPNe) that is a highly flexible extension of Hybrid Functional Petri Net (HFPN). The models created on Cell Illustrator are saved in the CSML format. Other formats such as CellML and SBML can also be imported. University of Tokyo Institute of Medical Science, Yamaguchi University Graduate School of Science and Engineering, Queensland University Institute for Molecular Bioscience (IMB), and the Visible Cell project led by ARC Bioinformatics Center currently use Cell Illustrator and view it as a program capable of cutting-edge research.

From Chapter 4 onwards, this book will use Cell Illustrator to explain how to model and simulate pathways. The CD-ROM included with this book has the Book Edition of Cell Illustrator, which works under Windows, Linux, and Mac OS X.

Cell Illustrator also has a companion tool called *Cell Animator*, which can visualize simulation results as animations (Figure 3.2).



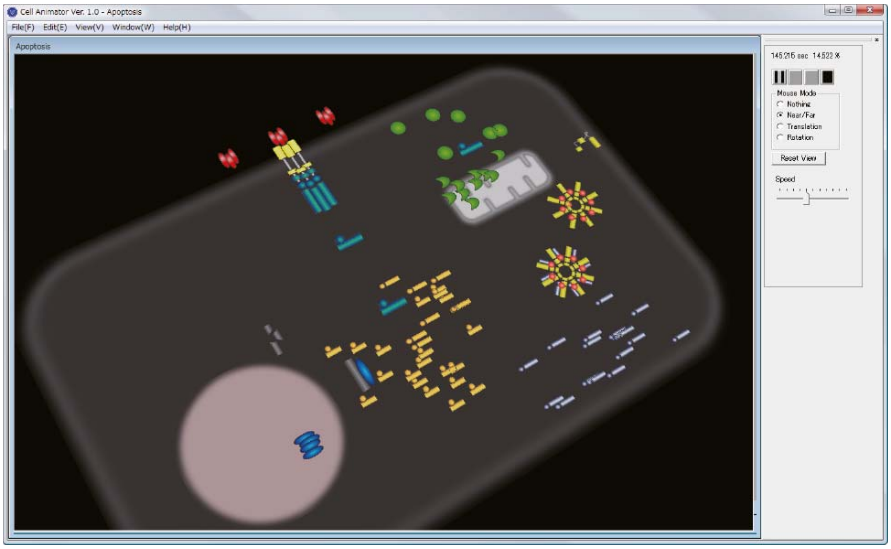
Name		Color
Fas ligand	☑	🟦
DNA fragment	☑	🟥

Name		Color
pro-caspase8	☑	🟦
pro-caspase3	☑	🟪
pro-caspase9	☑	🟨
caspase8	☑	🟩
caspase3	☑	🟦
caspase9	☑	🟥

speed:101/sampling interval80.00[mpt]/Simulation time1604.00[pt]/Left1396.00[pt]/Estimated left time for execution3[s]/Optimize:On

total/free499MB/263MB

Fig. 3.1


Cell Animator Ver. 1.0 - Apoptosis
File(F) Edit(E) View(V) Window(W) Help(H)
Apoptosis
Reset View
Speed

145.215 sec 14.522 %

Mouse Mode
○ Nothing
● Near/Far
○ Translation
○ Rotation

Fig. 3.2

3.2.7 Summary

Table 3.1 gives a comparison overview of the software tools introduced in this chapter. Because there are programs which rapidly develop or cease development (especially free programs), it is important to evaluate the long-term reliability of such software tools.

Table 3.1

Name	Quality of GUI	Architecture
Copasi	⋆	deterministic (ODE), analysis capabilities
Virtual Cell	⋆⋆	deterministic (ODE, PDE)
Cell Designer	⋆⋆⋆⋆	none
JDesigner	⋆⋆⋆⋆	none
SBW	none	hybrid (GD, GB, etc.)
Dizzy	⋆	hybrid (GD, DB, TL)
E-Cell	⋆	hybrid (HR)
Cell Illustrator	⋆⋆⋆⋆⋆	hybrid (HFPNe)

Chapter 4
Starting Cell Illustrator

In this chapter, we will use Cell Illustrator 3.0 (CI3.0) to understand the procedures for creating and simulating pathway models. No expert knowledge in differential equations or programming skill are required; the basic idea is to model and simulate by drawing a pathway.

4.1 Installing Cell Illustrator

4.1.1 Operating Systems and Hardware Requirements

Cell Illustrator operates under almost any OS (Operating System) that can run Java programs. Specifically, Cell Illustrator operates under Windows, Mac, and Linux. Install Java Runtime Environment (JRE) version 1.5.0 or later to run Cell Illustrator. There is no need to reinstall JRE if a newer version is already installed. Otherwise, either download JRE from http://java.sun.com, or use the installation packages for JRE 1.6.0 provided on the CD-ROM.

4.1.1.1 Supported Operating Systems

The following packages are prepared for Cell Illustrator depending on OS type and "with or without" JRE (Java VM):

- Windows NT/2000/XP/Vista (without Java VM)
- Windows NT/2000/XP/Vista (with Java VM)
- Linux (without Java VM)
- Linux (with Java VM)
- Mac OS X 10.4.8 (Tiger) or later (no version with Java VM)

4.1.1.2 Hardware Requirements

The following are the minimum requirements for running Cell Illustrator:

- CPU: 1 GHz
- RAM: 512 MB
- Disk Space: 100 MB

Under typical work load, the following are recommended specifications:

- CPU: over 2 GHz
- RAM: over 1 GB

4.1.2 Cell Illustrator Lineup

There are four versions of Cell Illustrator as of July 2007: Cell Illustrator Draw, Cell Illustrator Standard/Classroom, and Cell Illustrator Professional. CI Draw is free and retains full modeling capability, but lacks simulation capability. CI Standard/Classroom can simulate moderately sized pathways. (CI Classroom is the academic version of CI Standard.) CI Professional has no limitations, can create pathways of over 1000 elements, and can be used for the analysis of large-scale gene networks. The 30-day trial editions are available for all versions.

On the CD-ROM is a simplified version of Cell Illustrator 3.0, which has functionality between CI Draw and CI Standard. This version is more than capable of all of the exercises in Chapters 4 and 5. If it becomes necessary to upgrade for larger models, one license change is all that is needed. See http://www.cellillustrator.com/ for further information about licensing. Now let us approach Systems Biology with Cell Illustrator 3.0.

4.1.3 Installing and Running Cell Illustrator

This section covers the installation of Cell Illustrator. Though this book uses the installation to Windows Vista as an example, other Windows platforms should be similar.

4.1.3.1 Installation on Windows

4.1.3.1.1 Without JRE or unknown

1) Install by clicking CI3.0_wj_setup.exe in the CD-ROM.
2) After installation, CI will be registered in the start menu, click to start.

4.1.3.1.2 JRE is already installed.

1) Install by clicking CI3.0_setup.exe in the CD-ROM.
2) After installation, CI will be registered in the start menu, click to start.

4.1.3.2 Installation on Mac OS X

The Mac version of CI does not have an installer with JRE embedded. Mac OS X can update their JRE version by running “Software Update”. Note that Cell Illustrator cannot be installed in the “Classic” environment.

1) Move CI3.0x_m.tgz to any folder on your computer.
2) Double-click CI3.0x_m.tgz to start installation.
3) After installation, the icon of Cell Illustrator will appear on the desktop. To start, simply click the icon on the desktop.

4.1.3.3 Installation on Linux

Note that if there is already a JRE installed which is not version 1.5.0 or later, the user needs to manually upgrade the JRE. The embedded JRE installer does not work.

4.1.3.3.1 Embedded JRE

1) Move CI3.0x_lj.bin to any folder on your computer.
2) Run “chmod +x CI3.0x_lj.bin” from a terminal such as xterm.
3) Run “./CI3.0x_lj.bin” in the terminal (this will also install JRE).

To run Cell Illustrator, go to the newly made folder (usually called GNI) and run “./CI”.

4.1.3.3.2 Installation without JRE Embedded

1) Move CI3.0x_l.bin to any folder on your computer.
2) Run ”chmod +x CI3.0x_l.bin” from a terminal such as xterm.
3) Run ”./CI3.0x_l.bin” in the terminal” (this will also install JRE).

To run Cell Illustrator, go to the newly made folder (usually named GNI) and run “./CI”.

4.1.3.4 Installation on Unix

Since the Unix version lacks an embedded JRE installer, please install JRE separately if necessary.

1) Move CI3.0x_u.jar to any folder on your computer.
2) Run “chmod +x CI3.0x_u.jar”.
3) Run “./CI3.0x_u.jar”.

To run Cell Illustrator, go to the newly made folder (usually named GNI) and run "./CI".

4.1.4 License Install

If Cell Illustrator is installed using the installers on the CD-ROM, the license files will automatically be installed. For example, in Windows (for the default installation), the directory

"C:\Program Files\GNI\Cell Illustrator 3.0"

contains a file named license.txt, which is the license file.

For other versions such as CI Classroom or CI Professional, manually place the license files in the installation folder.

4.2 Basic Concepts in Cell Illustrator

4.2.1 Basic Concepts

Cell Illustrator uses a framework called the *Petri net* to model and simulate pathways. Carl A. Petri originally created the concept of Petri net as a mathematical description of concurrent systems. Cell Illustrator uses an extension of this concept for Systems Biology. This new concept is called a *Hybrid Functional Petri Net with extension* (HFPNe). In order to facilitate biological understanding, we use alternate terms to describe the notions in Petri net theory. As the basic architecture, Cell Illustrator employs the concept of HFPNe that leads to the following characteristics:

1) Pathway models can be built just like with a drawing tool.
2) Simulation can be done easily on the spot.
3) The same methodology can be applied to modeling simple pathways to complex systems.

On Cell Illustrator, we make a pathway model with three kinds of *elements*, called *entity*, *process*, and *connector*. Figure 4.1 shows the elements used in Cell Illustrator.

4.2.2 Entity

An entity describes "object" such as protein or mRNA. Three *types* of entities are defined: *discrete entity*, *continuous entity*, and *generic entity*. These three types of

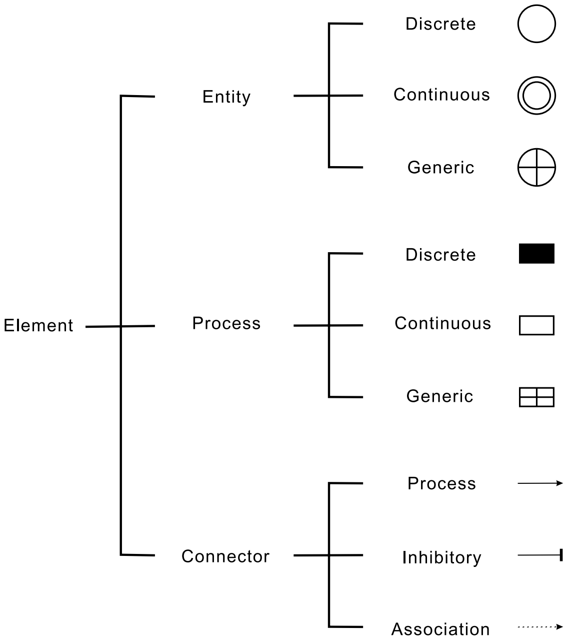


Fig. 4.1

entity have different characteristics, and therefore, an effective modeling of a pathway requires a proper use of these entities. These types are illustrated as shown in Figure 4.2.

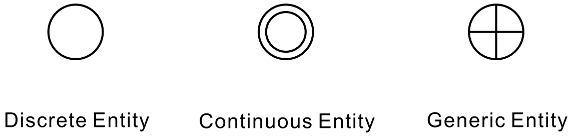


Fig. 4.2

A discrete entity has a discrete $(0,1,2,\ldots)$ value while a continuous entity can hold any nonnegative real value. For example, a discrete entity could be used to describe the number of molecules of a specific protein, while a continuous entity could be used to describe the concentration of the protein molecule. A generic entity can handle any type of object; its main use in Cell Illustrator is focused on integer, real number, string, and Boolean. We, however, omit the details about generic entities because the advanced applications of generic entity are outside the scope of this book.

Once placed on a *canvas*, an entity displays its three properties on the top left, top right, and bottom left, as seen in Figure 4.3. Creating a canvas will be explained further in Section 4.3. These three fields are called *Name*, *Variable*, and *Initial Value* which have the name of the entity, the variable name assigned to the entity, and the initial value of this variable, respectively. The default for Name is an "e" followed by a positive integer, while the default for Variable is an "m" followed by a positive integer. The default for Initial Value is 0.

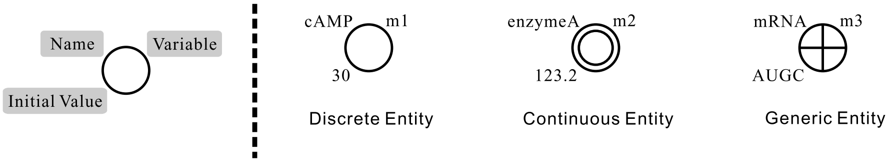


Fig. 4.3

There are three ways to modify the contents of the fields Name, Variable, and Initial Value:

Method 1: Double-click on the target text of the entity on the canvas and edit directly (Figure 4.4).

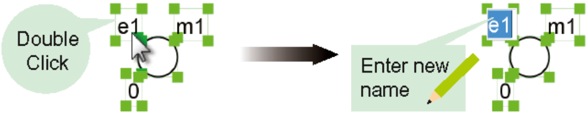


Fig. 4.4

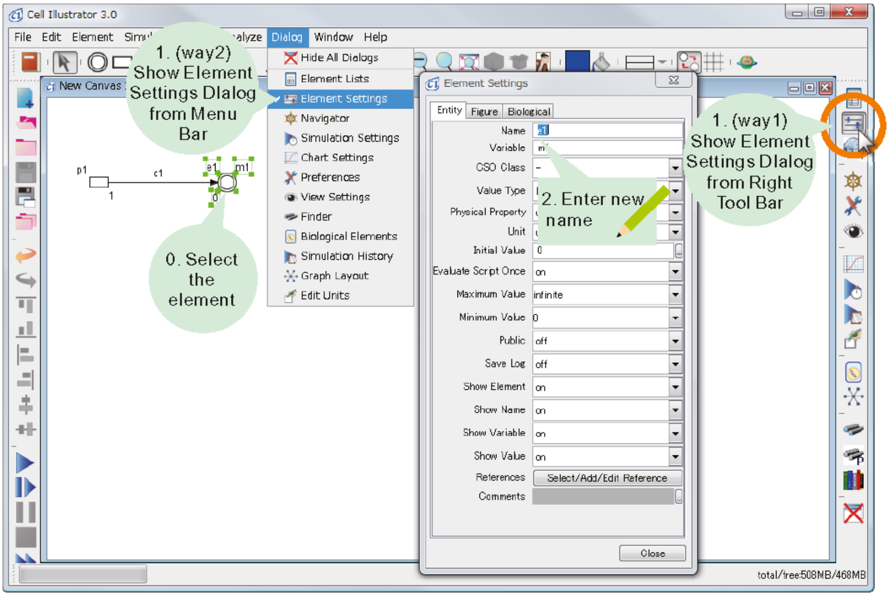
Method 2: Open the Element Settings Dialog and change the contents of the fields for Name, Variable, and Initial Value as shown in Figure 4.5.

Method 3: Open the Element Lists Dialog, double-click the fields for Entity/Variable/Initial Value, and change the contents as shown in Figure 4.6.

4.2.3 Process

A process is an "event" which occurs to entities such as proteins. A process could represent a variety of events such as activation, binding, expression, transportation, transcription, and translation. By combining entities with these processes, we can create models so that the system behaviors of complex events and reactions can be observed.

As in the case for entity, three *types* of processes are defined: *discrete process*, *continuous process*, and *generic process* . Discrete process, continuous process, and generic process are represented with the symbols in Figure 4.7. This book will cover discrete and continuous processes and skip generic processes.



0. Select the element

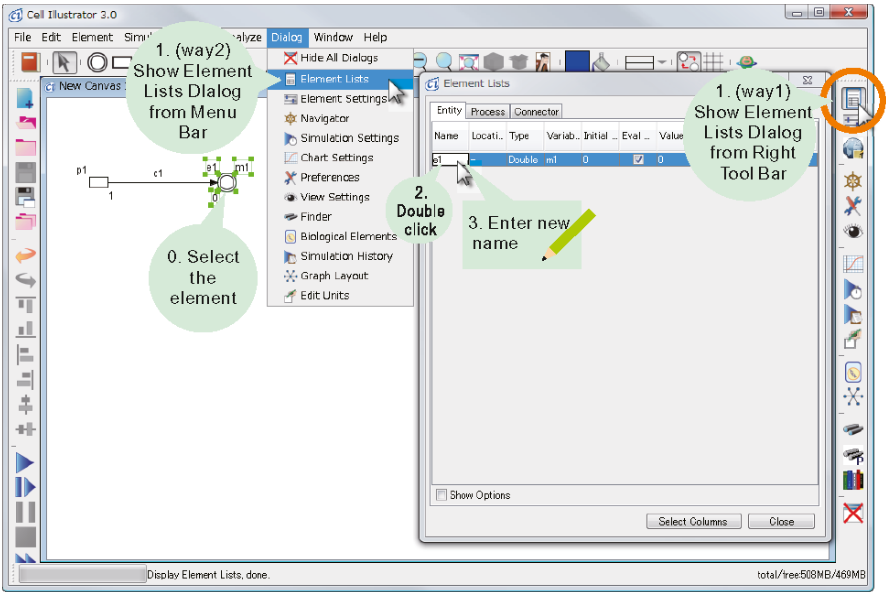
1. (way2) Show Element Settings Dialog from Menu Bar

1. (way1) Show Element Settings Dialog from Right Tool Bar

2. Enter new name

Name e1
Variable m1
CSO Class –
Value Type
Physical Property
Unit
Initial Value 0
Evaluate Script Once on
Maximum Value infinite
Minimum Value 0
Public off
Save Log off
Show Element on
Show Name on
Show Variable on
Show Value on
References Select/Add/Edit Reference
Comments

Fig. 4.5



0. Select the element

1. (way2) Show Element Lists Dialog from Menu Bar

1. (way1) Show Element Lists Dialog from Right Tool Bar

2. Double click

3. Enter new name

Name	Locati..	Type	Variab..	Initial ...	Eval ...	Value
e1		Double	m1	0	☑	0

Display Element Lists, done.

Fig. 4.6

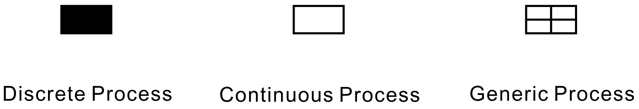


Fig. 4.7

Placing processes onto the canvas in Cell Illustrator, the three properties of a process are shown in the top left, bottom left, and bottom right as seen in the left of Figure 4.8. These three fields are called *Name*, *Delay*, and *Speed* which represent the name of the process, the *delay* of the process (explained later), and the *speed* of the process (explained later), respectively. By default, Name has a ‘p’ followed by a positive integer and Delay and Speed have a value of 1. The delay is involved only with discrete and generic processes, while the speed is involved with all types of processes.

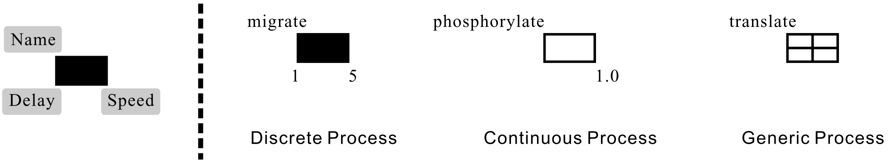


Fig. 4.8

There are three ways to modify the contents of the fields for Name, Delay, and Speed:

Method 1. Double-click on the target text of the process on the canvas and edit directly (Figure 4.9).

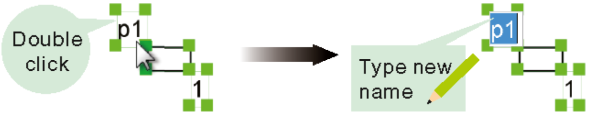


Fig. 4.9

Method 2. Open the Element Settings Dialog and change the contents of the fields for Name, Delay, and Speed.

Method 3. Open the Element Lists Dialog, click the Process tab, and change the contents of the fields for Name, Delay, and Speed.

4.2.4 Connector

A connector is an "arrow" that connects an entity to a process or a process to an entity. Connectors cannot connect entities to entities or processes to processes.

Specifically, a connector going from an entity to a process is called an *input connector*, while a connector going from a process to an entity is called an *output connector*. There are some combinations of entities and processes which cannot be connected according to their types (see Table 4.1).

Three *types* of connectors are defined: *process connector*, *inhibitory connector*, and *association connector*. These three types have symbols shown in Figure 4.10. Any of these connectors can act as an input connector, but only the process connector can act as an output connector.

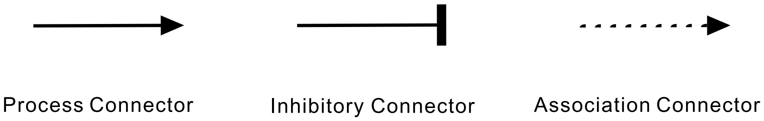


Fig. 4.10

Process connectors are the most versatile and can be used in any reaction. Inhibitory connectors are used to show an inhibition from an entity to a process. An association connector is used to denote an entity assisting a process. The specific uses of each connector will be covered in detail in Section 4.7.

After connecting a process and an entity on a canvas, two properties are displayed on top and bottom, as shown in Figure 4.11. These two fields are called *Name* and *Threshold* which represent the name of the connector and the threshold of the connector (explained later), respectively. By default, Name has a 'c' followed by a positive integer and Threshold has a value of 0.

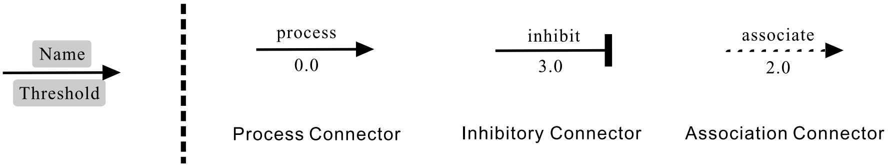


Fig. 4.11

There are three ways to modify the contents of the fields for Name and Threshold:

Method 1. Double-click on the target text of the connector on the canvas and edit directly (Figure 4.12).

Method 2. Open the Element Settings Dialog and change the contents of the fields for Name and Threshold.

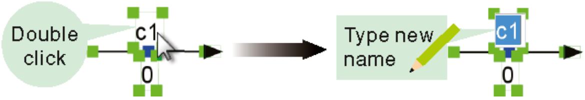


Fig. 4.12

Method 3. Open the Element Lists Dialog, click the Connector tab, and change the contents of the fields for Name and Threshold.

4.2.5 *Rules for Connecting Elements*

Cell Illustrator defined the rules for connecting entities and processes with connectors. As mentioned in Section 4.2.4, processes cannot be connected to processes and entities cannot be connected to entities. Connectors connect processes and entities, with limitations depending on the types of processes, entities, and connectors as shown in Table 4.1. The rules are summarized as follows:

1) Inhibitory connectors and association connectors can only be used as input connectors and not as output connectors. These connector types have no other restrictions.
2) Process connectors cannot connect a continuous process to a discrete entity and vice versa.
3) Process connectors can only connect generic entities to generic processes.

When attempting to connect elements on the canvas, a valid connection changes the element outline green (Figure 4.13), while an invalid connection makes no changes on the target (Figure 4.14). In this way, it is easy to recognize invalid connections without remembering the rules on the canvas.

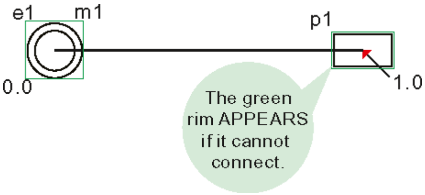


Fig. 4.13

Table 4.1 Connection rules between entities and processes.

connector type		input connector (process connector)		
process type		discrete	continuous	generic
entity type	discrete	√	–	√
	continuous	√	√	√
	continuous	–	–	√

connector type		input connector (inhibitory/association connector)		
process type		discrete	continuous	generic
entity type	discrete	√	√	√
	continuous	√	√	√
	continuous	√	√	√

connector type		output connector (process connector)		
process type		discrete	continuous	generic
entity type	discrete	√	–	√
	continuous	√	√	√
	continuous	√	√	√

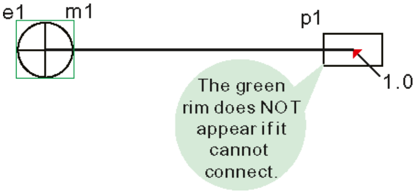
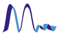


Fig. 4.14

4.2.6 Icons for Elements

Cell Illustrator has illustrated elements (icons for elements) in addition to the aforementioned ones (explained further in Section 4.6). If these are not enough, users can employ their custom-made illustrations for entities and processes. To change the illustration of an element, first select the element and “right-click” → [Replace Figure] → [Image] to paste the custom-made illustration. Using these features can make pathway models far more readable. These custom elements have the same variations and properties as their basic counterparts.

4.2.6.1 Examples of Entity Icons

Protein		mRNA	DNA
			

4.2.6.2 Examples of Process Icons

Activation	Cleavage	Translocation
Transcription	Translation	Degradation
Phosphorylation	Dephosphorylation	Binding

4.3 Editing a Model on Cell Illustrator

After starting Cell Illustrator, the *main window* will be displayed (Figure 4.15). The sections of the menu have names as given in Figure 4.15.

Creating or editing a new model requires a new canvas. There are two ways to create a new canvas:

1) Click the icon on the top of the left tool bar. By hovering the mouse cursor above the icon, *Create New Canvas* pops up.
2) Select [File] → [New] in the menu bar. Note that selecting [File] → [Close File] or [Save] closes or saves the current canvas.

4.3.1 Adding Elements

Recall that there are three types of elements: entities, processes, and connectors. Here are ways to add those elements to the canvas.

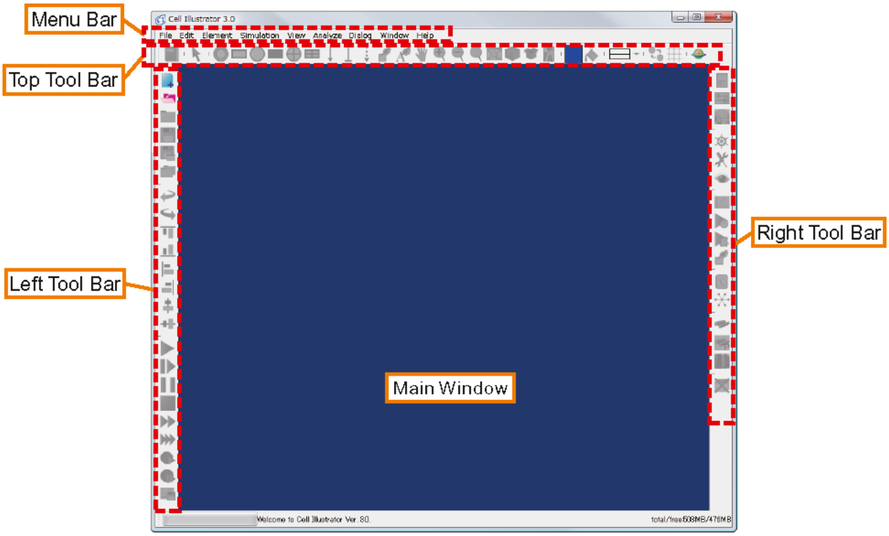

Menu Bar
Top Tool Bar
Left Tool Bar
Right Tool Bar
Main Window
Cell Illustrator 3.0
File Edit Element Simulation View Analyze Dialog Window Help
Welcome to Cell Illustrator Ver. 80.

Fig. 4.15

4.3.1.1 Adding Entities and Processes

There are four ways to add processes and entities onto the canvas:

1) From the top tool bar, select any of the icons ◎, ○, ⊕, ▭, ■, ⊞ which represent entities and processes, and click on the location to place the element.
2) Without selecting any elements, right-click the canvas and select *Insert Entity/Insert Process* to place a specific element on the canvas (Figure 4.16).

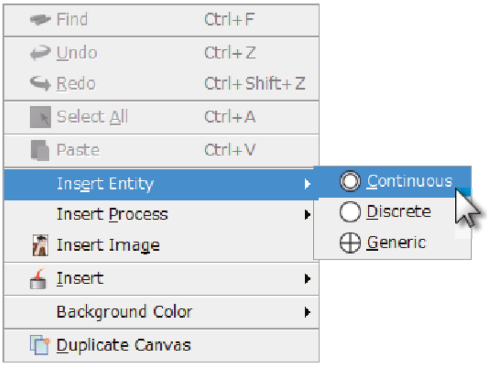


Fig. 4.16

3) Select [Edit] → [Insert Entity] or [Edit] → [Insert Process] from the menu bar to place the selected element (Figure 4.17). Elements added in this way will be placed in the center of the canvas.

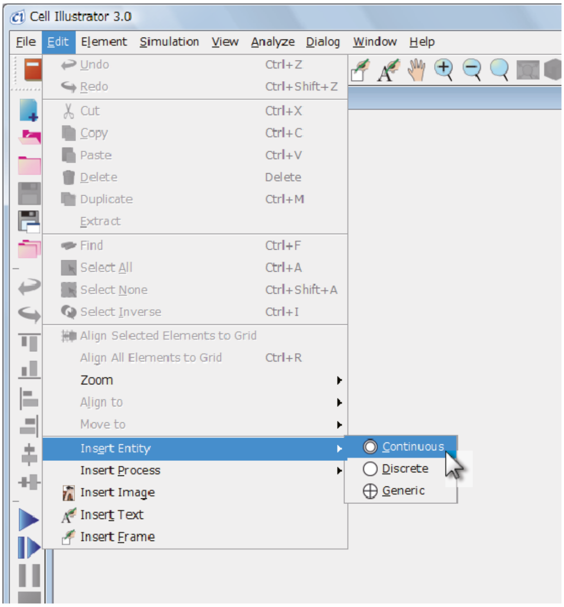


Fig. 4.17

4) Elements can be dragged and dropped from the Biological Elements Dialog, which will be explained in detail further on.

4.3.1.2 Adding Connectors

Entities and processes are connected in the following way:

1) From the top tool bar, click on any of the icons ↓ , ⊥ , ⇣ which represent connectors (Figure 4.18).
2) Select the source element from which the connector begins (Figure 4.19) and select the destination element at which the connector ends (Figure 4.20). There are some conditions on the combination of elements that can be connected (refer to Table 4.1).

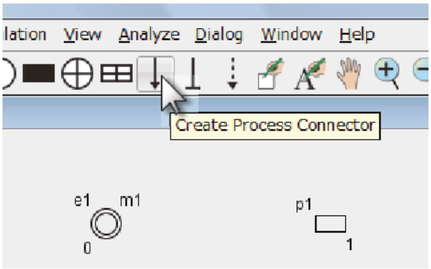


Fig. 4.18

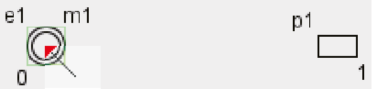


Fig. 4.19

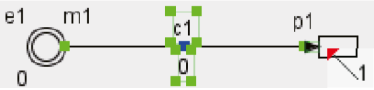


Fig. 4.20

4.3.2 *Model Editing and Canvas Controls*

Here are the basic controls for editing the elements on a canvas.

4.3.2.1 Element Editing

Select the ↖ at the top tool bar to enter *element selection mode*.

4.3.2.2 Selecting and Moving Elements

Elements can be moved in element selection mode by clicking and dragging. Alternatively, elements can be moved using Shift and the mouse pointer. Multiple elements can be selected by dragging the mouse pointer around elements, or holding Ctrl while selecting individual elements.

4.3.2.3 Cut, Copy, and Paste

Select the element with element selection mode. Then *Cut*, *Copy*, and *Paste* can be done in the following way:

1) Right-click and select [Cut], [Copy], or [Paste].
2) Select [Edit] → [Cut], [Copy], or [Paste] in the menu bar.
3) Press Ctrl+x, Ctrl+c, or Ctrl+v on the keyboard.

After pasting, the connectors can be bent back into shape by grabbing the small black square ■ (under the label ‘c1’ in Figure 4.20) in the center of the connector and moving the vertex.

4.3.2.4 Other Canvas Operations

Most common operations are placed at the top tool bar:

“Edit Parts” shows the set of ready-made pathway parts.

“Create Frame” creates a frame for the canvas.

“Create Note” adds comments to the canvas.

“Manual Move” pans the canvas when the canvas exceeds screen size. Similar commands are available in the *Navigator Dialog*.

“Zoom In” zooms in and displays the canvas.

“Zoom Out” zooms out and displays the canvas.

“Reset Zoom” returns the canvas to its original magnification.

“Fit Selection to Canvas Size” changes the magnification and pans to fit all selected elements. If no elements are selected, then the canvas size is made to fit all elements.

“Group” treats all selected elements as one group. If one element of a group is selected, the rest of the group is automatically selected.

“Ungroup” returns a grouped element to its previous state.

“Load Image” loads and adds an image to the canvas.

“Set Color” changes the color for the property of the element selected by the icons of “Select Color Tool”.

“Select Color Tool” determines which part of the element is colored by the “Set Color” operation. 🪣 specifies the inside of the element, ✏ specifies the outer rim of the element, and A specifies the color of the text for the element.

“Set Stroke” sets the line width for the selected element.

“Toggle Grid Visible Status” toggles the visibility of the canvas grid.

“Toggle Antialiasing Status” changes the rendering method of the canvas. When antialiasing is activated, the models look smoother.

"Go To BioPACS" imports the KEGG metabolic pathways using BioPACS. BioPACS is a tool that automatically converts KEGG metabolic pathways for use by Cell Illustrator.

4.4 Simulating Models

After a model is built, the next step is to simulate this model. Here are the minimal settings required for simulation.

4.4.1 Simulation Settings

The various settings for simulation are accessed through the Simulation Settings Dialog (Figure 4.21). This dialog is displayed by clicking the icon on the right tool bar.

This dialog has the following eight fields:

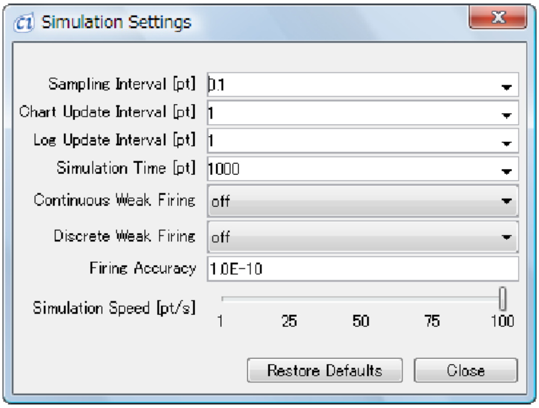
1) Sampling Interval
2) Chart Update Interval
3) Log Update Interval
4) Simulation Time
5) Continuous Weak Firing
6) Discrete Weak Firing
7) Firing Accuracy
8) Simulation Speed

Normally, the field of Simulation Time, which defines the time until the simulation stops, is required. For example, if this field is set to 100[pt], then the simulation will start at 0[pt] and stop at 100[pt]. Here, [pt] means *Petri net time* and is functionally equivalent to a unit of time such as the second. The default value is 1000[pt]. For the exercises in this book, however, 100[pt] is more than enough.

4.4.2 Graph Settings

Once the execution of simulation begins, each entity begins to vary according to the simulation state. Cell Illustrator can plot this change onto a graph. Although there are a few ways to create a graph, we show the simplest one:

Step 1: Select the entity to be shown on the graph. More than one entity can be selected (Figure 4.22).



Simulation Settings

Sampling Interval [pt] 0.1

Chart Update Interval [pt] 1

Log Update Interval [pt] 1

Simulation Time [pt] 1000

Continuous Weak Firing off

Discrete Weak Firing off

Firing Accuracy 1.0E-10

Simulation Speed [pt/s] 100

Fig. 4.21

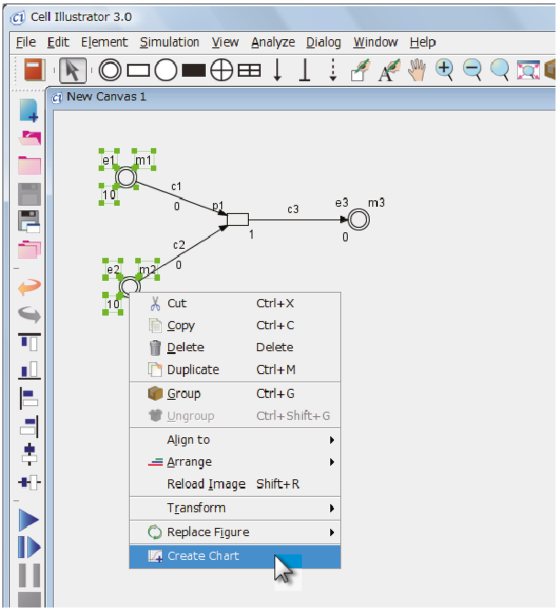
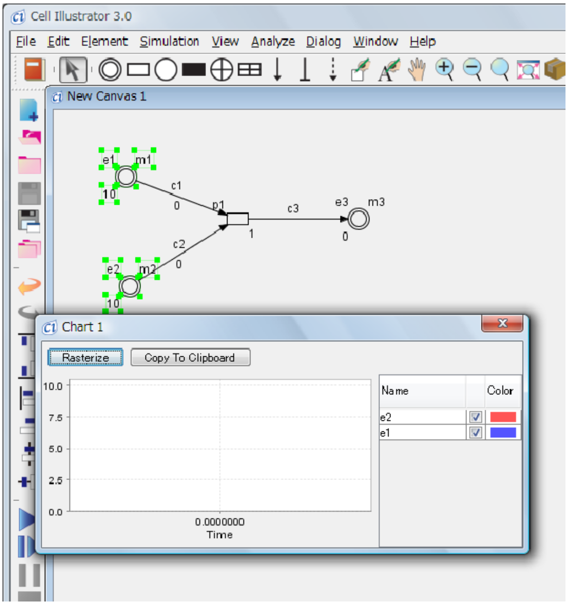


Fig. 4.22

Step 2: Right-click and select [Create Chart] on the canvas (Figure 4.22). This will result in a chart as seen in Figure 4.23.

Charts cannot be deleted in this method. Editing and deleting graphs are done through the Chart Settings Dialog (accessed through 📈 on the right tool bar).



Name		Color
e2	☑	🟥
e1	☑	🟦

Fig. 4.23

4.4.3 *Executing Simulation*

After setting the parameters for the simulation and graph (although it is not always necessary to set graph parameters), it is time for evaluating the model by executing simulation. There are a few types of buttons for different types of execution. ▶ is for standard speed execution. ● is for execution with animation. ⏩ is for fast forward execution. ▶▶▶is for high-speed execution. ⏯ is for one-step execution, and ● is for one-step execution with animation. Here, the normal speed sets 1[pt] to one second, and the high-speed execution depends on the processing power of the host computer.

Before execution, the canvas must be saved by selecting either “Save Current Canvas” 💾 or “Save Canvas To Selected File” 💾 . Attempting to execute by ▶ before saving will bring up a dialog prompting the user to save. Note that the canvas cannot be edited while the model is being executed; in order to stop the simulation, as you like (firstly) need to press ■ .

4.5 Simulation Parameters and Rules

As explained before, one can create a model by placing entities and processes on a canvas and by linking them with connectors. This section describes the basic parameters and functions of each basic element.

4.5.1 Creating a Model with Discrete Entity and Process

Create the model shown in Figure 4.24 in the following way:

Step 1: Place three discrete entities e1, e2, and e3 on the canvas.
Step 2: Place one discrete process p1 on the canvas.
Step 3: Draw two process connectors c1 and c2 from e1 and e2 on the left to p1 in the center.
Step 4: Draw a process connector c3 from p1 to e3.
Step 5: Select discrete entities e1, e2, and e3 and create a graph (using [Create Chart]).

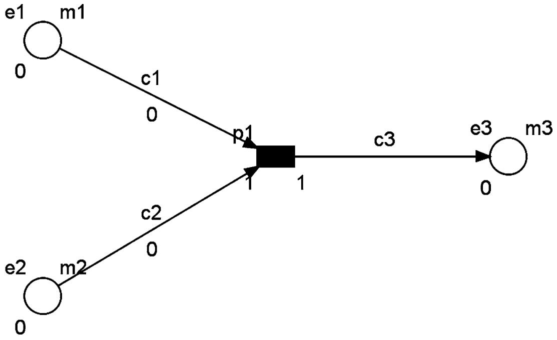
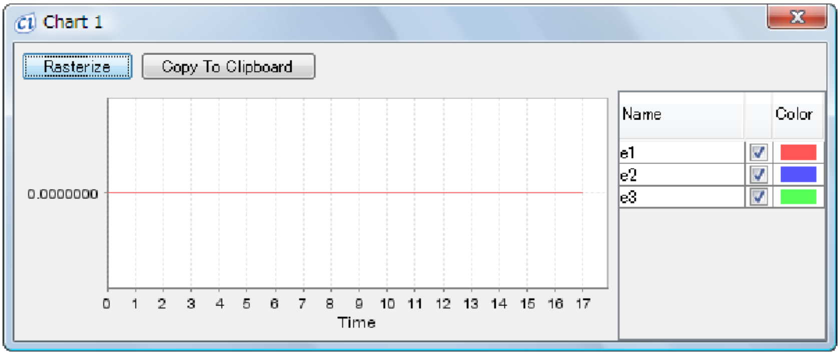


Fig. 4.24

Once the model is done, save and press the execution button ▶ (or ●) to execute the simulation. However, as seen in Figure 4.25 the values of the entities will remain at 0.

4.5.1.1 Initial Value

To create nonzero response, the *initial values* for the entities e1 and e2 will be changed to 10 as seen in Figure 4.26. If the simulation is still being executed, first stop the simulation using ■ and then enter 10 into the field of Initial Value. Now



Name		Color
e1	☑	🟥
e2	☑	🟦
e3	☑	🟩

Fig. 4.25

press the execution button ▶ (or ●) again. Then it displays a graph like Figure 4.27. The entities e1 and e2 will drop to zero and e3 will go to 10 but no further changes will occur.

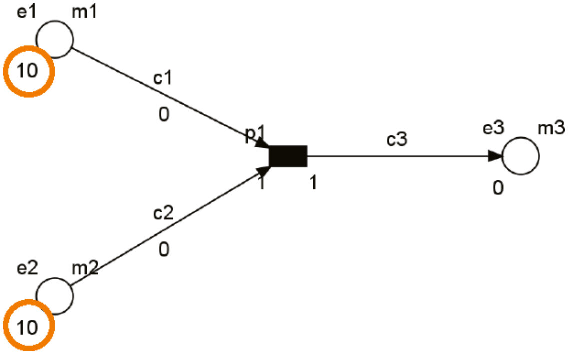
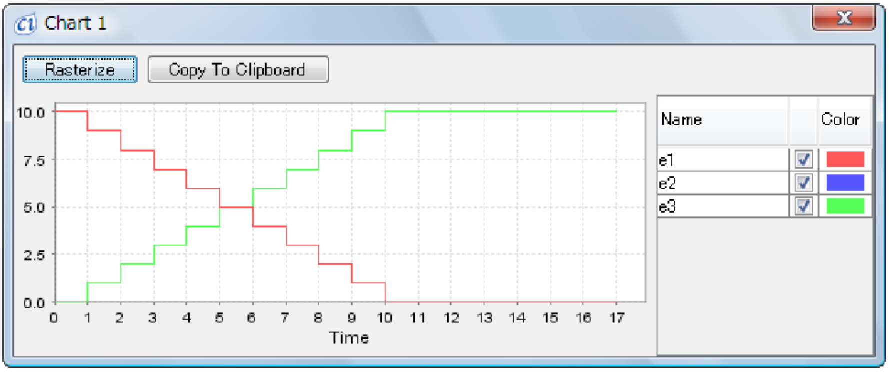


Fig. 4.26

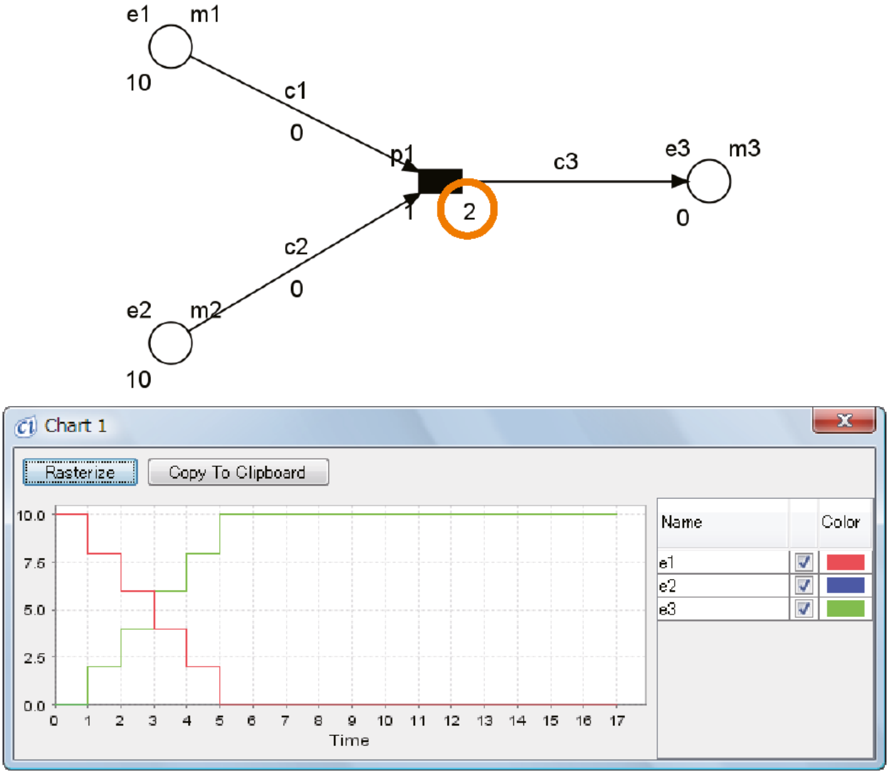
4.5.1.2 Speed

Figure 4.27 shows that the values for e1 and e2 become 0 at time 10[pt]. Change the *speed* of process p1 from 1 to 2, execute, and look at the graph. Now as in Figure 4.28, e1 and e2 become 0 and e3 became 10 at time 5[pt]. As seen here, the speed determines the rate at which e1 and e2 decrease and e3 increases.



Name		Color
e1	☑	🟥
e2	☑	🟦
e3	☑	🟩

Fig. 4.27



Name		Color
e1	☑	🟥
e2	☑	🟦
e3	☑	🟩

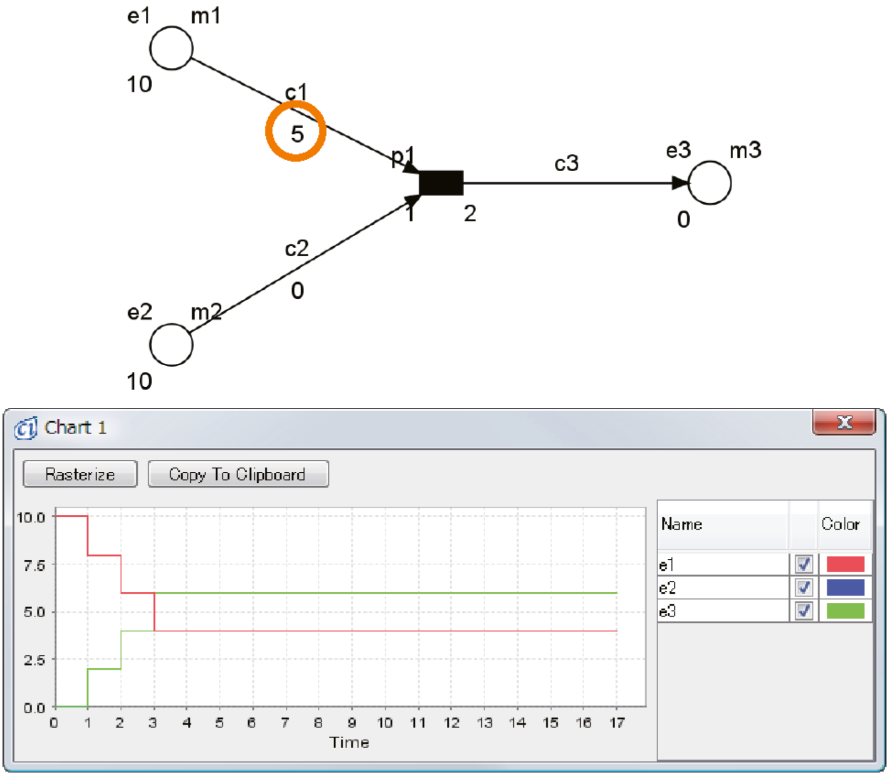
Fig. 4.28

4.5.1.3 Threshold and Conditions for Process Execution

Process p1 is connected to entities e1 and e2 with input connectors c1 and c2. The *threshold* for both these connectors are set to 0. Now set the threshold for connector c1 to 5 and reexecute the model. As seen in Figure 4.29, the entities do not change after 3[pt]. This is because p1 was not *executable* (this notion is called *firable* in Petri net theory). In order to execute, the following two conditions must be satisfied:

1) The entities e1 and e2 which are connected to p1 must have values greater than the thresholds of c1 and c2, respectively.
2) Entities e1 and e2 have values greater than or equal to the speed of p1.

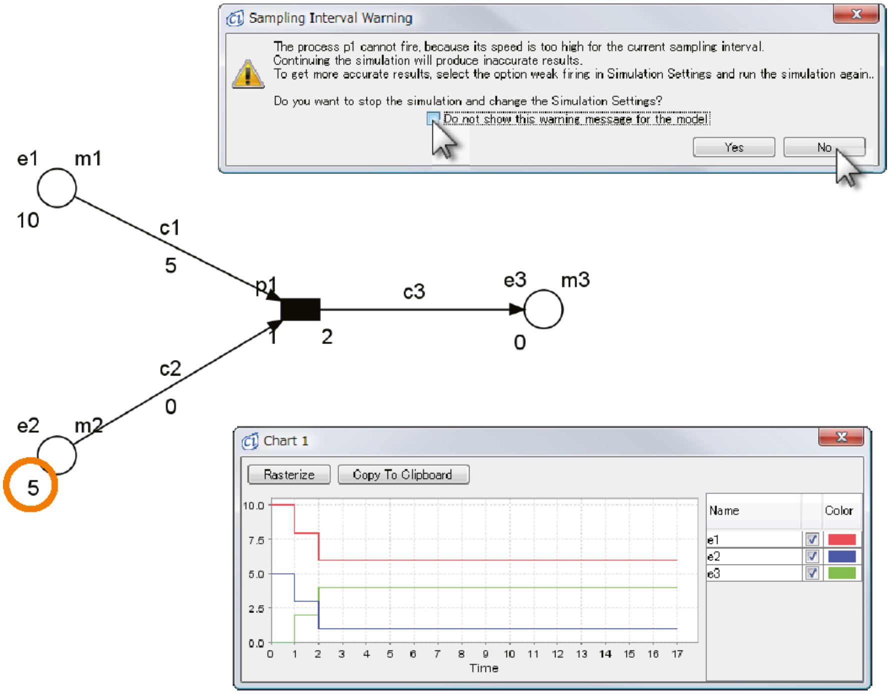
At time 3[pt], entities e1 and e2 have value 4. The value of entity e2 is higher than the threshold value of connector c2(0), but the value of entity e1 is less than the threshold value of connector c1(5). Therefore p1 is not executable and the entities connected to process p1 do not change.



Name		Color
e1	☑	🟥
e2	☑	🟦
e3	☑	🟩

Fig. 4.29

Next, set the initial value of e2 to 5 and reexecute the simulation. Then as shown in Figure 4.30, the model will cease execution at 2[pt] with “Warning”. Although the value of e1 is 6 and greater than c1 threshold of 5, e2 has a value of 1 which is less than the process speed of 2. This causes the simulation to halt even though e2 fulfills the first condition of being greater than the threshold. If "□Do not show this warning message for the model" is checked and "No" is pressed, this warning will disappear and the simulation will continue. This behavior is the same for 3-input connectors, while in the case that there are no input connectors, there is no chance of halting. The threshold does not necessarily have to be an integer. Any nonnegative real number can be selected.



The process p1 cannot fire, because its speed is too high for the current sampling interval.
Continuing the simulation will produce inaccurate results.
To get more accurate results, select the option weak firing in Simulation Settings and run the simulation again..

Do you want to stop the simulation and change the Simulation Settings?

□ Do not show this warning message for the model

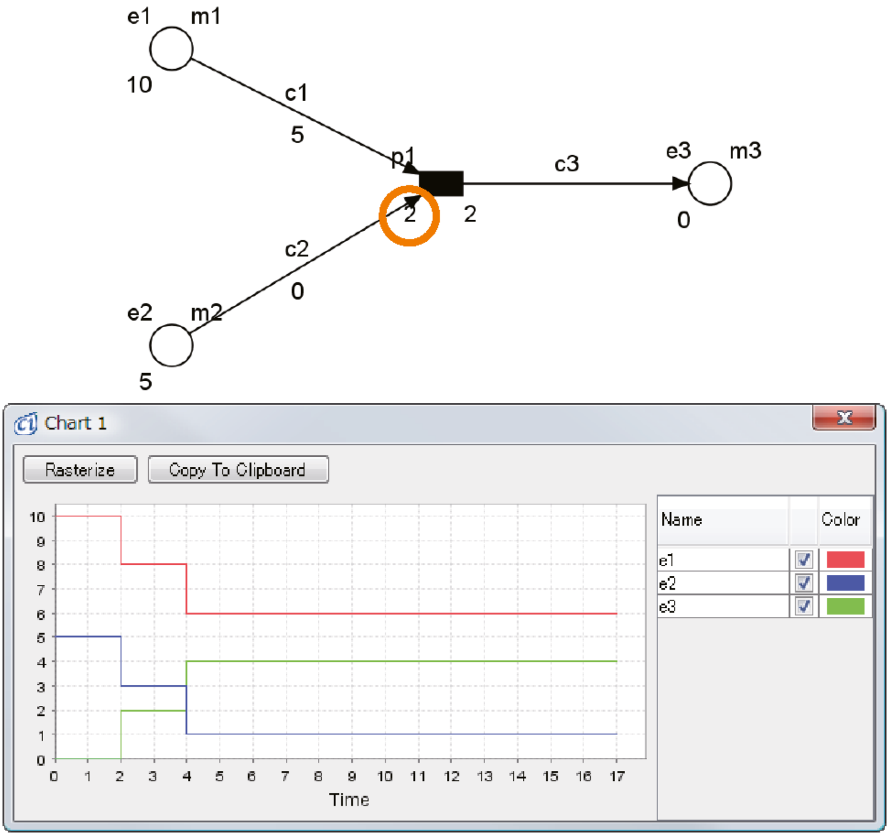
Name		Color
e1	☑	🟥
e2	☑	🟦
e3	☑	🟩

Fig. 4.30

4.5.1.4 Delay

In the same model, change the delay of p1 from 1 to 2 and reexecute the model. The graph will show a similar behavior in Figure 4.31. While there is a decrease of 2 for every 1[pt] in Figure 4.30, there is a decrease of 2 for every 2[pt] in Figure 4.31. In Figure 4.31, e1 and e2 have values of 8 and 3 at 2[pt], respectively, and p1 is executable. After 2[pt], the value decreases again at 4[pt] where the values decrease by the process speed of 2 and e1 and e2 have values of 6 and 1, respectively. This is because process p1 is executed after a delay of 2[pt] once p1 becomes executable.

As seen in this example, the delay is the time between the time when it becomes executable and the time when it is executed. Once the process reaches 4[pt], the process is no longer executable and therefore stays the same. In Figure 4.30, the delay is set to 1. Therefore the process begins to be executed 1[pt] after it becomes executable, and it is executed and the value of the entity is decreased by 2 at every 1[pt].



Name		Color
e1	☑	🟥
e2	☑	🟦
e3	☑	🟩

Fig. 4.31

4.5.2 *Creating a Model with Continuous Entity and Process*

Create the model shown in Figure 4.32 in the following way:

Step 1: Place three continuous entities e1, e2, and e3 on the canvas.
Step 2: Place one continuous process p1 on the canvas.
Step 3: Draw two process connectors c1 and c2 from e1 and e2 to p1.

Step 4: Draw a process connector from p1 to the continuous entity e3.

Step 5: Select all continuous entities e1, e2, and e3 and create a graph (using [Create Chart]).

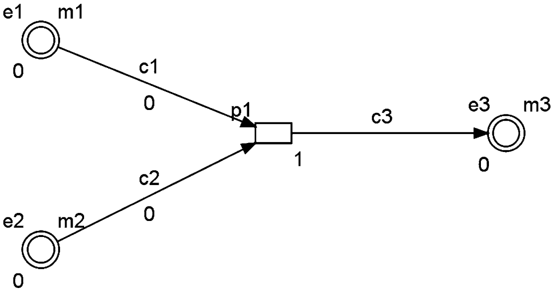


Fig. 4.32

4.5.2.1 Initial Value, Speed, and Threshold

While discrete entities can only hold integer values, continuous entities can hold any nonnegative real numbers. Similarly, *initial value*s for continuous entities can be any nonnegative real numbers. Here, set the initial values for e1 and e2 to 10, and set the *threshold*s for connectors c1 and c2 to 5 and 0, respectively. Although discrete processes have the speed and delay properties, continuous processes have only the property of *speed*. In Figure 4.33, the speed of p1 is set to 1.

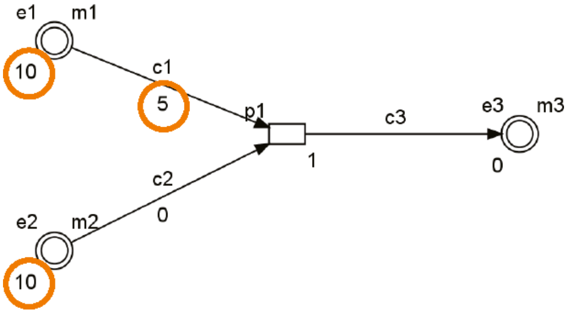
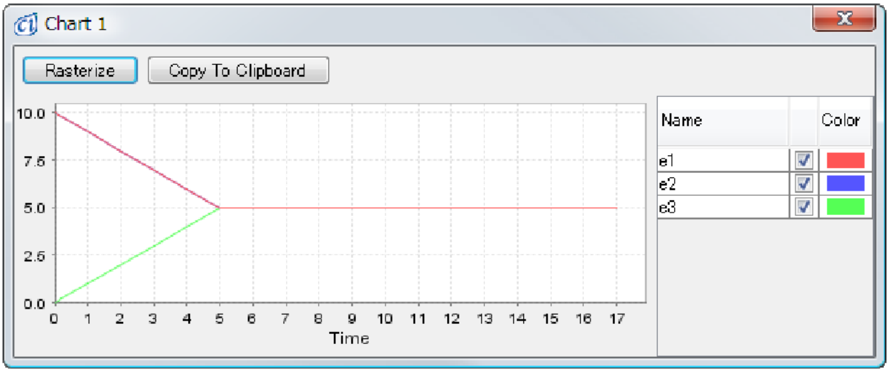


Fig. 4.33

4.5.2.2 Conditions for Process Execution

A continuous process is *executable* if all entities connected to its input connectors have values greater than or equal to their thresholds. As long as the process is executable, the process is *executed* and this *execution* will decrease the value of each entity at some rate while increasing the entity on the output connector at some rate. Although Figure 4.33 has only one output connector, the same increase applies for two or more connectors. By executing the model shown in Figure 4.33, the resulting graph should look like Figure 4.34. The values of entities e1 and e2 will be decreased from the initial value at speed 1, and the value of e3 is increased at speed 1 until e1 and e2 reach the value 5 at which point the process is not executable. Since a discrete process has a delay between "becoming executable" and "execution", the graph is step-like as shown in Figure 4.29.



Name		Color
e1	☑	🟥
e2	☑	🟦
e3	☑	🟩

Fig. 4.34

4.5.3 Concepts of Discrete and Continuous

Cell Illustrator can describe a hybrid event consisting of discrete and continuous elements by using entities and processes.

Discrete representation $(0,1,2,\ldots)$ is suited for events which can be described with integers. For example, a digital clock which changes value every minute is a discrete event. Biologically, molecular quantity is often represented using discrete entities. For example, discrete entities can be used to describe an mRNA which functions in few copy numbers in a cell. As other usage, they can be used to model the switching regulation in a cell, e.g. on/off regulation by transcription factor.

On the other hand, continuous is well suited for concentrations and values which are continuously changing. In the case of ATP and other molecules produced in mass, it is more reasonable to treat the whole unit as a concentration. If “discrete” is a digital clock, then “continuous” is an analog clock.

The graphical differences for discrete and continuous representations are shown in Figure 4.35. The discrete graph, which uses only integers, looks step-like, while the continuous graph shows a smooth line.

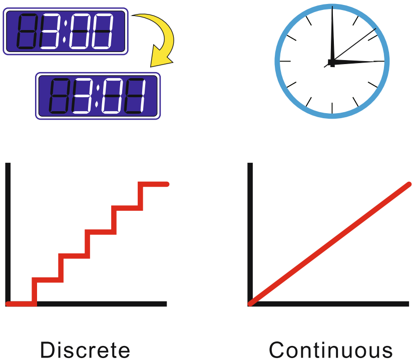


Fig. 4.35

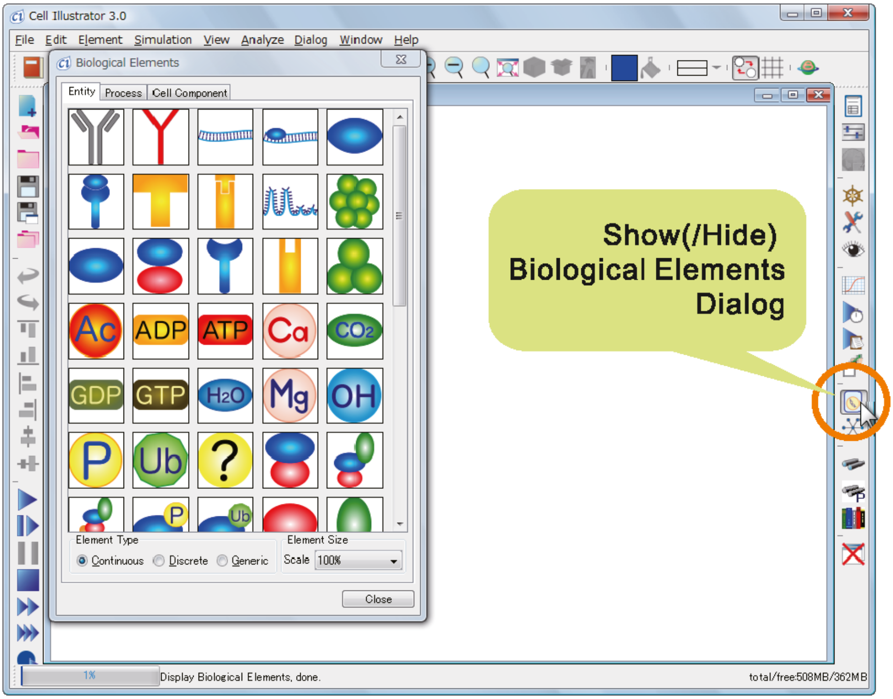
4.6 Pathway Modeling Using Illustrated Elements

As explained in Section 4.2.6, Cell Illustrator has illustrated *biological entities* such as mRNAs and proteins and *biological processes* such as phosphorylation. These illustrated elements are called *biological elements*. These biological elements allow us to create pathway models more easily in the following way:

Step 1: Click the icon on the right tool bar. Then the Biological Elements Dialog will be shown as in Figure 4.36.

Step 2: In this dialog there are three tabs: Entity, Process, and *Cell Component*, as seen in Figure 4.37. Under the element listing is a selection of *element type*: continuous, discrete, or generic. The cell components are used to specify the locations of entities and processes in the cell. By placing them at the appropriate locations on the selected cell component, the model will be enhanced by knowledge such as subcellular localization information.

Step 3: Add these elements from the dialog by dragging and dropping as in Figure 4.38.
Dropping an illustrated element on an existing element updates its properties (such as graphics and ontology). This can be used to speed up the process of updating a model.


Cell Illustrator 3.0
File Edit Element Simulation View Analyze Dialog Window Help
Biological Elements
Entity Process Cell Component
Ac ADP ATP Ca CO2
GDP GTP H2O Mg OH
P Ub ?
Close
Show(/Hide) Biological Elements Dialog

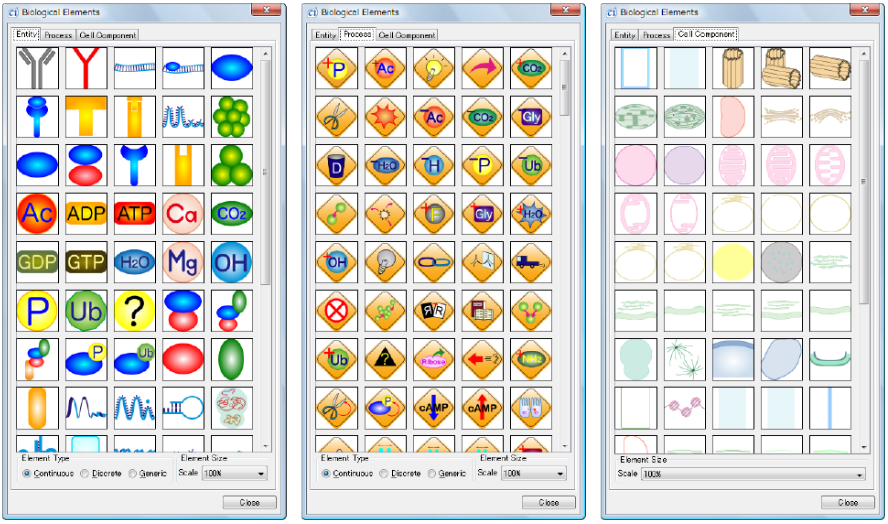
Element Type ● Continuous ○ Discrete ○ Generic

Element Size Scale 100%

1% Display Biological Elements, done.

total/free:508MB/362MB

Fig. 4.36

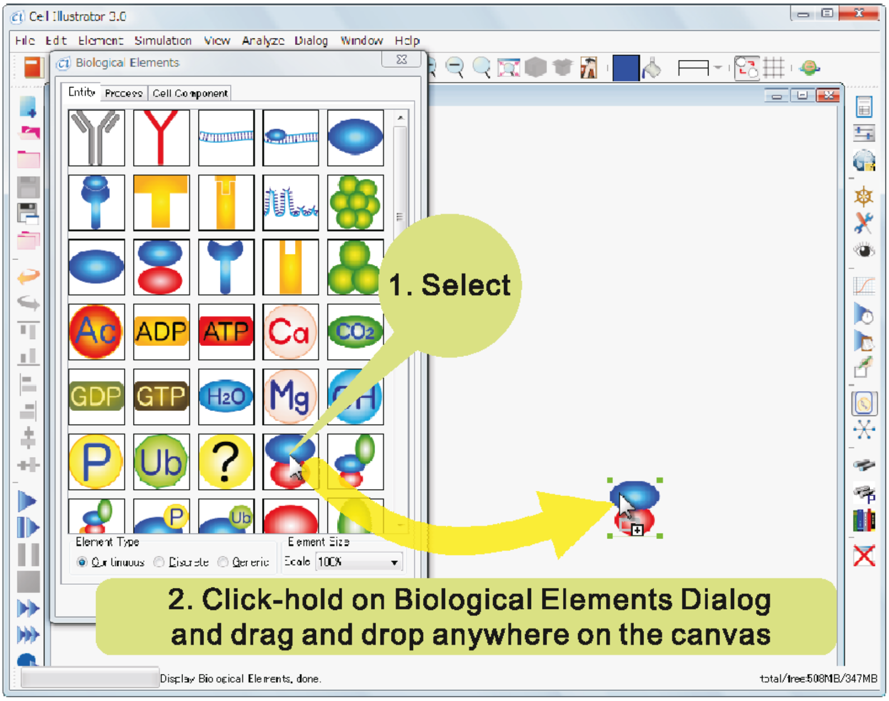


Element Type ● Continuous ○ Discrete ○ Generic

Element Size Scale 100%

Element Size Scale 100%

Fig. 4.37



Element Type: ● Continuous ○ Discrete ○ Generic

Element Size: Scale 100%

1. Select

2. Click-hold on Biological Elements Dialog and drag and drop anywhere on the canvas

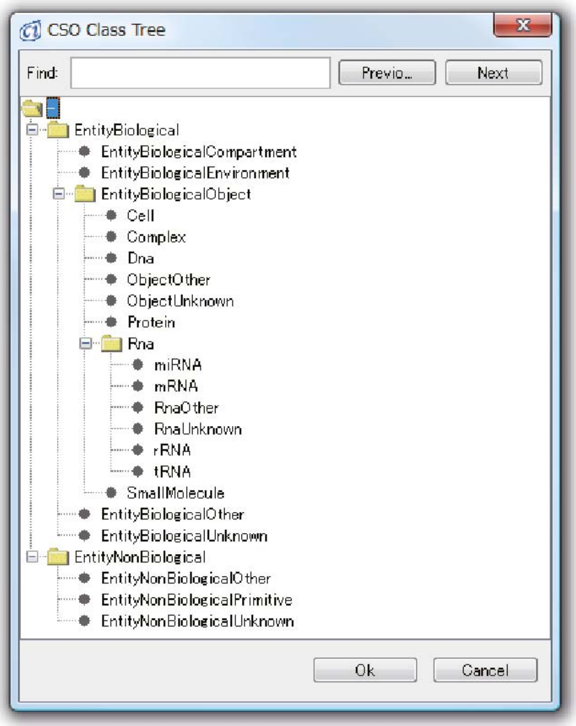
Display Biological Elements, done.

Fig. 4.38

COLUMN 3

Ontology and Illustrated Elements

In Section 4.6, the illustrated elements were explained simply as elements with pictures. However, actually, every illustrated element has a predefined set of ontologies. Cell Illustrator makes use of an ontology called CSO (Cell System Ontology) which strictly defines the relationship between biological terms. Cell Illustrator transparently implements this so that the user creates a pathway with the ability to distribute, reuse, and validate, with nearly no work.



CSO Class Tree

Find: Previo... Next

```
EntityBiological
    EntityBiologicalCompartment
    EntityBiologicalEnvironment
    EntityBiologicalObject
        Cell
        Complex
        Dna
        ObjectOther
        ObjectUnknown
        Protein
        Rna
            miRNA
            mRNA
            RnaOther
            RnaUnknown
            rRNA
            tRNA
        SmallMolecule
    EntityBiologicalOther
    EntityBiologicalUnknown
EntityNonBiological
    EntityNonBiologicalOther
    EntityNonBiologicalPrimitive
    EntityNonBiologicalUnknown
```

Ok Cancel

4.7 Creating Pathway Models Using Cell Illustrator

In this section, we will model several well-known biological events. These examples will serve as tutorials for modeling biological pathways. For these examples, continuous entities and processes are used unless specified otherwise.

4.7.1 Degradation

Inside the cell, mRNA and protein naturally break down. These reactions are called *degradation*. This type of reaction can be modeled in the following way by using a *degradation process* :

Step 1: Place an entity on the canvas. Set the name and initial value of the entity to some appropriate value such as (name, initial value)=(p53, 100).

Step 2: Place a degradation process on the canvas.

Step 3: Create a process connector ↓ from the entity p53 to the degradation process .

Step 4: Specify the degradation speed by setting the speed on the degradation process D. The default value is set to 1.

This leads to the model in Figure 4.39.

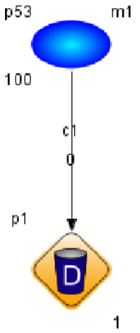
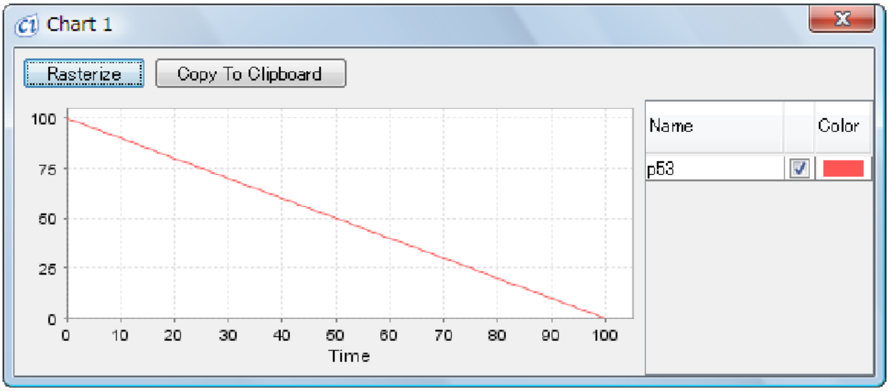


Fig. 4.39

Execute the model and make a graph for p53 ●. The resulting graph should look like Figure 4.40. In the above model, the degradation speed is not dependent on the concentration of p53 ●. In order to make the degradation more realistic by adding p53 ● dependency, change the parameter of the degradation process D to "m1/10" using the variable name (m1) assigned to p53 ● (Figure 4.41). The result should look like Figure 4.42.



Name		Color
p53	☑	🟥

Fig. 4.40

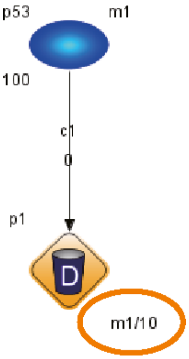
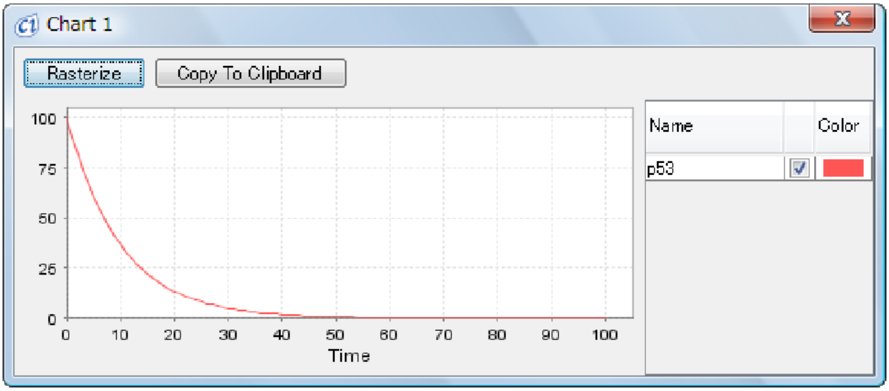


Fig. 4.41



Name		Color
p53	☑	🟥

Fig. 4.42

4.7.2 Translocation

Inside the cell, mRNAs and proteins are transported to and from a variety of places such as from the cytoplasm to the nucleus, or from the nucleus to the Golgi body. This process is called a *translocation*. These actions can be modeled in the following way by using a *translocation process* :

Step 1: Place common cell components by pressing "Edit Parts" and selecting [cells] → [animal_cell_nucleus] (Figure 4.43). Although it is possible to add each cell component from the Biological Elements Dialog, commonly used sets such as these are placed in .

Step 2: Place an entity in the cytoplasm and nucleoplasm and name them "p53_cytoplasm" and "p53_nuclear". In addition, set the initial value. In this example, set cytoplasmic p53 to 100 and nucleoplasmic p53 to 0.

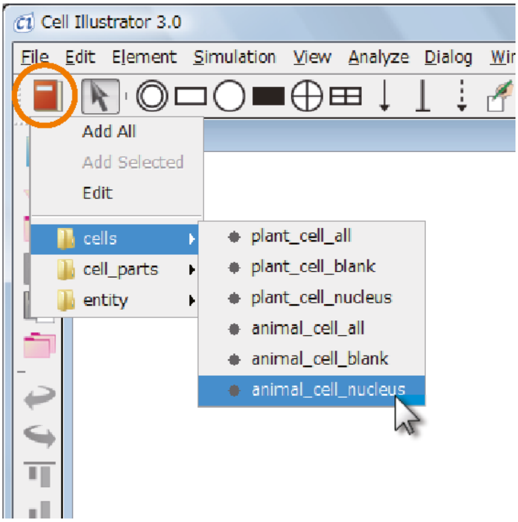


Fig. 4.43

Step 3: Place a process ◆ near the nuclear membrane and name appropriately. For example, a name could be "translocation". Then create a connector c1 from cytoplasmic p53 ⬬ to translocation process ◆ and then another connector c2 from translocation process ◆ to nucleoplasmic p53 ⬬.

Step 4: Set the translocation speed as the process speed. The default value is 1.

This model should look like Figure 4.44. The graphs for cytoplasmic p53 ⬬ and nucleoplasmic p53 ⬬ are shown in Figure 4.45.

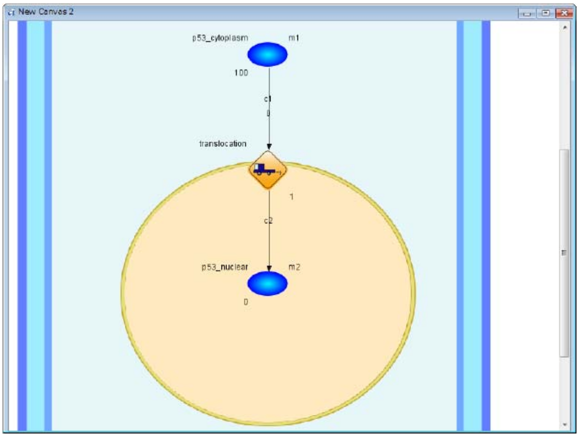
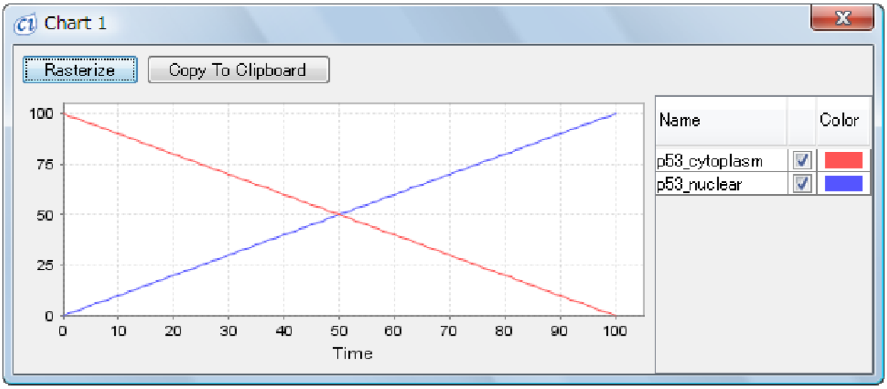


Fig. 4.44



Name		Color
p53_cytoplasm	☑	🟥
p53_nuclear	☑	🟦

Fig. 4.45

If, like the previous example, the translocation speed depends on p53 concentration, then set the process speed to be dependent such as "m1/10" (Figure 4.46). The simulation result is shown in Figure 4.47.

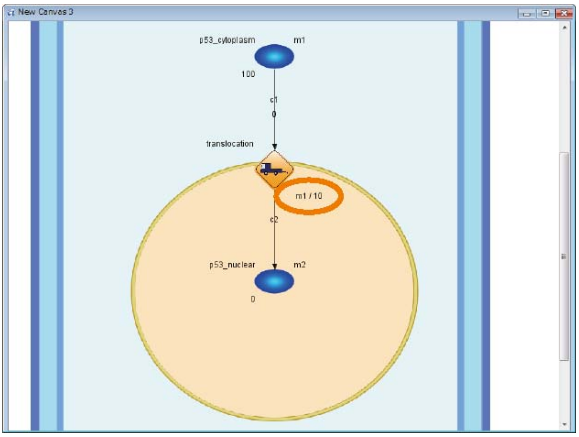
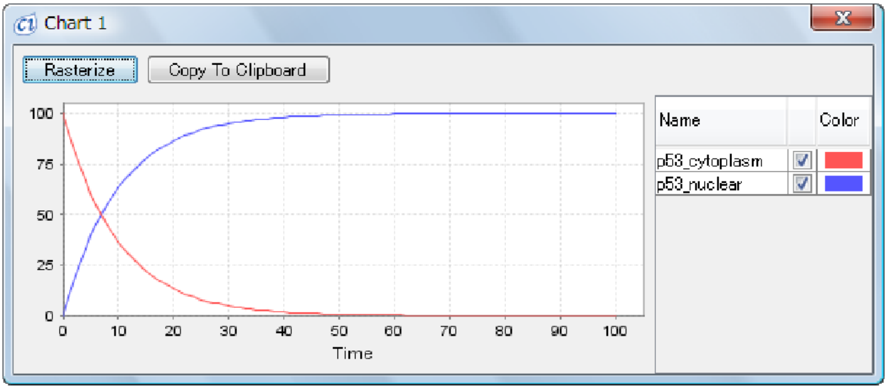


Fig. 4.46

Now, for the types of entities and processes used in Step 1 and Step 2, the connection rules dictate that for (nucleoplasmic p53, process, cytoplasmic p53): (discrete, discrete, discrete), (discrete, discrete, continuous), (continuous, discrete, discrete), (continuous, discrete, continuous), (continuous, continuous, continuous) are possible.



Name		Color
p53_cytoplasm	☑	🟥
p53_nuclear	☑	🟦

Fig. 4.47

4.7.3 Transcription

The *transcription* model for producing mRNA can be modeled in Cell Illustrator with the following steps by using a *transcription process* :

Step 1: Press "Edit Parts" icon on the top tool bar and select [cells] → [animal_cell_nucleus]. Appropriate cell components will be placed on the canvas with this operation.

Step 2: Place the transcription process on the canvas, and set the contents of the field for Name to an appropriate value. For this example, rename it "transcription".

Step 3: Place the entity on the canvas and name it appropriately, such as "mRNA_p53".

Step 4: Draw a process connector from the transcription process to the entity mRNA_p53 .

Step 5: Set the transcription rate by changing the contents of the field for Speed in the transcription process . The default value is 1.

The resulting model should look like Figure 4.48.

The graph for mRNA_p53 by simulation is shown in Figure 4.49.

By the connection rules of entities and process, the entity and process of Steps 1 and 2 can take on the types (transcription, mRNA_p53): (discrete, discrete), (discrete, continuous), (continuous, continuous).

Normally mRNA is degrading at the same time as transcription. The model in Figure 4.50 adds a degradation model to the previous transcription model in Section 4.7.1. The simulation result looks like Figure 4.51.

Although in this model only the final amount of mRNA produced is modeled, the intermediate stages can be modeled using generic entities and processes. This

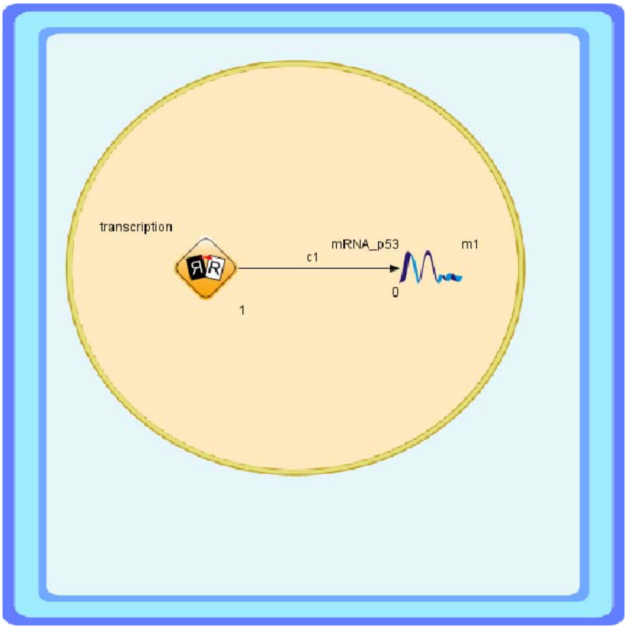
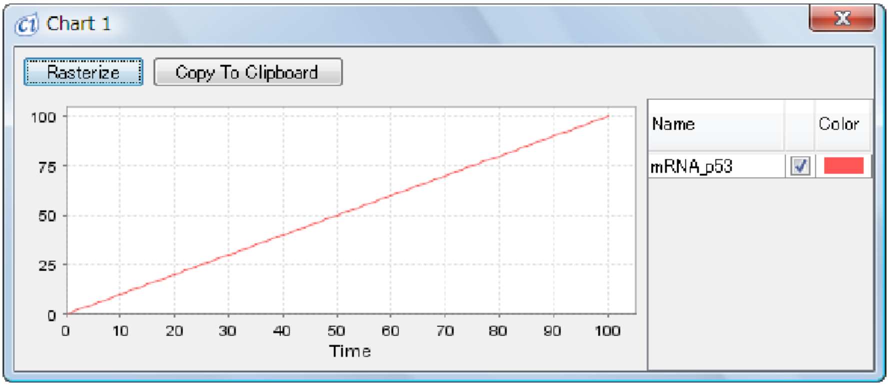


Fig. 4.48



Name		Color
mRNA_p53	☑	🟥

Fig. 4.49

generic model is lightly covered in Column 5 at the end of this chapter, this topic is outside the scope of this introductory textbook.

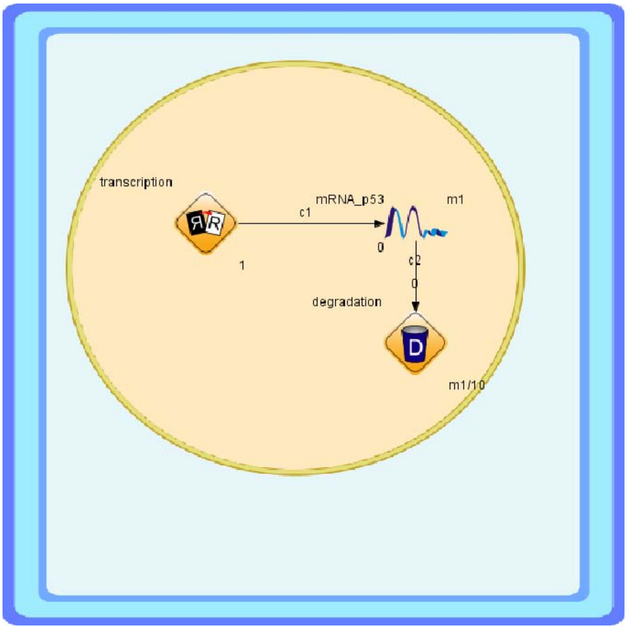


Fig. 4.50

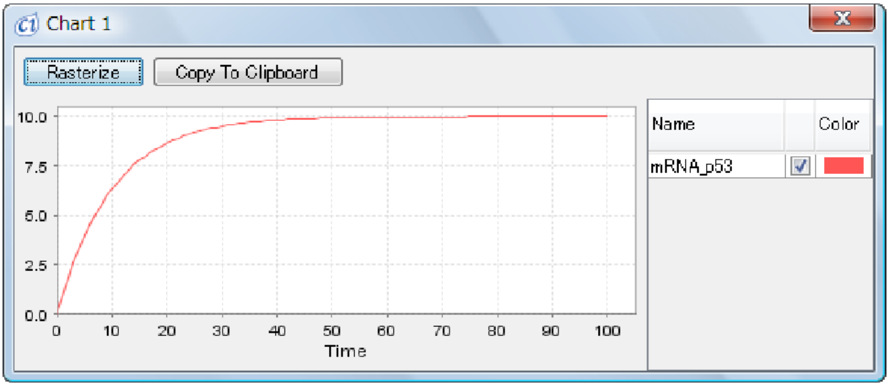


Fig. 4.51

4.7.4 Binding

Inside the cell, multiple proteins often combine to form a complex. This event is called *binding* and can be modeled in the following way by using a *binding process* :

Step 1: Place three entities 🔵, 🔴, 🔵🔴 on the canvas. Set appropriate names and initial values such as (name, initial value)=(p53, 100), (mdm2, 50), (p53_mdm2,0).

Step 2: Place the process ◆ on the canvas. Set an appropriate name such as "binding".

Step 3: Create process connectors from p53 🔵 and mdm2 🔴 to process binding ◆. Then create a connector from process binding ◆ to entity p53_mdm2 🔵🔴.

Step 4: Set the binding rate by changing the speed of the binding process ◆. The default value is 1.

The result is given in Figure 4.52.

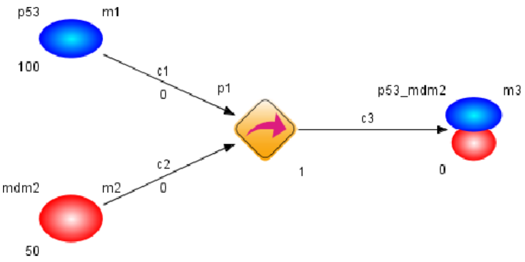
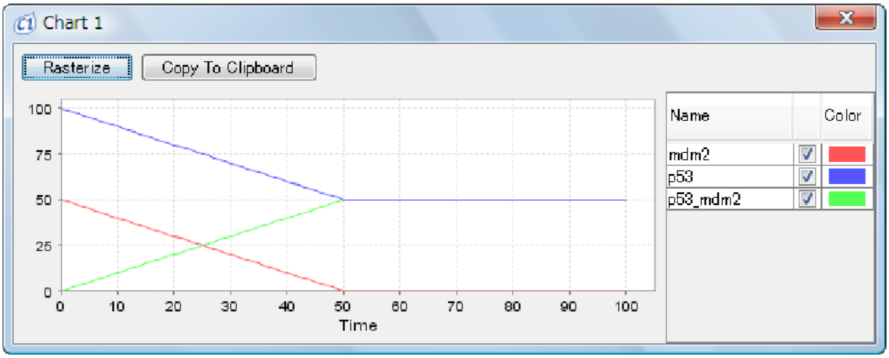


Fig. 4.52

Creating a graph for all entities in this model, the result should look like Figure 4.53.



Name		Color
mdm2	☑	🟥
p53	☑	🟦
p53_mdm2	☑	🟩

Fig. 4.53

If the binding rate relies upon entities p53 🔵 (variable m1) and mdm2 🔴 (variable m2), the process speed should be set to (m1*m2)/300 (300 is an arbitrary constant). In this case, the model looks like Figure 4.54 and the simulation result looks like Figure 4.55.

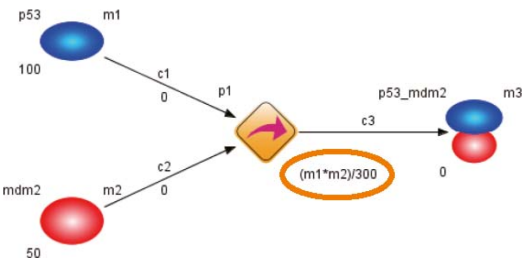
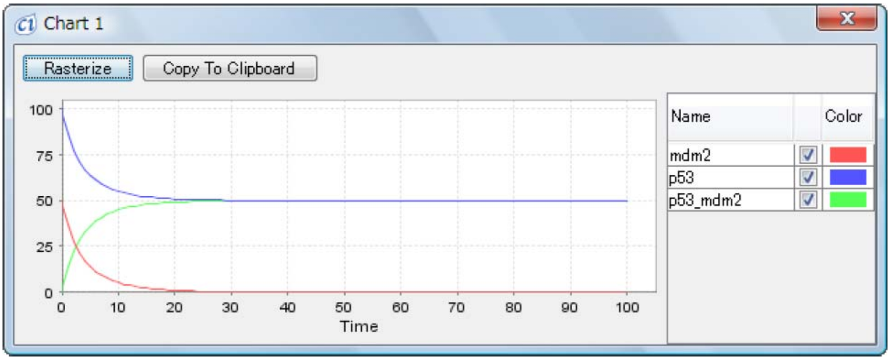


Fig. 4.54



Name		Color
mdm2	☑	🟥
p53	☑	🟦
p53_mdm2	☑	🟩

Fig. 4.55

For the combinations of element types possible in Steps 1 and 2, there are nine possible patterns; the specifics are left as an exercise to the reader.

4.7.5 Dissociation

There are cases in which a protein complex will break into its original proteins. This event is called *dissociation* and can be modeled in the following way by using a *dissociation process* ◆:

Step 1: Place three entities ●, ●, ● on the canvas. Set appropriate names and values for each entity. For example, (name, initial value)=(p53, 50), (mdm2, 0), (p53_mdm2, 50).

Step 2: Place the process ◇ on the canvas and set the process name. For this example, set process name to “dissociation”.

Step 3: Create a process connector from entity p53_mdm2 ● to process ◇. In addition, create process connector from the process ◇ to p53 ● and mdm2 ●.

Step 4: Set the dissociation rate as the speed of the process ◇. The default value is 1.

The result should look like Figure 4.56.

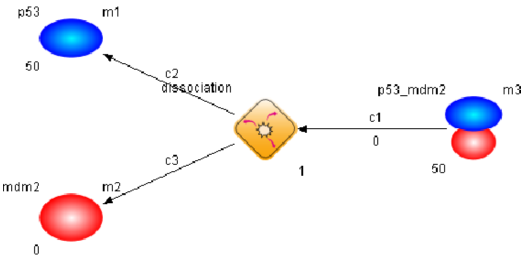
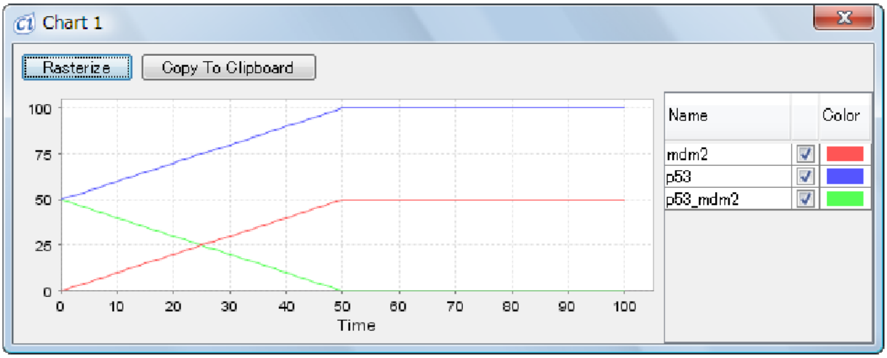


Fig. 4.56

Creating a graph for all entities in this model and executing its simulation, the result should look like Figure 4.57.



Name		Color
mdm2	☑	🟥
p53	☑	🟦
p53_mdm2	☑	🟩

Fig. 4.57

If the dissociation rate depends on the concentration of p53_mdm2 ●, set the dissociation rate to be an appropriate value. For example, m3/20 would be an arbitrary but dependent value. In this case, the model looks like Figure 4.58 and the simulation result should look like Figure 4.59.

Similar to the previous event, there are nine possible patterns for the combinations of entities and processes given in Steps 1 and 2.

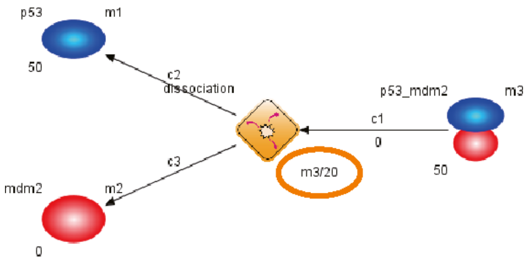
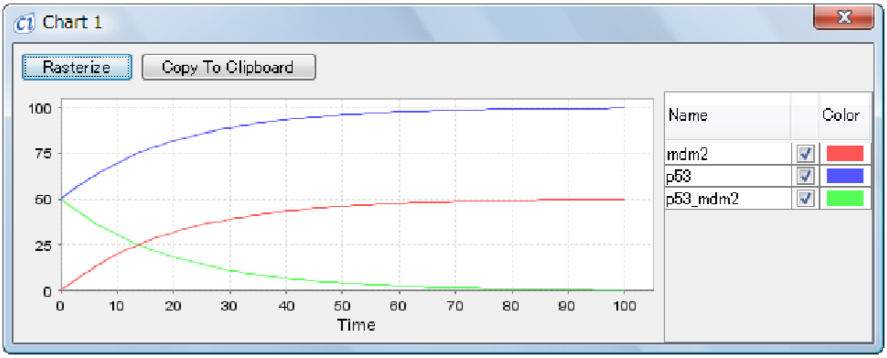


Fig. 4.58



Name		Color
mdm2	☑	🟥
p53	☑	🟦
p53_mdm2	☑	🟩

Fig. 4.59

Note that if the entity and process are both discrete, then it is impossible to make the dissociation rate fully dependent on p53_mdm2 concentration.

4.7.6 Inhibition

Up until now all connectors used were process connectors. This example will utilize the *inhibitory connector*, one of the two types of connectors remaining, which acts as *inhibition* of a process.

Inside the cell, it is possible to block transcription with certain drugs. By modifying the model used in Section 4.7.3 with an inhibitory connector, we can simulate this event:

Step 1: Add some entity ◎ to the canvas. And set an appropriate name such as "doxorubicin".

Step 2: Add a process ▭ to the canvas and draw a process connector from this process to doxorubicin ◎ .

Step 3: Draw an inhibitory connector ⊥ from doxorubicin ◎ to the transcription process ◈ . The default value of the threshold is set to 0.

Step 4: Connect the process transcription to the entity mRNA using a process connector.

The result should look like Figure 4.60.

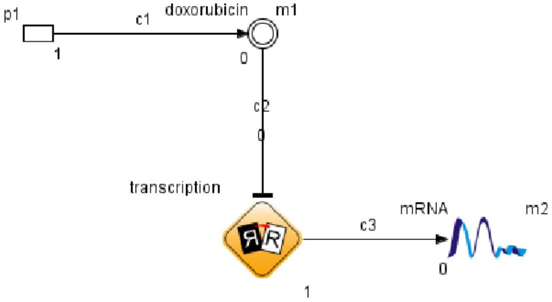
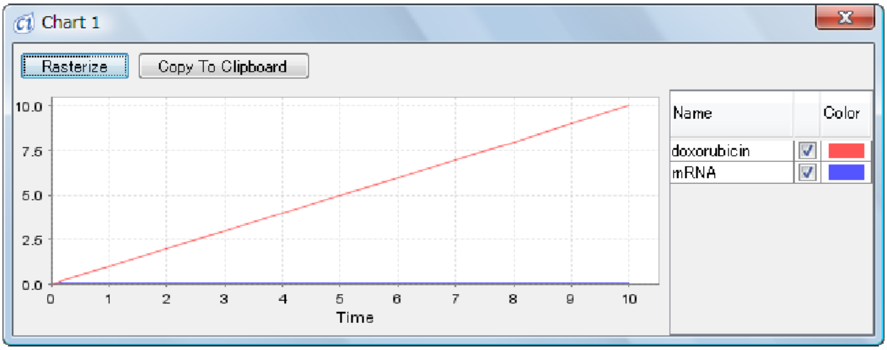


Fig. 4.60

The inhibitory connector has the effect of inhibiting a certain process. The threshold determines whether or not a process is completely inhibited: if the value is greater than the threshold, the process will not be executed.

In this model, as is seen in Figure 4.60, the initial value of doxorubicin ◎ is 0, so initially the transcription process is executed, but right after, the doxorubicin ◎ value becomes greater than 0 and the transcription process is stopped as in Figure 4.61.



Name		Color
doxorubicin	☑	🟥
mRNA	☑	🟦

Fig. 4.61

The strength of a drug can be specified through the threshold value. For example, if the threshold is raised to 5 (Figure 4.62), the drug is a weak drug which has effects only when used at values of 5 or higher (Figure 4.63).

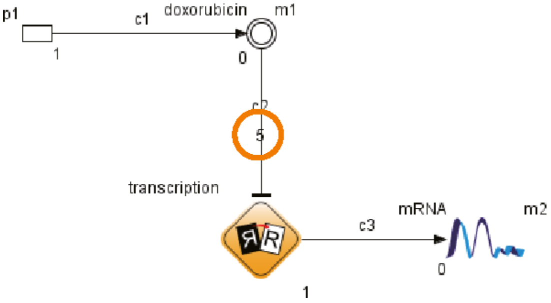
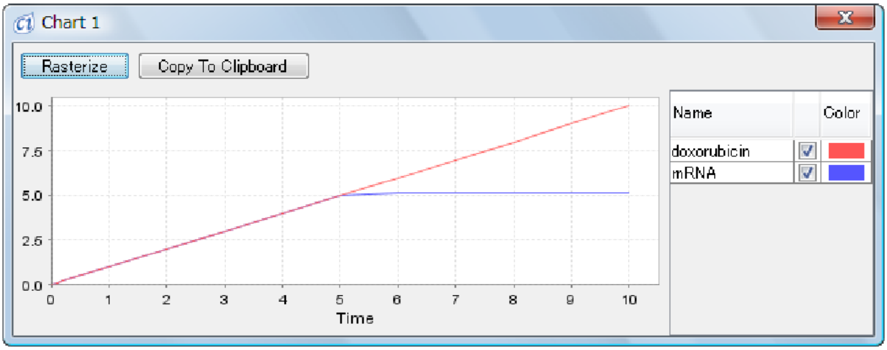


Fig. 4.62



Name		Color
doxorubicin	☑	🟥
mRNA	☑	🟦

Fig. 4.63

Now for the entity ◎ and process ▭ added in Steps 1 and 3, combinations of (discrete, discrete), (discrete, continuous), and (continuous, continuous) are possible.

4.7.7 *Phosphorylation by Enzyme Reaction*

This example uses the *association connector*, the last unused connector.

Enzyme reactions are key to cellular processes. Although enzymes are not affected by the reactions, they help speed the reactions. These types of events can be modeled by using the association connector. The model below shows phosphorylation using an enzyme ⬮ and a *phosphorylation process* +P :

Step 1: Add two entities ⬬ and ⬬P to the canvas. Set appropriate names and initial values such as (name, initial value)=(p53,10), (p53{p},0).

Step 2: Add a phosphorylation process +P to the canvas. Set an appropriate name such as “phosphorylation”. Then draw a process connector from entity p53 to phosphorylation and another from phosphorylation to p53{p} .

Step 3: Set the enzyme reaction speed by changing the speed of the phosphorylation process . The default value should be set to 1.

Step 4: Add another entity as an enzyme and set an appropriate name such as "CAK".

Step 5: Add a translation process to the canvas and draw a process connector from translation to CAK .

Step 6: Draw an association connector from CAK to phosphorylation .

Step 7: Though the default value of the threshold is 0, for the purpose of demonstrating the association connector, set this to 1.

The resulting model should look like Figure 4.64.

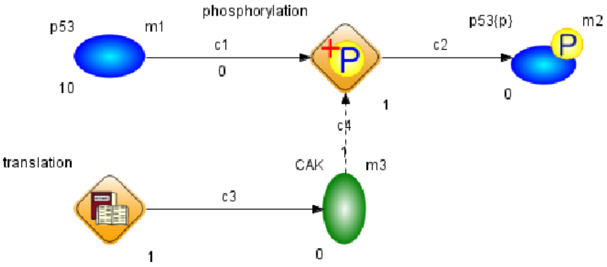


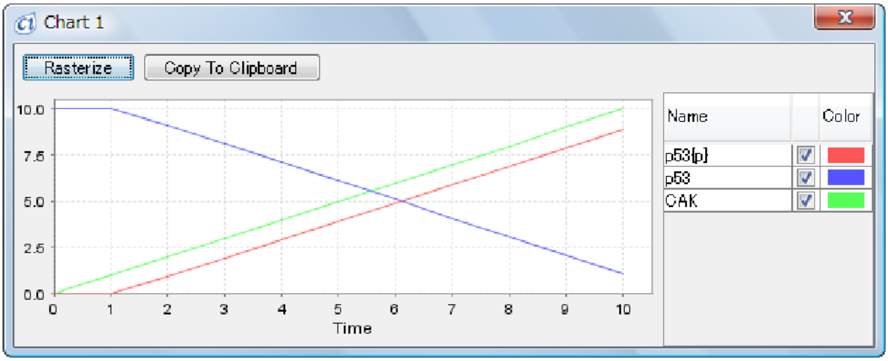
Fig. 4.64

The association connector helps modeling enzyme reactions. The connector is activated when the value of the entity is greater than the threshold. Although the connector acts nearly identically to the process connector, the association connector does not decrease the contents of entity (CAK) by executing the process (the phosphorylation process), similar to a real enzyme action.

Creating graphs for all entities in this model, the result of simulation should look like Figure 4.65.

In this model, the initial value of CAK is 0 as is given in Figure 4.64, and therefore initially the association connector is not active and the phosphorylation process is not executable. Soon after, the value of CAK becomes greater than 1 and it activates the association connector , which starts the phosphorylation process .

The strength of enzymes can be represented by the threshold. For example, if the enzyme threshold is changed from 2 to 5 (Figure 4.66), the quantity of the enzyme with threshold 5 must be increased to produce the same effect (Figure 4.67).



Name		Color
p53{p}	☑	🟥
p53	☑	🟦
CAK	☑	🟩

Fig. 4.65

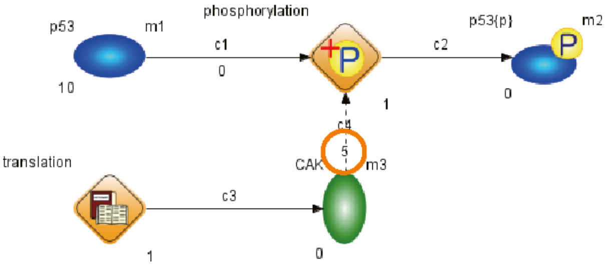
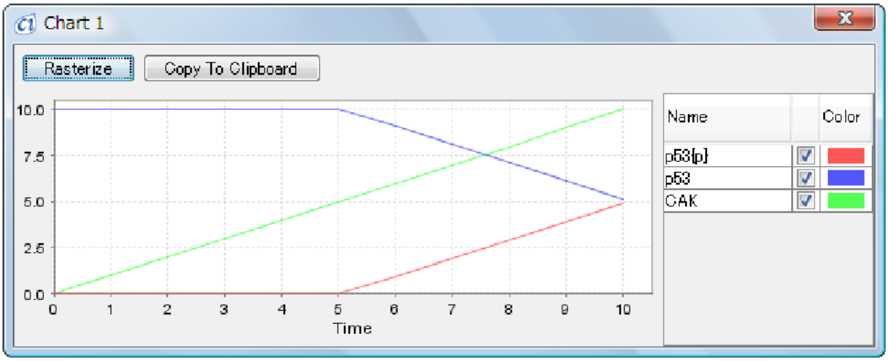


Fig. 4.66



Name		Color
p53{p}	☑	🟥
p53	☑	🟦
CAK	☑	🟩

Fig. 4.67

Inside the actual cell, phosphorylation by enzymes depends on both the enzyme (m3) and substrate (m1). In such a situation an appropriate speed value would be (m1*m3)/10 (the value of 10 is arbitrary). Making these changes and using continuous elements, the result should look like Figure 4.68.

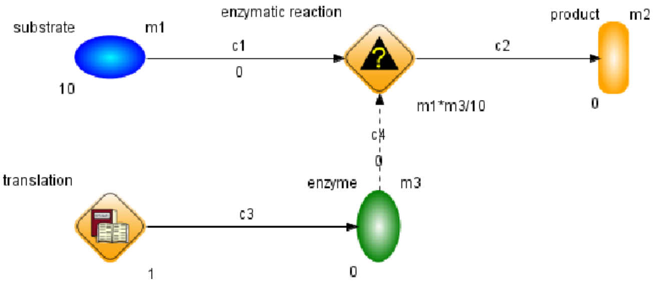
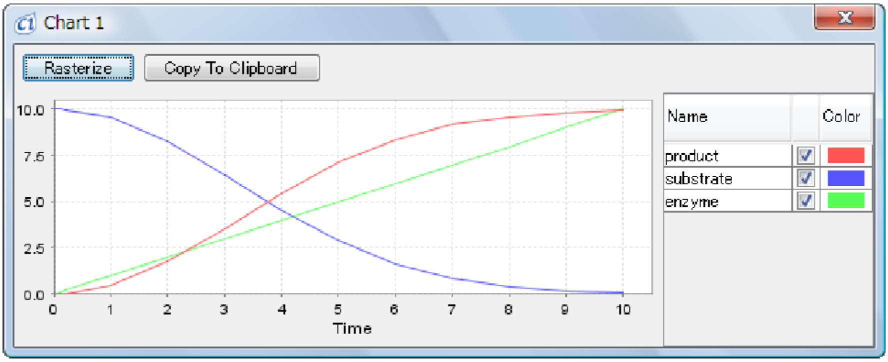


Fig. 4.68

The simulation result is shown in Figure 4.69.



Name		Color
product	☑	🟥
substrate	☑	🟦
enzyme	☑	🟩

Fig. 4.69

Note that the process and entity added in Steps 1 and 3 can be (discrete, discrete), (discrete, continuous), and (continuous, continuous).

COLUMN 4

Process Conflicts

Consider a network made from two discrete processes p1 and p2 and three discrete entities e1, e2, and e3 (shown in Figure 4.70). The initial value of entity e1 is 5 while e2 and e3 have a value of 0. Looking at time 2[pt], e1 has a value of 1 and both processes are executable. However, in order to execute both p1 and p2, there must be a value of 2 or more (1+1=2). This situation is called a *conflict*. In Cell Illustrator, processes are given *priority* in order to resolve these types of conflicts. If the processes are on the same priority level, then a process is randomly selected.

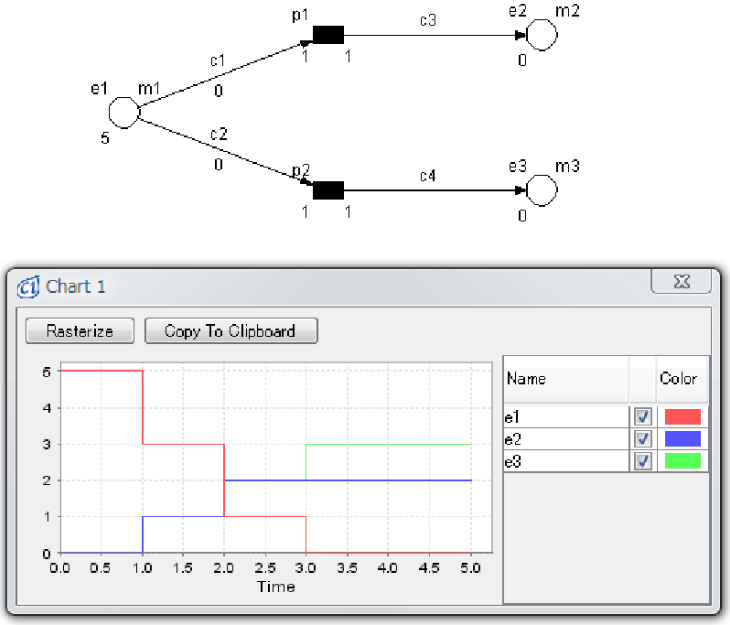


Fig. 4.70

4.8 Conclusion

In this chapter, we described installation of Cell Illustrator, its basic concepts, model editing, and execution, as well as the workflow of converting a real cellular process to a model. Although the generic processes and entities are outside the scope of this book, see [3, 4] for theoretical background and applications in pathway modeling.

COLUMN 5

Using Generic Elements

While modeling various pathways, it is sometimes apparent that there are some events which discrete and continuous elements cannot model. For those cases, Cell Illustrator, which uses the HFPNe framework, allows for generic entities and processes. For example, in order to model the transcription process of Section 4.7.3 in the level of DNA sequences, it is possible to create a model as seen in Figure 4.71. Modeling the transcription process in the DNA sequence level can visualize phenomena such as alternative splicing and frame shifts. A protein with many modifications can be effectively modeled with a generic entity. Table 4.2 defines the actual behavior of the generic process. By executing the model "fig4_71_generic_model_transcription.xml" contained in the provided CD-ROM, we can see how the actual base pairs are transcribed into mRNA.

Table 4.2

connector	update function
c1	`m1;`

c2	`import("gon.Transcription");`
	`totalnum=m1.length();`
	`num=m2.length();`
	`if (totalnum ≥ num){`
	`nextcode=m1.substring(totalnum−num−1,`
	`totalnum−num);`
	`newsequence=m2+Transcription::Trans(nextcode);`
	`}`
	`else{newsequence="";};`
	`newsequence;`

c3	`m3;`
c4	`if (m1.length()==m3.length()){m3=m3+1;}`

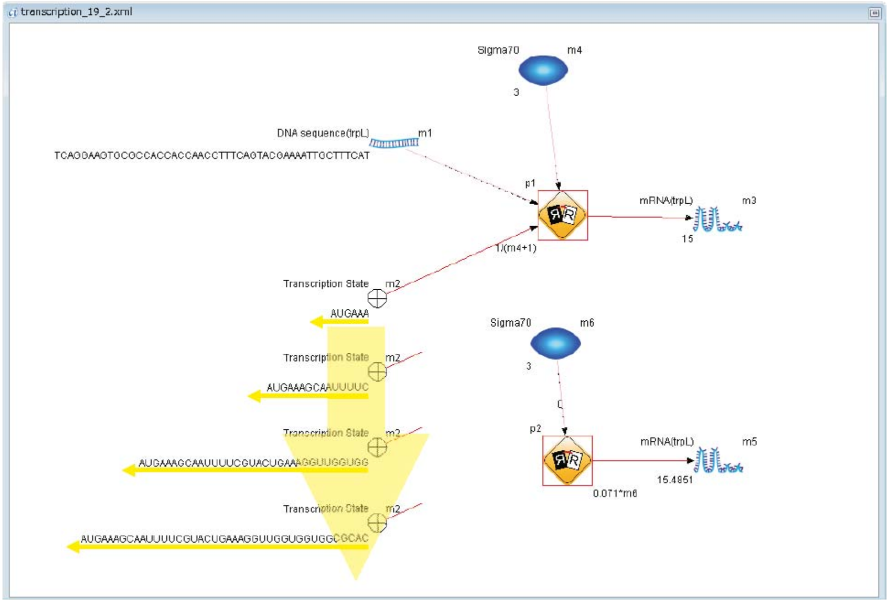


Fig. 4.71 Above: a model using generic entities. Below: a model using only a continuous entity.

Chapter 5
Pathway Modeling and Simulation

In this chapter we will combine the modeling methods described in Chapter 4 to model and simulate larger pathways. Specifically, we will create models for the gene regulatory, metabolic, and signaling pathways using Cell Illustrator. This chapter should enable the reader to create a variety of new pathways.

5.1 Modeling Signaling Pathway

In this section, we will model the signaling pathway for the epidermal growth factor receptor. Although this is a rough model based on known biological facts, this type of model creation will provide us a new method for organizing biological knowledge on pathways.

5.1.1 Main Players: Ligand and Receptor

Cell growth and differentiation are regulated by signaling molecules (*ligands*) from the outside of the cell. The cell has a molecule called a *receptor* that binds and recognizes the ligand and sends signals to induce a new reaction in the cell. For example, the rat adrenal pheochromocytoma cell PC12 is known to differentiate to nerve cells once the tropomyosin receptor kinase A (TrkA) receives the nerve growth factor (NGF). In this section, we will model the early stage of the signaling pathway for the epidermal growth factor (EGF). The receptor for EGF is the epidermal growth factor receptor (EGFR) on the cell membrane. EGFR is also called ErbB1 and belongs to the ErbB family, a subfamily of tyrosine kinase receptors. EGFR is expressed in a variety of cells such as epidermal, mesenchymal, and neuronal cells, and it controls cell growth and differentiation. It is also known that EGFR overexpression and mutations related to EGFR overexpression could result in cancer.

5.1.2 Modeling EGFR Signaling with EGF Stimulation

There are a few steps in creating the canvas. First create a new canvas by pressing the icon (Figure 5.1). Since the Biological Elements Dialog will be used to create the model, open the dialog . The processes and entities used will be continuous unless otherwise noted.

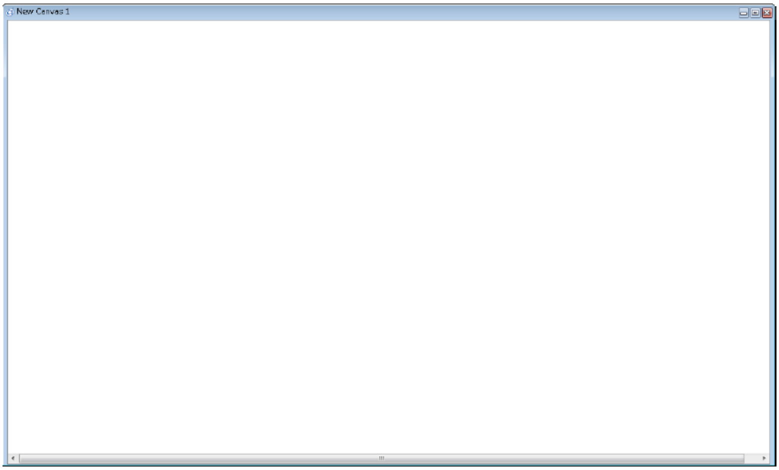


Fig. 5.1

5.1.2.1 Cell Placement

Since the model involves signals on the cell membrane, create a cell illustration on the canvas as seen in Figure 5.2. Click on “Edit Parts” and select [cells] → [animal_cell_blank]. To change the shape, select the illustration and drag the corners while holding the Shift key. Although the shape does not affect simulation, modeling is made simpler since placing the cursor on the illustration displays location such as [plasma_membrane] or [cytoplasm].

5.1.2.2 Placement of EGFR

EGFR is known to be located on the cell membrane.

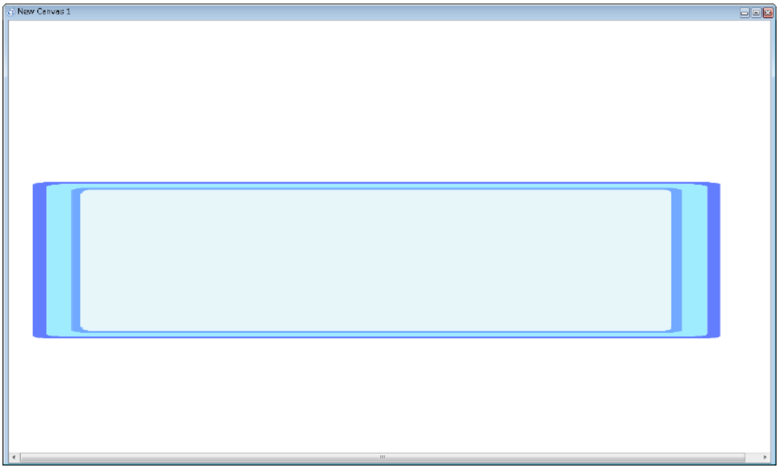


Fig. 5.2

Exercise 5.1.1
By following the steps in Section 4.7.3, create the translation process for EGFR. The entity EGFR represents the concentration of EGFR and use Y for its icon. Set the translation speed to 1 (Figure 5.3).

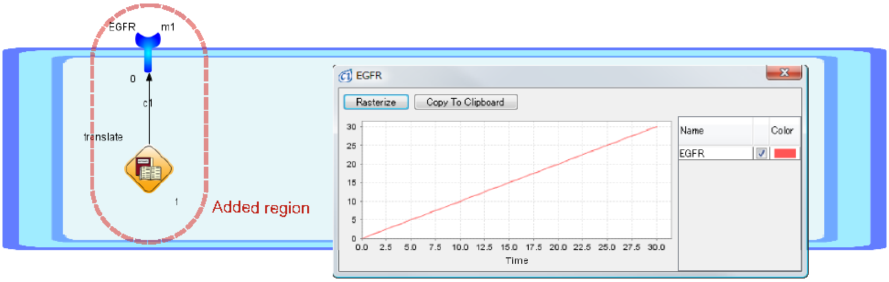


Fig. 5.3

Exercise 5.1.2
Simulate the model made in Exercise 5.1.1 and confirm the time course of the EGFR concentration with the graph.

From the simulation of the model, we see that the concentration of EGFR is continually increasing. In reality, the concentration of EGFR will stabilize at some value (steady state) because of the natural degradation built into the cell. In order to model this, we need to add a degradation process to the model.

Exercise 5.1.3
By following Figure 5.3 and Section 4.7.1, add a degradation process D with a speed of 0.04 times the concentration of EGFR.

The resulting canvas should look like Figure 5.4.

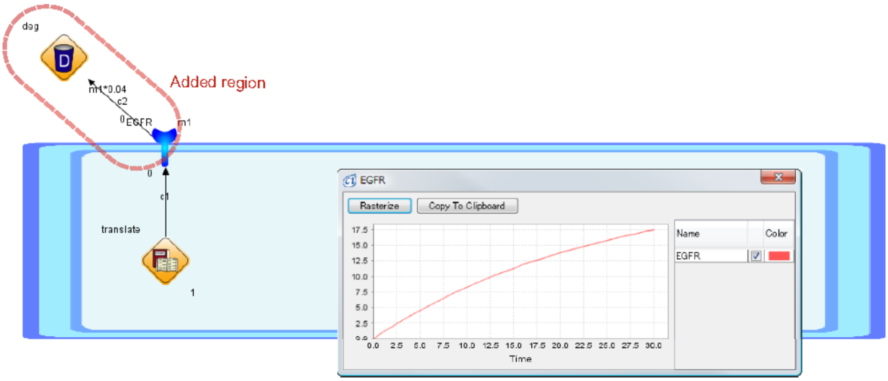


Fig. 5.4

5.1.2.3 Binding of EGF and EGFR

EGFR is known to receive the extracellular molecule EGF for signal transduction.

Exercise 5.1.4
Place an entity EGF outside the cell. Use ● for its icon.

Exercise 5.1.5
By following Section 4.7.3, add a process ◇ which produces EGF.

Exercise 5.1.6
Similarly to Exercise 5.1.3, add a degradation process D for EGF with a speed of 0.01 times the concentration of EGF.

The canvas at this point should look like Figure 5.5.

In order for EGFR to activate the signal transduction, EGF and EGFR must be bound together.

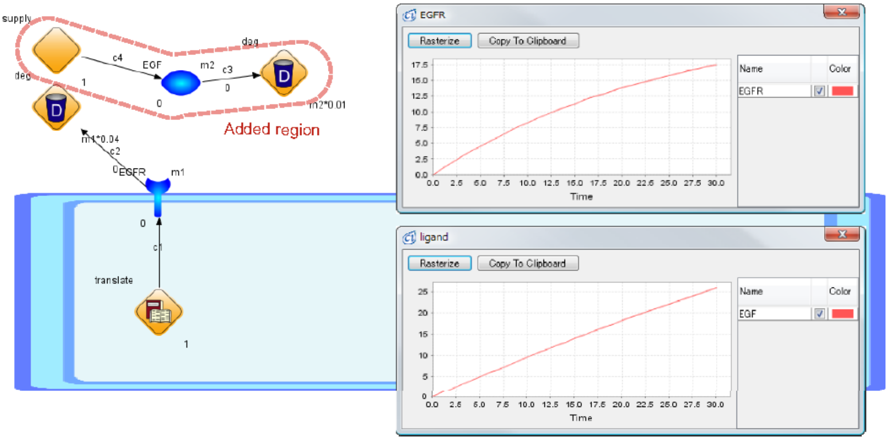


Fig. 5.5

Exercise 5.1.7
By following Section 4.7.4, add a process which combines EGF and EGFR with a speed of 0.1 times the product of the concentrations of EGF and EGFR. Use for the complex of EGF and EGFR denoted as EGF/EGFR.

Sometimes the complex EGF/EGFR will break back into EGF and EGFR.

Exercise 5.1.8
By following Section 4.7.5, add a dissociation process for the complex EGF/EGFR . The dissociation speed should be slower than the binding speed and is set to 0.001 times the concentration of the complex .

Exercise 5.1.9
Add a degradation process for the complex EGF/EGFR with a speed of 0.01 times the concentration of the complex .

The resulting canvas should look like Figure 5.6.

5.1.2.4 Dimerization of the Complex EGF/EGFR

The complex EGF/EGFR is known to *dimerize*. We will include this in the model.

We will introduce a new feature for this. In Chapter 4, the speed of a process decreased the values of entities connected by its input connectors and increased the values of entities connected by its output connectors. In other words, the process moves values at the same rate for every connector. However, since Cell Illustrator

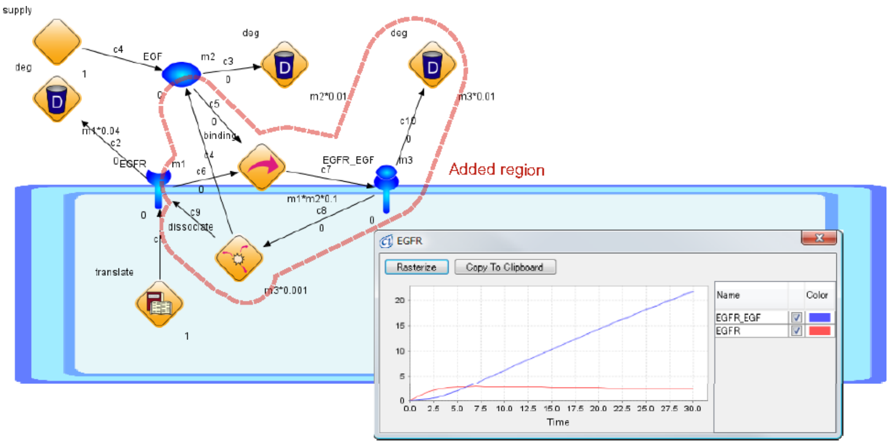


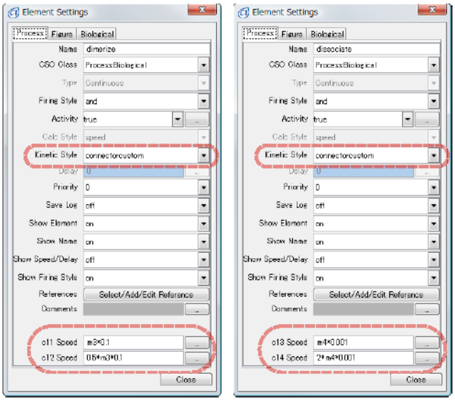
Fig. 5.6

is based on the HFPNe framework, the rate of the flow for each connector can be changed. We will use this in the following exercises:

Exercise 5.1.10

By following Section 4.7.4 on binding, create a *dimerization process* ◆ by which the complex EGF/EGFR ● is dimerized. The icon for the dimerized entity ●● is provided in the "PrepareElements" folder on the CD-ROM in the file ReceptorLigandDimer.xml. We call this dimer an EGF/EGFR dimer. Set the speed as 0.1 times the concentration of the complex ● . However, since only one dimer is made from two complexes, the rate of flow for each connector needs to be changed.

- Select the dimerization process ◆ . If the Element Settings Dialog is not displayed, call it up by clicking ☰ .
- In the Process tab of the dialog, set the kinetic style field as "connectorcustom".
- This will make it possible to enter the speed of each connector at the bottom (left side in Figure 5.7).



Name	dimerize	dissociate
CSO Class	ProcessBiological	ProcessBiological
Type	Continuous	Continuous
Firing Style	and	and
Activity	true	true
Calc Style	speed	speed
Kinetic Style	connectorcustom	connectorcustom
Delay	0	0
Priority	0	0
Save Log	off	off
Show Element	on	on
Show Name	on	on
Show Speed/Delay	off	off
Show Firing Style	on	on
References	Select/Add/Edit Reference	Select/Add/Edit Reference
Comments		
c11 Speed	m3*0.1	
c12 Speed	0.5*m3*0.1	
c13 Speed		m4*0.001
c14 Speed		2*m4*0.001

Fig. 5.7

Exercise 5.1.11
By following Section 4.7.5, create a dissociation process for EGF/EGFR dimer. Since the dissociation process is slower, set the process to a speed equal to 0.001 times the dimer concentration. However, since each dimer produces two monomers, the parameters need to be reversed from the previous exercise. The steps are below:

- Select the dissociation process , and when the dialog of the Element Settings Dialog is open, select the Process tab.
- On the process tab of the dialog, select "connectorcustom" in the field of Kinetic Style.
- Then on the bottom of the dialog, set the speed for each connector (Figure 5.7 right).

Exercise 5.1.12
Add a degradation process for EGF/EGFR dimer with a speed of 0.01 times its concentration.

The result should look like Figure 5.8.

5.1.2.5 Phosphorylation and Dephosphorylation of EGFR

After EGF/EGFR dimers in the cytoplasm are phosphorylated, the signal transduction pathway will be activated.

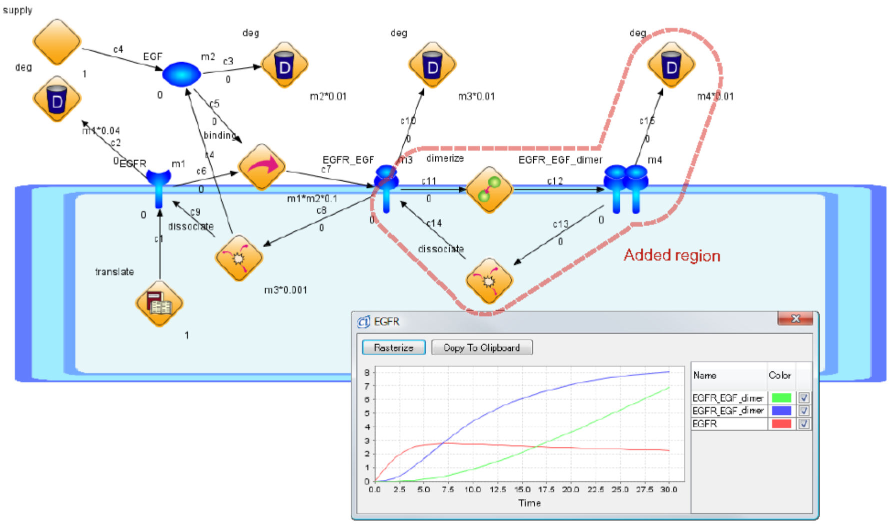


Fig. 5.8

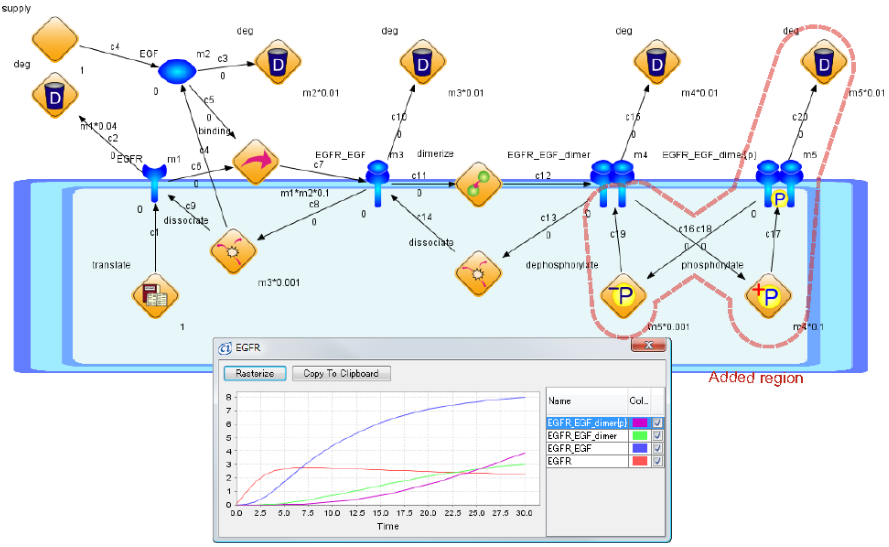
Exercise 5.1.13
Add a phosphorylation process +P for the dimer with a speed of 0.1 times the dimer concentration. The illustration for the phosphorylated dimer P is provided by PhosphorylatedReceptorLigandDimer.xml in the folder "PrepareElements" on the CD-ROM.

Exercise 5.1.14
Add a *dephosphorylation process* −P for the phosphorylated dimer P . Since this process is less active than the phosphorylation process, set the speed of this process to 0.01 times the concentration of the phosphorylated dimer P .

Exercise 5.1.15
Add a degradation process D for the phosphorylated dimer P . Set the speed of degradation to 0.01 times its concentration in a similar way to other molecules.

This should result in a model looking like Figure 5.9. This model omits the enzymes for phosphorylation and dephosphorylation.

Simulating this model, one can see that EGF and EGFR are bound to make a complex, then these complexes are dimerized and phosphorylated. Looking at the graph of simulation, the concentration of EGFR goes up first, then the concentrations of EGF/EGFR complex, its dimer, and its phosphorylation go up in this order.



Name	Col..	
EGFR_EGF_dimer(p)	🟪	☑
EGFR_EGF_dimer	🟩	☑
EGFR_EGF	🟦	☑
EGFR	🟥	☑

Fig. 5.9

In this case, the possibility of dissociation of the phosphorylated dimers is not considered; if this is important to the model, the model must be modified to take this into account. Similarly, there is the possibility of the ligand dissociating in the dimerized form. Although these types of possibilities were ignored, the important fact is that it is easy to model, simulate, and modify existing pathways by including new ideas on the pathways.

5.1.2.6 Inhibition of EGFR Signaling by Agent

AG compounds are known to inhibit the EGFR signal pathway. We will add the effects of an AG compound to this model.

Specifically, AG1478 is termed a *tyrosine kinase inhibitor* of the EGFR pathway. In other words, AG1478 inhibits the phosphorylation in the model.

Exercise 5.1.16
Add an entity AG1478 and modify the phosphorylation process so that AG1478 inhibits the phosphorylation. For the illustration of AG1478, use ? and set the initial value of this entity to 2. Add an inhibitory connector ⊥ from AG1478 to the phosphorylation process +P and set the threshold to 1. Since AG1478 wears over time, add a degradation process D for AG1478 and set the speed to 0.01 times the concentration of AG1478.

Exercise 5.1.17
Simulate the new model and compare the effects of adding AG1478.

The newly updated model is shown in Figure 5.10. The phosphorylation process is inhibited and the concentration of the phosphorylated dimer stays at 0. After running the simulation for a longer time, the inhibition stops because of AG degradation and the phosphorylated dimers appear again (Figure 5.11).

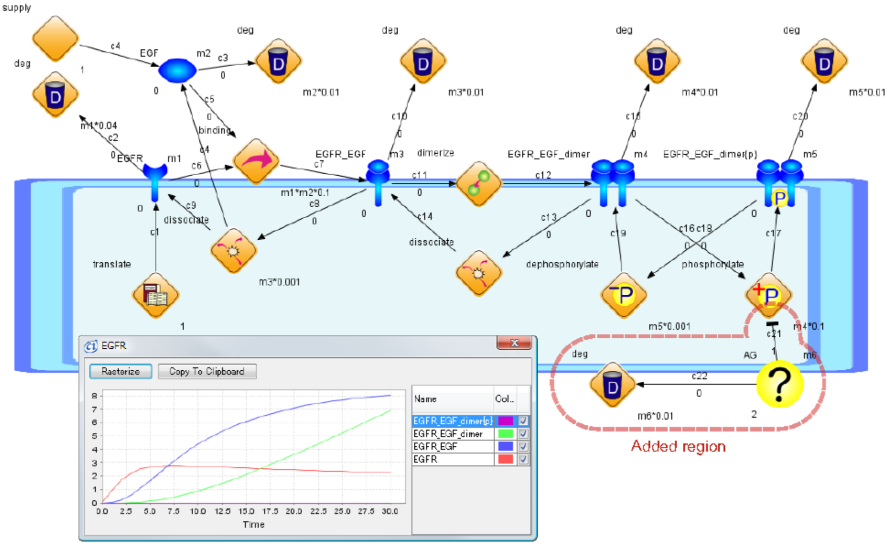
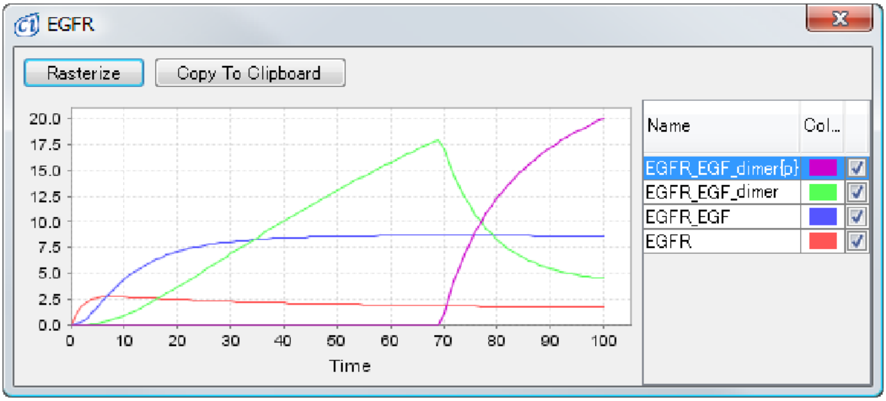


Fig. 5.10



Name	Col...	
EGFR_EGF_dimer{p}	🟪	☑
EGFR_EGF_dimer	🟩	☑
EGFR_EGF	🟦	☑
EGFR	🟥	☑

Fig. 5.11

COLUMN 6

Editing Illustrations

It is very typical that we want more customized illustrations for elements instead of the icons provided as in Chapters 3 and 4. Cell Illustrator has a function to create custom illustrations by editing and changing them by users themselves. Right-click the appropriate element and select [Edit Image] (Figure 5.12). This starts the CI SVG Editor. There are a few easy steps to modify and create custom illustrations.

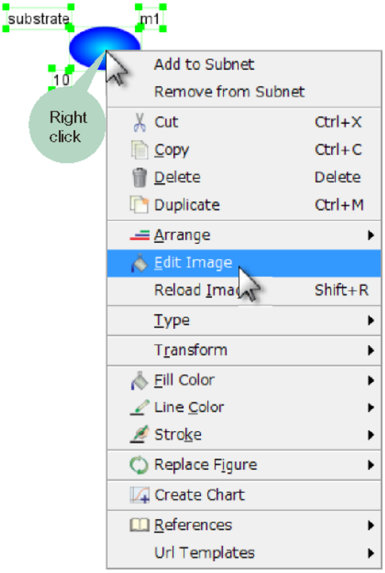


Fig. 5.12

Changing the Color

It is simple to change the blue protein ● to a red one. Click "Edit Image" and start the CI SVG Editor for editing ● . The blue protein ● is defined by a circular gradation from light to deep blue. Redefining this gradation can change the color. Select the [Dialogs] → [Show Resources] on the menu bar to display the resource dialog bar. Then select the [Radial Gradient] field inside the tab to show a gradation definition called “color1”. Bring up the Radial Gradient Property Dialog and redefine the gradation by double clicking color1. Since there are two colors used in this gradation, both must be changed. There are two ways to do this, but the easier way is to select the square near the top of the dialog. Select a light pink for the left and red for the right. After pressing OK and closing the dialog, the CI SVG Editor should show the changes and make the element red. Lastly, the changes will be applied to all elements on the current canvas by selecting [File] → [Save] on the menu.

🟢🟧 Combining Two Elements

In order to create the combination of two proteins 🟢 and 🟧 , these illustrations must be combined. The basic workflow involves copying one element to the other. First, place both 🟢 and 🟧 on the canvas and select “Edit Image” for each to open it in CI SVG Editor. To copy 🟧 to 🟢 , select [Edit] → [Copy] while the window for 🟧 is open. Then open the window for 🟢 and select [Edit] → [Paste]. Ensure that 🟧 is pasted on 🟢 and change the size of the canvas using CI SVG Editor. On the *Resize Dialog* which is at [Transforms] → [Canvas] → [Resize], select the width and height. In this case, about 200 horizontal and 212 vertical is appropriate. After pressing OK, drag the 🟧 to fit in the canvas. Run [File] → [Save] to commit changes.

AA Changing Aspect Ratio

To change the horizontal and vertical scales of an element, simply grab the corner of the element, and drag it. This can be done without using CI SVG Editor. Note that aspect ratio can be preserved during scaling by holding the Shift button.

+ Others

When editing and modifying, there is a chance that the element will not be displayed correctly on the canvas. In this case, right-click the element and run [Reload Image] to refresh the image.

In addition, CI SVG Editor can be used to create an illustration from scratch. These advanced features should be used as needed.

5.2 Modeling Metabolic Pathways

The glycolysis pathway is a pathway which breaks down glucose into pyruvic acid. In usual chemical equations form it is written as

$$\text{Glucose} + 2\text{ADP} + 2\text{NAD} \rightarrow 2\text{Pyruvic acid} + 2\text{ATP} + 2\text{NADP}$$

However, in reality, the model is made up of 10 separate reactions as seen in Figure 5.16. In this section, this pathway will be modeled by showing how a metabolic pathway is modeled with Cell Illustrator.

5.2.1 Chemical Equations and Pathway Representations

Figure 5.13 shows a pathway representation for the following chemical equation:

$$\text{Glucose} + \text{ATP} \longrightarrow \text{Glucose-6-phosphate} + \text{ADP}$$

Although discrete elements are used here, continuous elements could be used just as well.

The chemical reaction can be thought of as separated into the left and right sides. Each substance is represented as an entity and the process in between represents the reaction. The value of each entity represents the number of molecules, and the execution of the process on Cell Illustrator is equivalent to the reaction occurring in cells. Once the model in Figure 5.13 is executed, the glucose and ATP tokens decrease and move to glucose-6-phosphate and ADP, which represent the chemical equation above.

Figure 5.13 also includes the enzyme which assists the reaction. Since the enzyme is not affected by the reaction, an association connector is used here.

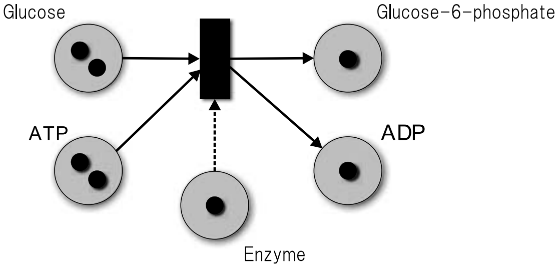


Fig. 5.13

5.2.2 Michaelis-Menten Kinetics and Cell Illustrator Pathway Representation

An enzyme reaction occurs through interaction of the substrate and enzyme. Let E be an enzyme, S be a substrate, ES be the enzyme-substrate complex, and P be the product. The enzyme reaction can be written as follows:

$$\mathrm{E} + \mathrm{S} \rightleftharpoons \mathrm{ES} \longrightarrow \mathrm{E} + \mathrm{P}$$

This reaction can be represented using continuous elements as seen in Figure 5.14. If there is a sufficient amount of the enzyme and substrate, the Michaelis-Menten kinetics can be used to simplify the reaction rate as

$$v = \frac{V_{\max}[S]}{[S] + K_m}$$

Here the $V_{\max}$ is the highest reaction rate, and K_m is called the *Michaelis constant*.

When the concentration of a given enzyme is constant, $V_{\max}$ and K_m are also constant and this fact allows the equation to be reduced to

$$\mathrm{S} \longrightarrow \mathrm{P}$$

Figure 5.15 shows this as a model using continuous entities. Since an enzyme is used for this metabolic reaction, an association connector is drawn here.

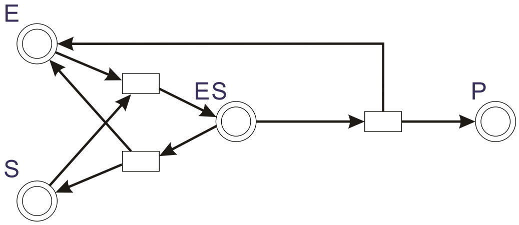


Fig. 5.14

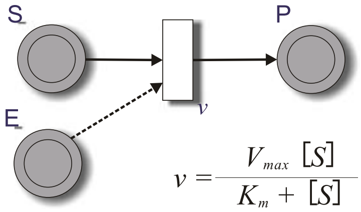


Fig. 5.15

5.2.3 Creating Glycolysis Pathway Model

The glycolysis pathway consists of the following parts:

(1) Glucose + ATP $\longrightarrow$ Glucose-6-phosphate + ADP
(2) Glucose-6-phosphate $\rightleftharpoons$ Fructose-6-phosphate
(3) Fructose-6-phosphate + ATP $\longrightarrow$ Fructose-1,6-bisphosphate + ADP
(4) Fructose-1,6-bisphosphate $\rightleftharpoons$ Glyceraldehyde-3-phosphate + Dihydroxyacetone phosphate
(5) Dihydroxyacetone phosphate $\rightleftharpoons$ Glyceraldehyde-3-phosphate
(6) Glyceraldehyde-3-phosphate + Phosphate + NAD$^+$ $\longrightarrow$ 1,3-Bisphosphoglycerate + NADH
(7) 1,3-Bisphosphoglycerate + ADP $\rightleftharpoons$ 3-Phosphoglycerate + ATP
(8) 3-Phosphoglycerate $\rightleftharpoons$ 2-Phosphoglycerate
(9) 2-Phosphoglycerate $\rightleftharpoons$ Phosphoenolpyruvate
(10) Phosphoenolpyruvate + ADP $\rightleftharpoons$ Pyruvate + ATP

Since this is a continuous chain of reactions, the right-hand side of each equation becomes the left-hand side of the next reaction. In addition, there are enzymes for each reaction from (1) to (10) (Table 5.1). A summary of these reactions is given in Figure 5.16.

Table 5.1 Enzymes functioning in the glycolytic pathway.

	Enzyme	Reaction
(1)	Hexokinase	irreversible
(2)	Phosphoglucose isomerase	reversible
(3)	Phosphofructokinase	irreversible
(4)	Aldolase	reversible
(5)	Triosephosphate isomerase	reversible
(6)	Glyceraldehyde 3-phosphate	reversible
(7)	Phosphoglycerate kinase	reversible
(8)	Phosphoglyceromutase	reversible
(9)	Enolase	reversible
(10)	Pyruvate kinase	irreversible

The numbers (1)–(10) correspond to those in Figure 5.16.

First, let us model the initial stage of the reaction in Figure 5.16. This is the reaction given in Section 5.2.1:

$$\text{Glucose} + \text{ATP} \longrightarrow \text{Glucose-6-phosphate} + \text{ADP}$$

As previous models, simply turn substances such as glucose into entities and connect each entity to the reaction process. For glucose and glucose-6-phosphate, the illustration for small_molecule ⌬ should be used, while for the metabolic reaction we use ◆. Since ATP ATP and ADP ADP are also used, connect those entities

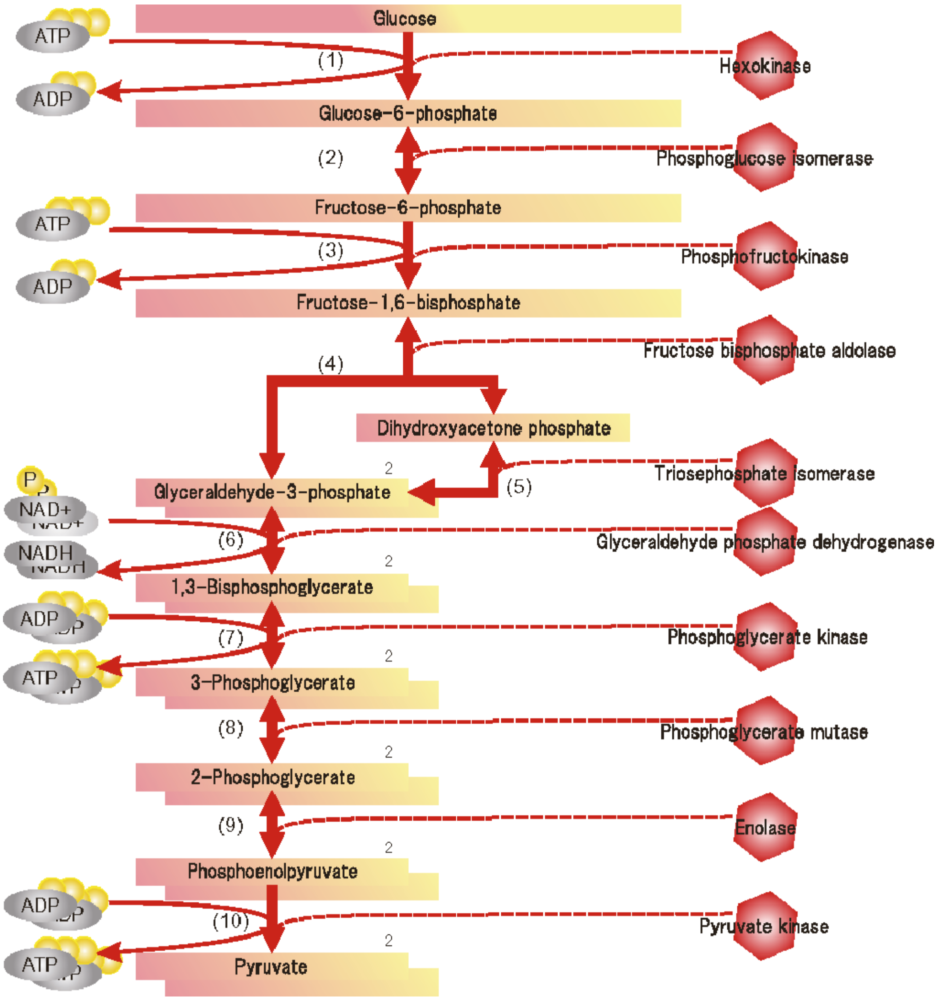


Fig. 5.16

to the process ◇. In addition, the enzyme hexokinase ● should be added to the canvas, and connected to the process ◇ with an association connector ⇣ with a threshold of 3 (Figure 5.17).

Since glucose ⌬, glucose-6-phosphate ⌬, ATP ATP, and ADP ADP degrade, add a degradation process D for each of these entities. The degradation speed should be 0.1 times the concentration for all entities. The threshold of glucose-6-phosphate should be set to 1.

Also add a degradation process D for hexokinase ● with the same speed of 0.1 times the concentration. Since in this case it can be assumed that hexokinase ● was already produced, set the initial value to 5. Also add a translation process ◇ with a speed of 1 to produce hexokinase (Figure 5.18).

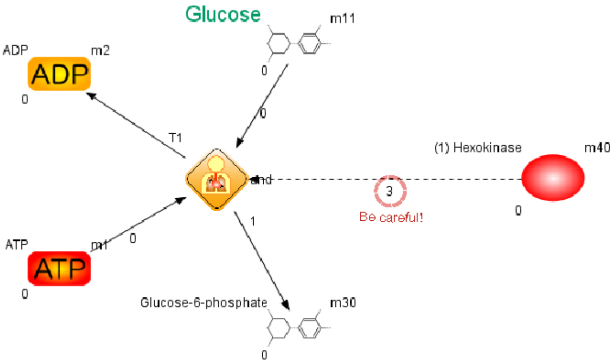


Fig. 5.17

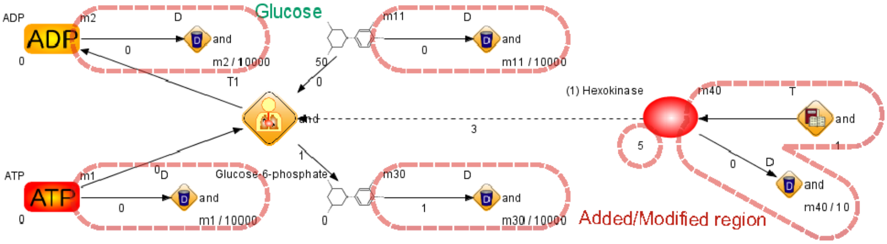


Fig. 5.18

For the metabolic reaction ◇, its speed should be set according to the Michaelis-Menten equation at $(V_{\max}[S])/([S]+K_m)$. In order to set the values of $V_{\max}$ and K_m, create two entities ◎ named Pv and Pk. The value of the entity Pv will be used for $V_{\max}$ while the value of the entity Pk will be used for K_m. Here let $V_{\max}=1$ and $K_m=1$ temporarily. By setting $V_{\max}$ and K_m as variables, it becomes extremely easy to change these parameters when the model gets larger and complicated.

Assuming the concentrations of ATP ATP and ADP ADP are high enough, set the initial values of ATP and ADP to 300 and set the initial value of glucose to 50 and the initial value of glucose-6-phosphate to 0.

This should result in the model in Figure 5.19.

Every metabolic reaction can be modeled in a similar way by putting an entity for the substrate S on the left, an entity for the product on the right, a process ◇ between them, and setting the speed v of the reaction according to the Michaelis-Menten equation.

Now the rest of the metabolic pathway will be modeled, where all the values for K_m and $V_{\max}$ are set to 1.0. All reactions except (1), (3), and (10) are *reversible*. However, for simplicity, in this model we do not consider the possibility of the reverse reaction. ● will be used for representing enzymes and ⌬ for reagents

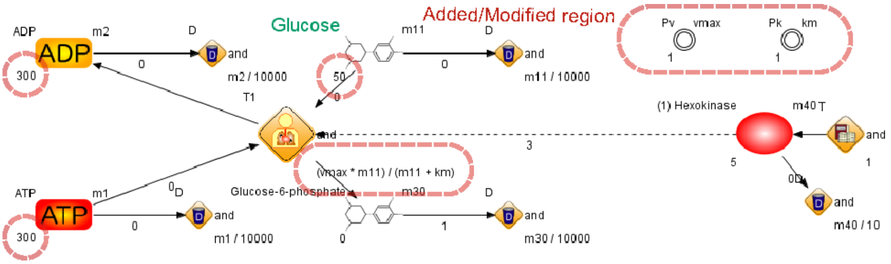


Fig. 5.19

and products. Process will used for representing metabolic reactions. Of course, more detailed custom-made illustrations can be used by users.

Exercise 5.2.1
Add the following reaction (2) to the model in Figure 5.19.

$$\text{Glucose-6-phosphate} \Longleftrightarrow \text{Fructose-6-phosphate}$$

Then the model should look like Figure 5.20.

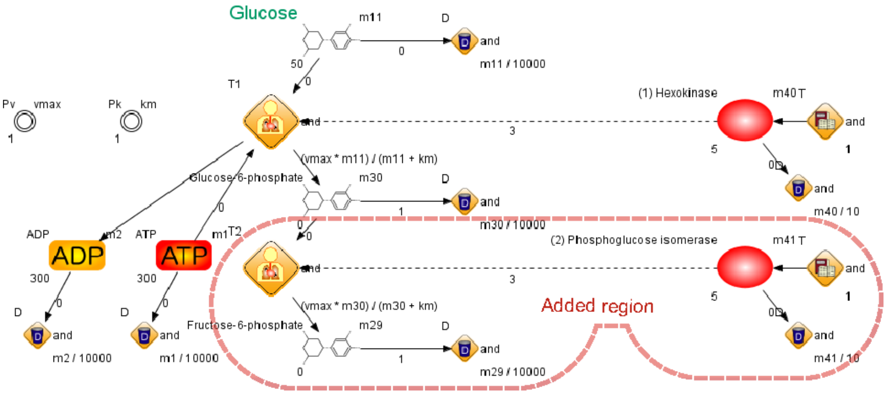


Fig. 5.20

Exercise 5.2.2
Add the following reaction (3) to the model created in Exercise 5.2.1:

$$\text{Fructose-6-phosphate} + \text{ATP} \longrightarrow \text{Fructose-1,6-bisphosphate} + \text{ADP}$$

Then the model should look like Figure 5.21.

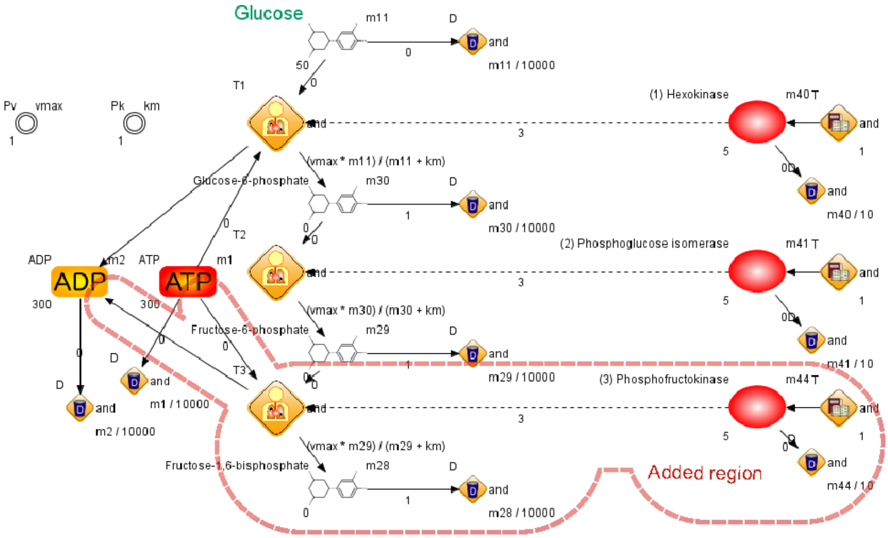


Fig. 5.21

Exercise 5.2.3
Add the following reaction (4) to the model created in Exercise 5.2.2:

$$\text{Fructose-1,6-bisphosphate} \Longleftrightarrow \text{Glyceraldehyde-3-phosphate} + \text{Dihydroxyacetone phosphate}$$

Then the model should look like Figure 5.22.

Exercise 5.2.4
Add the following reaction (5) to the model created in Exercise 5.2.3:

$$\text{Dihydroxyacetone phosphate} \Longleftrightarrow \text{Glyceraldehyde-3-phosphate}$$

Then the model should look like Figure 5.23.

Exercise 5.2.5
Add the following reaction (6) to the model created in Exercise 5.2.4:

$$\text{Glyceraldehyde-3-phosphate} + \text{Phosphate} + \text{NAD}^{+} \longrightarrow \text{1,3-Bisphosphoglycerate} + \text{NADH}$$

Use P , NADH , and NAD+ for phosphate, NADH, and NAD^{+} and add a degradation process D with a speed of 1/10,000th concentration for each entity. Set the initial values of phosphate P and NAD^{+} NAD+ to 300.

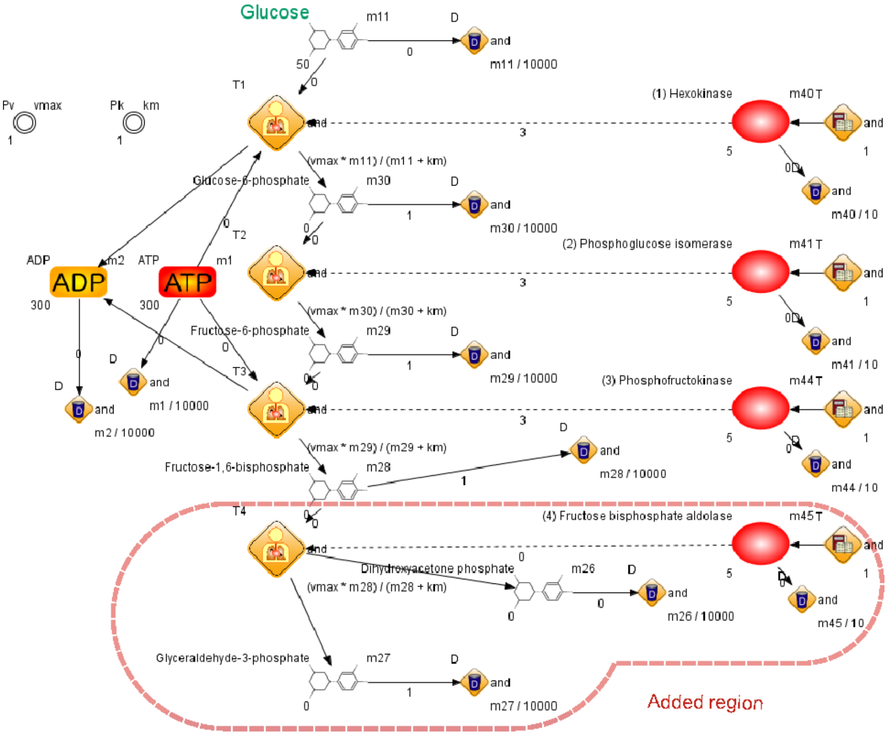


Fig. 5.22

Then the model should look like Figure 5.24.

Exercise 5.2.6
Add the following reaction (7) to the model created in Exercise 5.2.5:

$$\text{1,3-Bisphosphoglycerate} + \text{ADP} \rightleftharpoons \text{3-Phosphoglycerate} + \text{ATP}$$

Then the model should look like Figure 5.25.

Exercise 5.2.7
Add the following reaction (8) to the model created in Exercise 5.2.6:

$$\text{3-Phosphoglycerate} \rightleftharpoons \text{2-Phosphoglycerate}$$

Then the model should look like Figure 5.26.

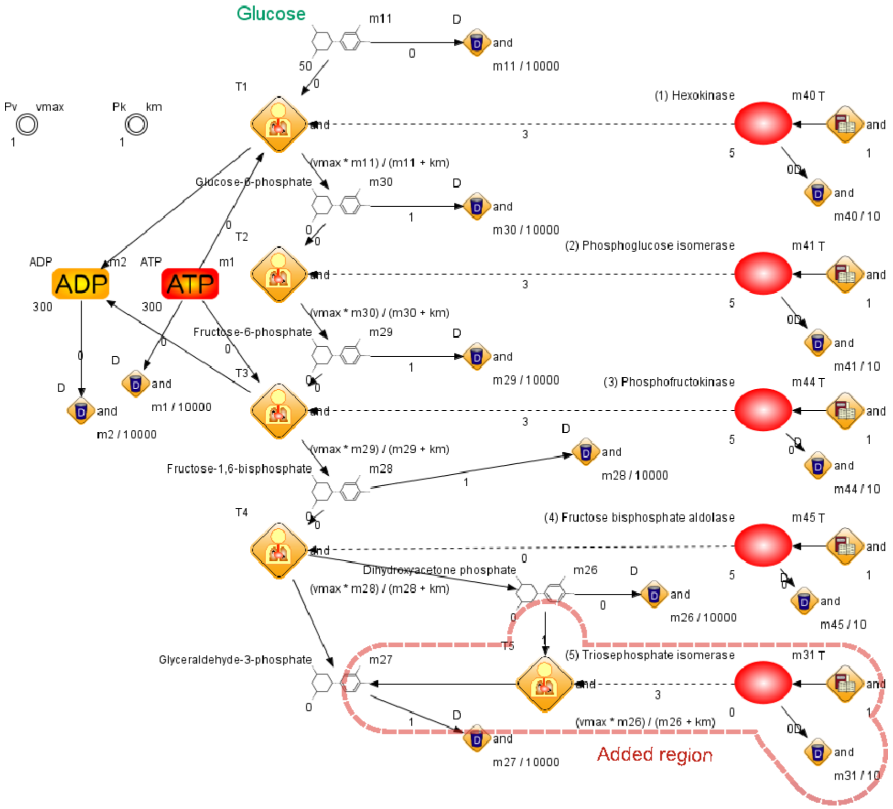


Fig. 5.23

Exercise 5.2.8
Add the following reaction (9) to the model created in Exercise 5.2.7:

$$\text{2-Phosphoglycerate} \Longleftrightarrow \text{Phosphoenolpyruvate}$$

Then the model should look like Figure 5.27.

Exercise 5.2.9
Add the following reaction (10) to the model created in Exercise 5.2.8:

$$\text{Phosphoenolpyruvate} + \text{ADP} \Longleftrightarrow \text{Pyruvate} + \text{ATP}$$

Then the model should look like Figure 5.28 that is a complete model for the metabolic pathway shown in Figure 5.16.

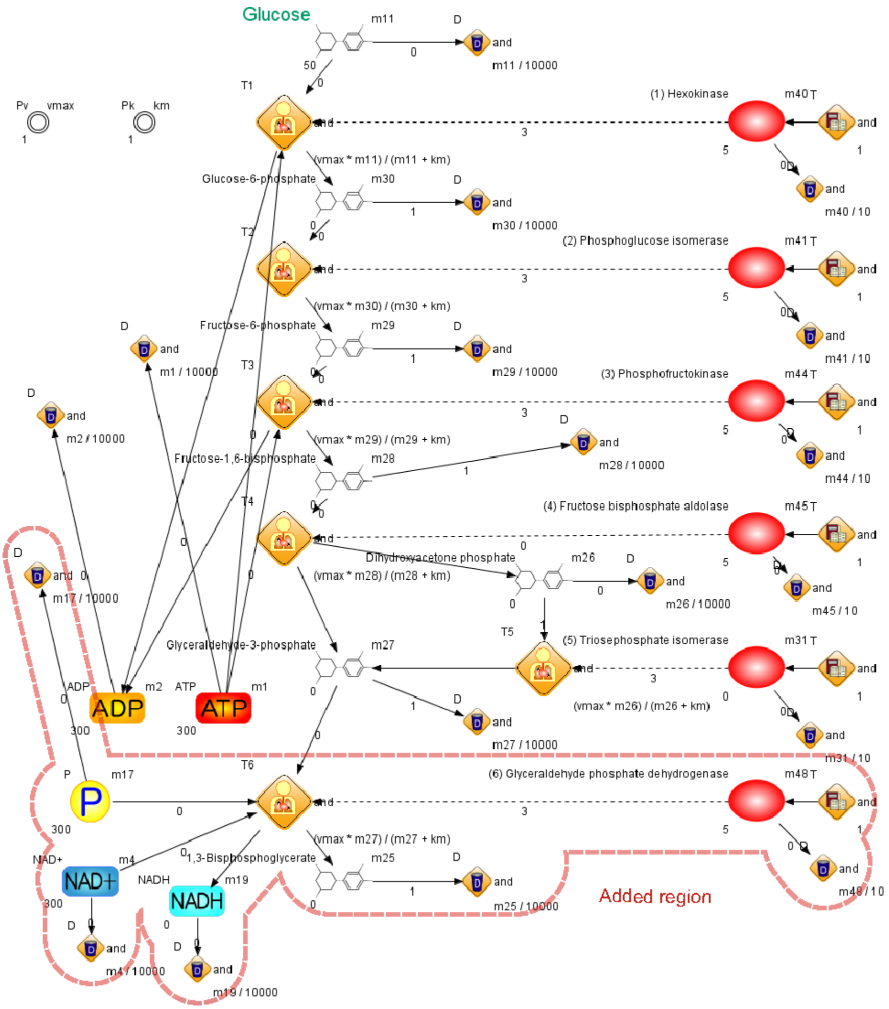


Fig. 5.24

COLUMN 7

Forum for Cell Illustrator

Any questions about the contents of this book are welcome to the Forum for Cell Illustrator located at:

http://www.csml.org/forum/en/

In addition, any questions about CSML and modeling pathways on Cell Illustrator can also be submitted to this forum.

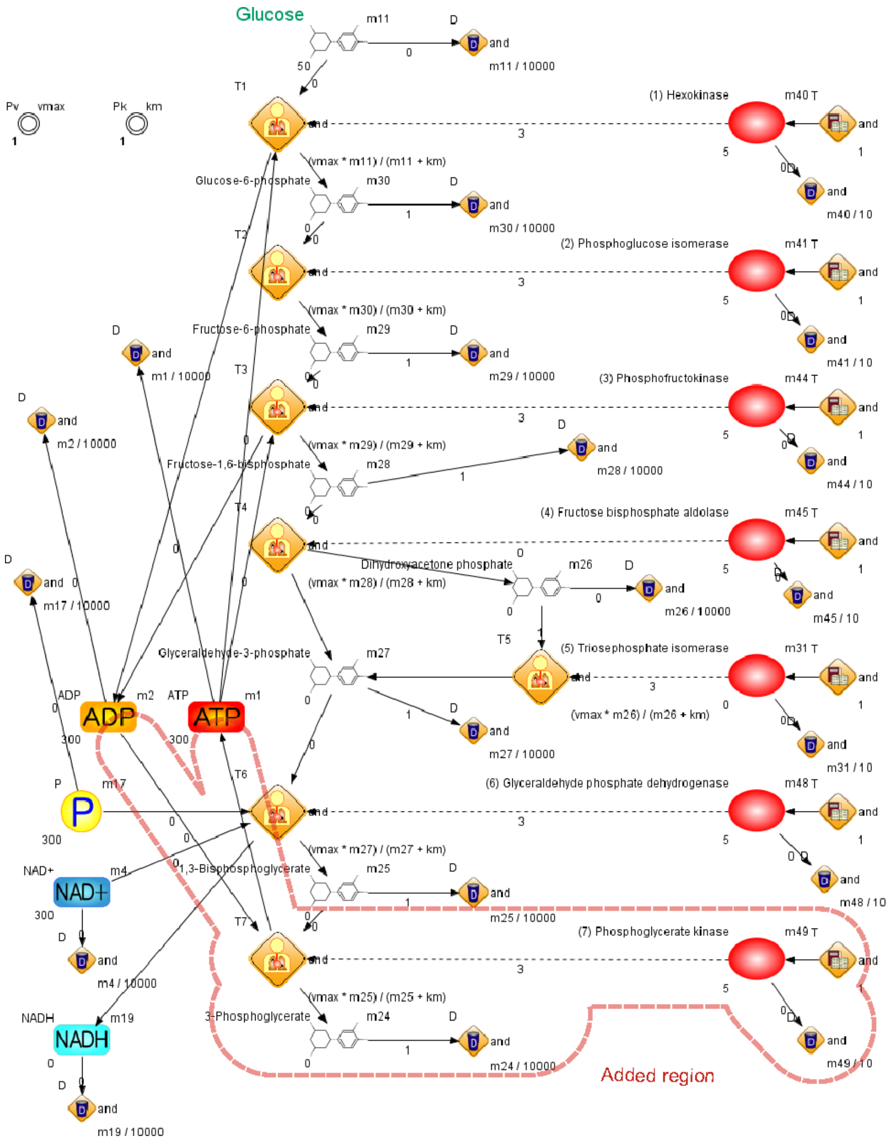

Glucose
m11
D
and
m11 / 10000
50
0
T1
Pv vmax
1
Pk km
1
and
(1) Hexokinase
m40 T
3
5
and
1
0 D
and
m40 / 10
(vmax * m11) / (m11 + km)
Glucose-6-phosphate
m30
D
and
m30 / 10000
1
0
T2
(2) Phosphoglucose isomerase
m41 T
and
m41 / 10
(vmax * m30) / (m30 + km)
Fructose-6-phosphate
m29
D
and
m29 / 10000
m1 / 10000
T3
(3) Phosphofructokinase
m44 T
m44 / 10
m2 / 10000
(vmax * m29) / (m29 + km)
Fructose-1,6-bisphosphate
m28
m28 / 10000
T4
(4) Fructose bisphosphate aldolase
m45 T
m45 / 10
m17 / 10000
Dihydroxyacetone phosphate
(vmax * m28) / (m28 + km)
m26
m26 / 10000
T5
(5) Triosephosphate isomerase
m31 T
m31 / 10
Glyceraldehyde-3-phosphate
m27
(vmax * m26) / (m26 + km)
m27 / 10000
ADP
m2
ATP
m1
ADP
ATP
300
300
T6
P
m17
P
300
(6) Glyceraldehyde phosphate dehydrogenase
m48 T
m48 / 10
(vmax * m27) / (m27 + km)
NAD+
m4
NAD+
300
1,3-Bisphosphoglycerate
m25
m25 / 10000
T7
(7) Phosphoglycerate kinase
m49 T
m4 / 10000
(vmax * m25) / (m25 + km)
NADH
m19
NADH
3-Phosphoglycerate
m24
m24 / 10000
m49 / 10
Added region
m19 / 10000

Fig. 5.25

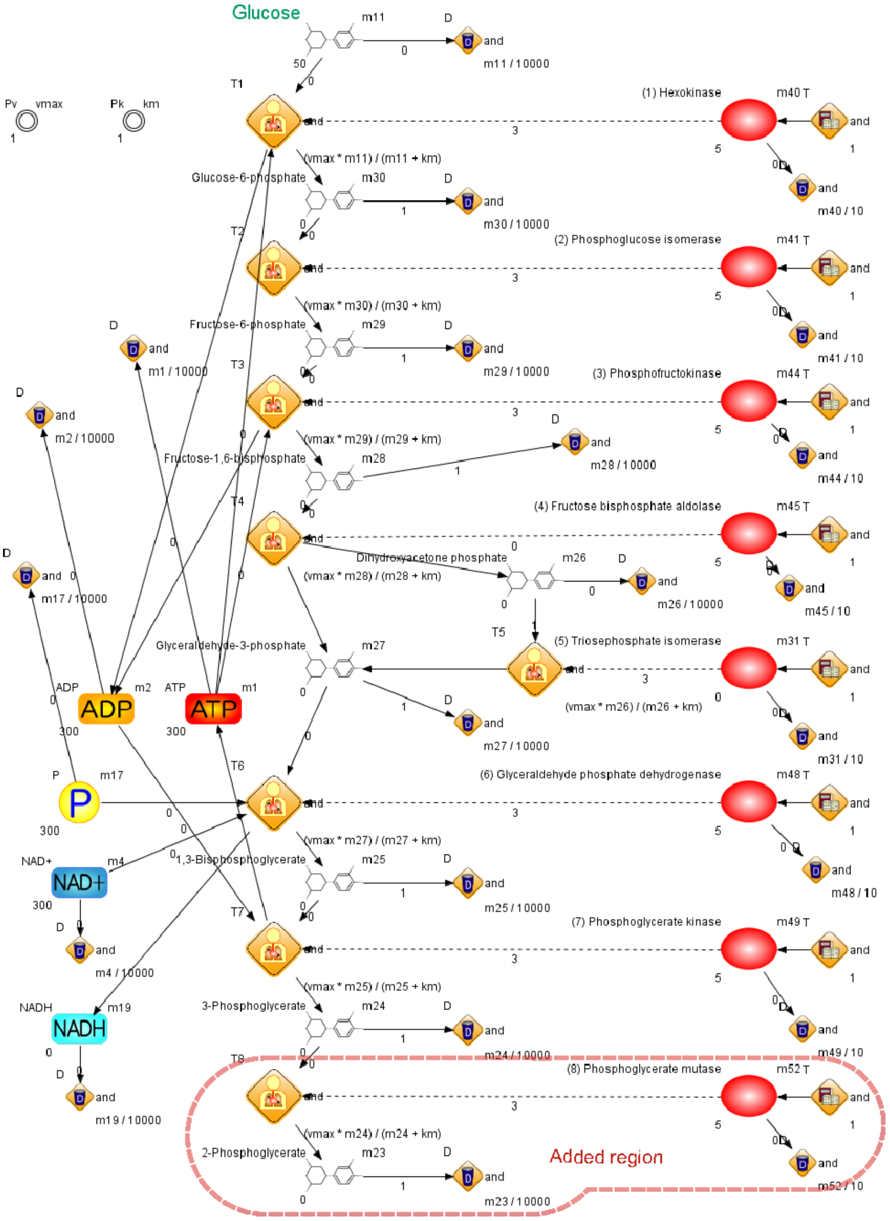

Glucose
m11
D
and
50
0
m11 / 10000
T1
Pv vmax
1
Pk km
1
(1) Hexokinase
m40 T
and
3
5
1
0D
and
m40 / 10
(vmax * m11) / (m11 + km)
Glucose-6-phosphate
m30
D
and
1
m30 / 10000
T2
(2) Phosphoglucose isomerase
m41 T
and
m41 / 10
(vmax * m30) / (m30 + km)
Fructose-6-phosphate
m29
m29 / 10000
D
and
m1 / 10000
T3
(3) Phosphofructokinase
m44 T
m44 / 10
D
and
m2 / 10000
(vmax * m29) / (m29 + km)
Fructose-1,6-bisphosphate
m28
m28 / 10000
T4
(4) Fructose bisphosphate aldolase
m45 T
m45 / 10
D
and
m17 / 10000
Dihydroxyacetone phosphate
(vmax * m28) / (m28 + km)
m26
m26 / 10000
T5
(5) Triosephosphate isomerase
m31 T
m31 / 10
(vmax * m26) / (m26 + km)
Glyceraldehyde-3-phosphate
m27
m27 / 10000
ADP
m2
ADP
300
ATP
m1
ATP
300
T6
(6) Glyceraldehyde phosphate dehydrogenase
m48 T
m48 / 10
P
m17
P
300
NAD+
m4
NAD+
300
1,3-Bisphosphoglycerate
(vmax * m27) / (m27 + km)
m25
m25 / 10000
m4 / 10000
T7
(7) Phosphoglycerate kinase
m49 T
m49 / 10
NADH
m19
NADH
m19 / 10000
(vmax * m25) / (m25 + km)
3-Phosphoglycerate
m24
m24 / 10000
T8
(8) Phosphoglycerate mutase
m52 T
m52 / 10
(vmax * m24) / (m24 + km)
2-Phosphoglycerate
m23
m23 / 10000
Added region

Fig. 5.26

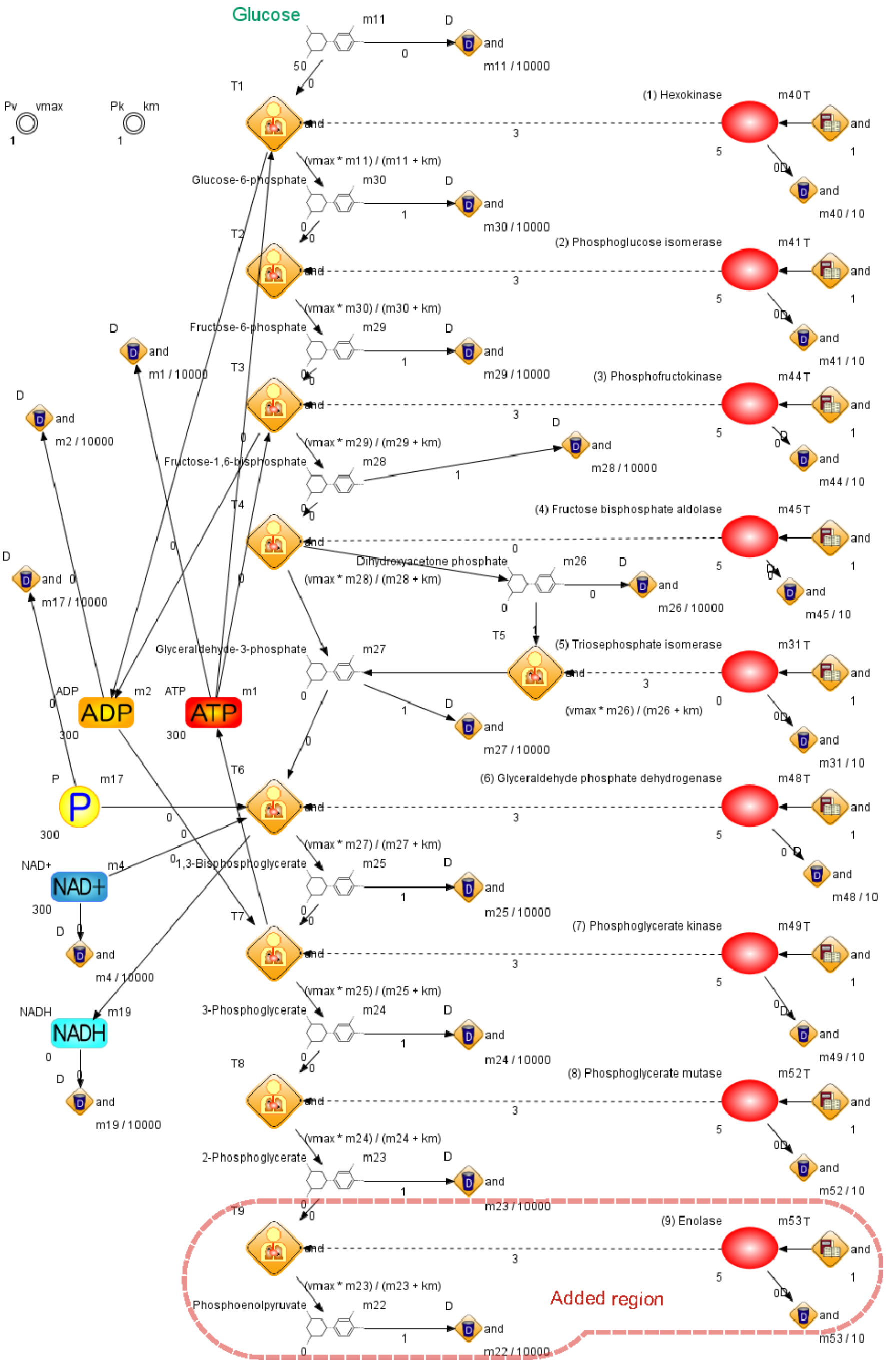

Glucose
m11
D
and
0
m11 / 10000
50
0
T1
Pv vmax
1
Pk km
1
(1) Hexokinase
m40 T
and
3
5
1
0D
and
m40 / 10
(vmax * m11) / (m11 + km)
Glucose-6-phosphate
m30
D
and
1
m30 / 10000
T2
(2) Phosphoglucose isomerase
m41 T
and
m41 / 10
(vmax * m30) / (m30 + km)
Fructose-6-phosphate
m29
and
m29 / 10000
D
and
m1 / 10000
T3
(3) Phosphofructokinase
m44 T
and
m44 / 10
D
and
m2 / 10000
(vmax * m29) / (m29 + km)
Fructose-1,6-bisphosphate
m28
D
and
m28 / 10000
T4
(4) Fructose bisphosphate aldolase
m45 T
and
m45 / 10
D
and
m17 / 10000
Dihydroxyacetone phosphate
(vmax * m28) / (m28 + km)
m26
D
and
m26 / 10000
T5
(5) Triosephosphate isomerase
m31 T
and
m31 / 10
(vmax * m26) / (m26 + km)
Glyceraldehyde-3-phosphate
m27
ADP
m2
ATP
m1
ADP
ATP
300
300
D
and
m27 / 10000
T6
(6) Glyceraldehyde phosphate dehydrogenase
m48 T
and
m48 / 10
P
m17
P
300
(vmax * m27) / (m27 + km)
NAD+
m4
NAD+
300
1,3-Bisphosphoglycerate
m25
D
and
m25 / 10000
T7
(7) Phosphoglycerate kinase
m49 T
and
m49 / 10
D
and
m4 / 10000
(vmax * m25) / (m25 + km)
NADH
m19
NADH
3-Phosphoglycerate
m24
D
and
m24 / 10000
T8
(8) Phosphoglycerate mutase
m52 T
and
m52 / 10
D
and
m19 / 10000
(vmax * m24) / (m24 + km)
2-Phosphoglycerate
m23
D
and
m23 / 10000
T9
(9) Enolase
m53 T
and
m53 / 10
(vmax * m23) / (m23 + km)
Added region
Phosphoenolpyruvate
m22
D
and
m22 / 10000

Fig. 5.27

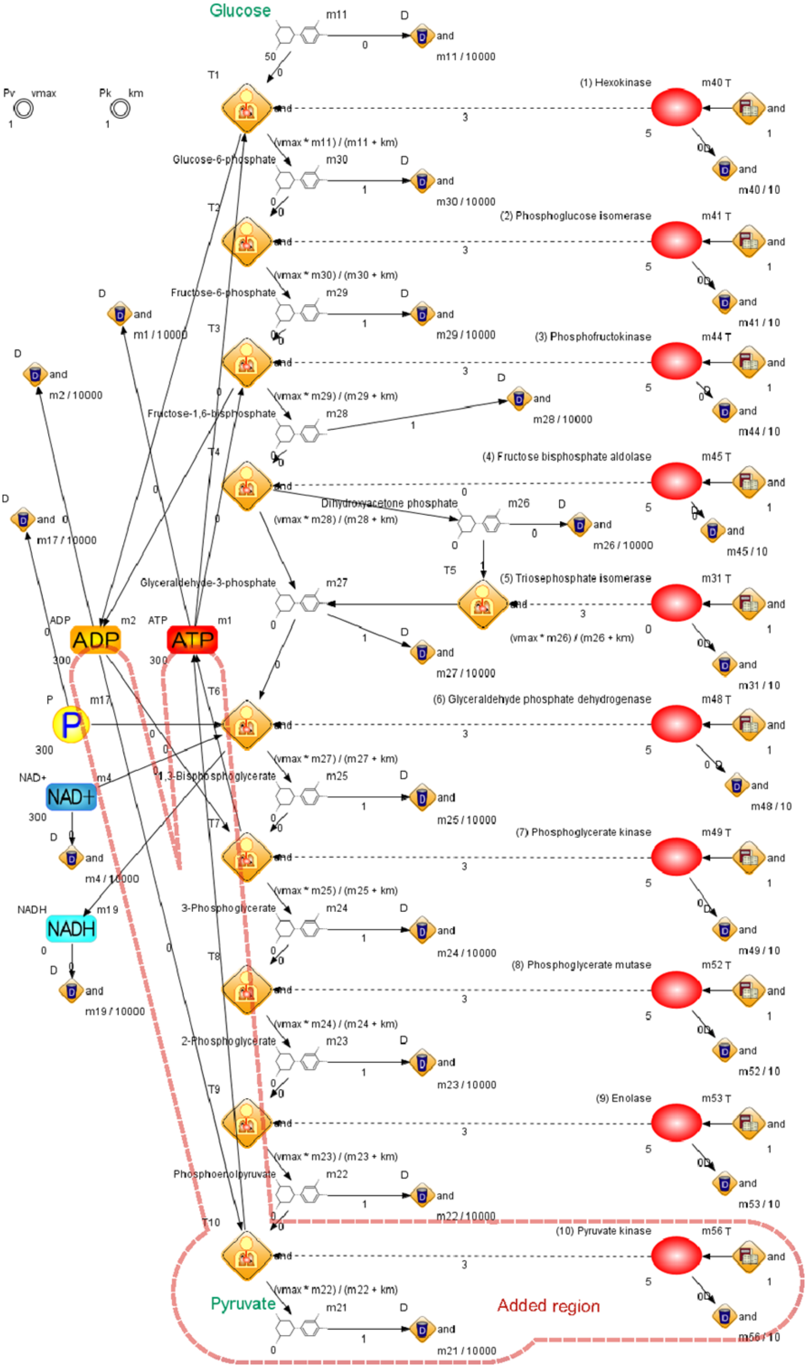

Glucose
m11
D
and
m11 / 10000
50
0
T1
Pv vmax
1
Pk km
1
(1) Hexokinase
m40 T
and
3
5
1
m40 / 10
(vmax * m11) / (m11 + km)
Glucose-6-phosphate
m30
m30 / 10000
T2
(2) Phosphoglucose isomerase
m41 T
m41 / 10
(vmax * m30) / (m30 + km)
Fructose-6-phosphate
m29
m29 / 10000
m1 / 10000
T3
(3) Phosphofructokinase
m44 T
m44 / 10
m2 / 10000
(vmax * m29) / (m29 + km)
Fructose-1,6-bisphosphate
m28
m28 / 10000
T4
(4) Fructose bisphosphate aldolase
m45 T
m45 / 10
Dihydroxyacetone phosphate
m26
(vmax * m28) / (m28 + km)
m26 / 10000
m17 / 10000
T5
(5) Triosephosphate isomerase
m31 T
m31 / 10
Glyceraldehyde-3-phosphate
m27
(vmax * m26) / (m26 + km)
m27 / 10000
ADP
m2
ATP
m1
300
P
m17
T6
(6) Glyceraldehyde phosphate dehydrogenase
m48 T
m48 / 10
NAD+
m4
1,3-Bisphosphoglycerate
(vmax * m27) / (m27 + km)
m25
m25 / 10000
m4 / 10000
T7
(7) Phosphoglycerate kinase
m49 T
m49 / 10
NADH
m19
(vmax * m25) / (m25 + km)
3-Phosphoglycerate
m24
m24 / 10000
m19 / 10000
T8
(8) Phosphoglycerate mutase
m52 T
m52 / 10
(vmax * m24) / (m24 + km)
2-Phosphoglycerate
m23
m23 / 10000
T9
(9) Enolase
m53 T
m53 / 10
(vmax * m23) / (m23 + km)
Phosphoenolpyruvate
m22
m22 / 10000
T10
(10) Pyruvate kinase
m56 T
m56 / 10
(vmax * m22) / (m22 + km)
Pyruvate
m21
m21 / 10000
Added region

Fig. 5.28

5.2.4 Simulation of Glycolysis Pathway

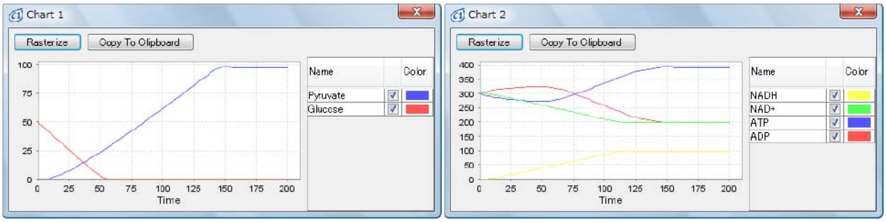
The result of plotting the graph of the concentrations for glucose, pyruvic acid, ADP, NAD$^+$, and NADH is given in Figure 5.29 (in this case, the time 200[pt] is appropriate for simulation).

As mentioned at the beginning of Section 5.2, the model can be summarized in the following reaction:

$$\text{Glucose} + 2\text{ADP} + 2\text{NADH} \longrightarrow 2\text{Pyruvic} + 2\text{ATP} + 2\text{NAD}^+$$

Figure 5.29 shows about 2 times the pyruvic acid as the glucose consumed.

By building such a detailed model, it becomes easier to capture small differences in pathways which a simple equation cannot. This allows us to understand the states of intermediate products as well as the states of abnormality in intermediate steps.



Name	Color
Pyruvate	☑ 🟦
Glucose	☑ 🟥

Name	Color
NADH	☑ 🟨
NAD+	☑ 🟩
ATP	☑ 🟦
ADP	☑ 🟥

Fig. 5.29

5.2.5 Improving the Model

The model will be improved with a few small events that serve to make it more realistic.

A real organism uses ATP in a variety of activities, creating excess ADP, while various reactions create NAD$^+$ from NADH. There are some cases in which ADP and NAD$^+$ are naturally produced.

Figure 5.30 shows a model with these improvements. In this model a set amount of glucose (50 here) is added at a constant interval (400[pt] here) to simulate eating. Since this model adds a periodic event, the simulation time is increased (to 2000[pt]), and the "ChartUpdateInterval" is increased (above 2[pt]). The result of simulating this model is shown in Figure 5.31.

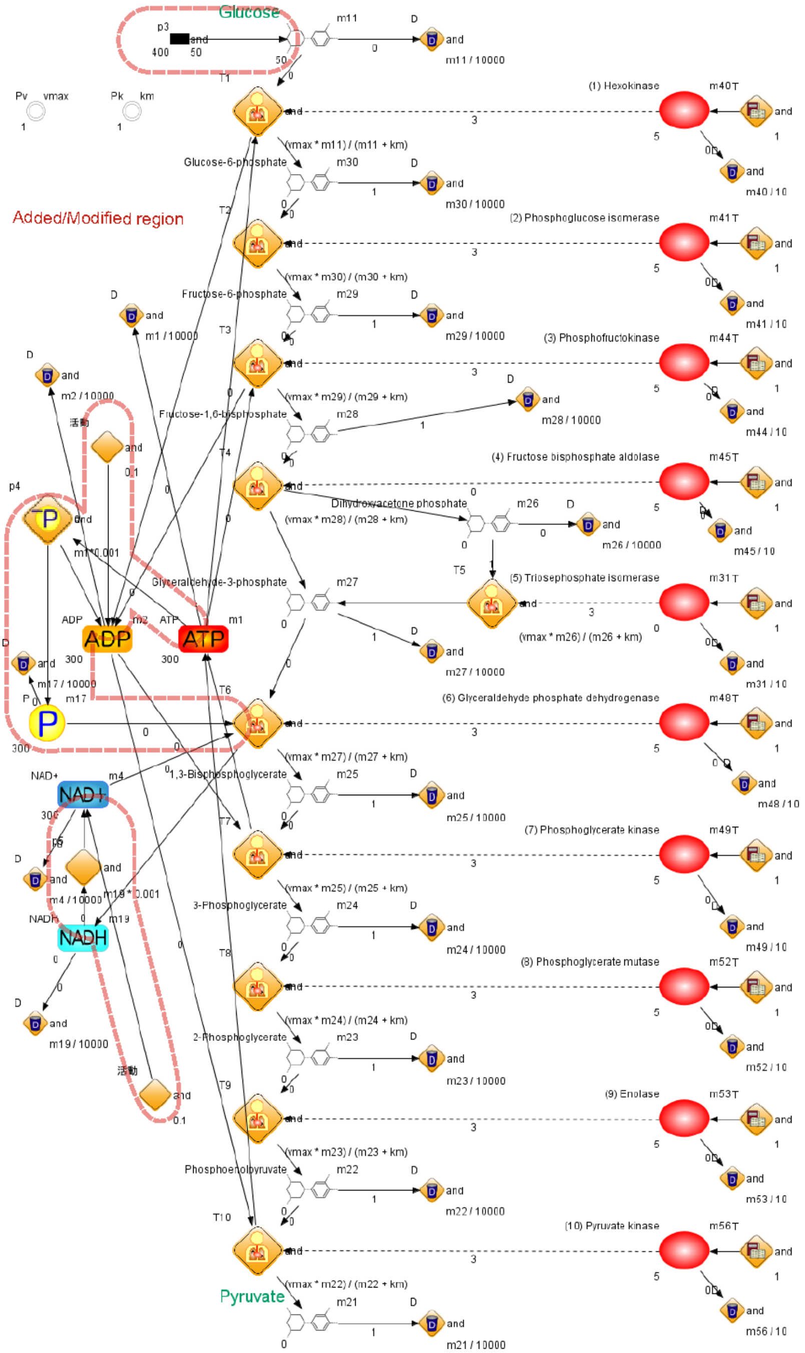

Glucose
p3
400
50
and
50
m11
D
0
and
m11 / 10000
T1
0
Pv vmax
1
Pk km
1
(1) Hexokinase
m40T
and
3
5
1
0D
and
m40 / 10
(vmax * m11) / (m11 + km)
Glucose-6-phosphate
m30
D
1
and
m30 / 10000
Added/Modified region
T2
(2) Phosphoglucose isomerase
m41T
and
m41 / 10
(vmax * m30) / (m30 + km)
Fructose-6-phosphate
m29
and
m29 / 10000
D
and
m1 / 10000
T3
(3) Phosphofructokinase
m44T
and
m44 / 10
and
m2 / 10000
(vmax * m29) / (m29 + km)
Fructose-1,6-bisphosphate
m28
D
and
m28 / 10000
活動
and
0.1
p4
P
and
m1*0.001
T4
(4) Fructose bisphosphate aldolase
m45T
and
m45 / 10
Dihydroxyacetone phosphate
(vmax * m28) / (m28 + km)
m26
D
and
m26 / 10000
T5
(5) Triosephosphate isomerase
m31T
and
m31 / 10
(vmax * m26) / (m26 + km)
Glyceraldehyde-3-phosphate
m27
ADP
m2
ATP
m1
ADP
ATP
300
300
D
and
m27 / 10000
and
m17 / 10000
m17
P
300
T6
(6) Glyceraldehyde phosphate dehydrogenase
m48T
and
m48 / 10
(vmax * m27) / (m27 + km)
1,3-Bisphosphoglycerate
m25
and
m25 / 10000
NAD+
m4
NAD+
300
p5
and
m4 / 10000
m19 * 0.001
NADH
m19
NADH
T7
(7) Phosphoglycerate kinase
m49T
and
m49 / 10
(vmax * m25) / (m25 + km)
3-Phosphoglycerate
m24
and
m24 / 10000
T8
(8) Phosphoglycerate mutase
m52T
and
m52 / 10
and
m19 / 10000
(vmax * m24) / (m24 + km)
2-Phosphoglycerate
m23
and
m23 / 10000
活動
and
0.1
T9
(9) Enolase
m53T
and
m53 / 10
(vmax * m23) / (m23 + km)
Phosphoenolpyruvate
m22
and
m22 / 10000
T10
(10) Pyruvate kinase
m56T
and
m56 / 10
(vmax * m22) / (m22 + km)
Pyruvate
m21
and
m21 / 10000

Fig. 5.30

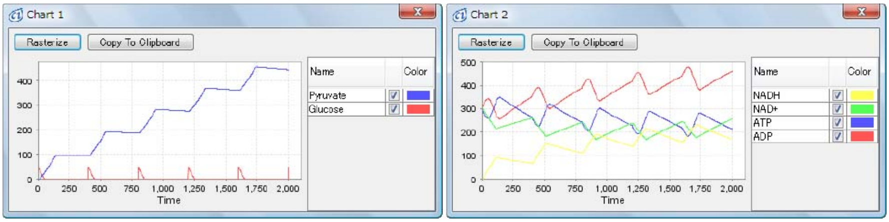


Fig. 5.31

COLUMN 8

Kinetic Styles in Cell Illustrator 3.0

In Cell Illustrator, seven types of *kinetic styles* can be set: *mass*, *stochasticmass* (*stochastic mass*), *stochasticlognormalmass* (*stochastic log normal mass*), *michaelismenten* (*Michaelis-Menten*), *connectorrate* (*connector rate*), *connectorcustom* (*connector custom*), *custom*. Below are the description and parameters for each type. In the case that an entity is connected to a process through an input connector, the rate of decrease is called *consumption speed*. In the case of the entity connected by the output connector, this is called *production speed*. In the following explanations, we assume a process p1 with one output connector c0 and *k* input connectors c1,...,c*k* and the variables for these entities are m0 to m*k*. The notation c*i* indicates the *i*th connector.

Usually, simple models can be made using only "mass" and "michaelis menten" kinetic styles. For an example of kinetic style use, see

Models\Chapter5\kineticStyle.xml

in the CD-ROM.

mass

Table 5.2 shows the details of "mass" which is a standard reaction depending upon the mass of entities concerned.

stochasticmass

This is similar to "mass", but instead includes Gaussian noise as part of reaction rate (Table 5.3).

Table 5.2

Kinetic equation	$\mathrm{mass}(a, b, \mathrm{m}1, \ldots, \mathrm{m}k) = a \cdot b^k \cdot \mathrm{m}1 \cdots \mathrm{m}k$	
Name	Coefficient Function	Default Value
Coefficient1 $(= a)$	One of the coefficients which determine reaction rate, the larger the value the faster the reaction.	0.1
Coefficient2 $(= b)$	One of the coefficients which determine reaction rate, the number of entities are linked.	1.0
ci Stoichiometry $(= \mathrm{s}i)$	i specifies a connector. If the given ci is an input connector, then its consumption rate becomes $\mathrm{mass}(a, b, \mathrm{m}1, \ldots, \mathrm{m}k) \cdot \mathrm{s}i$. If c$i$ is an output connector, then si determines production rate.	1.0

Table 5.3

Kinetic equation	$\mathrm{smass}(a, b, \mathrm{m}1, \ldots, \mathrm{m}k) = a \cdot b^k \cdot \mathrm{m}1 \cdots \mathrm{m}k + \varepsilon$ with $\varepsilon \sim N(0, \sigma^2)$	
Name	Coefficient Function	Default Value
Coefficient1 $(= a)$	One of the coefficients which determine reaction rate, the larger the value the faster the reaction.	0.1
Coefficient2 $(= b)$	One of the coefficients which determine reaction rate, the number of entities are linked.	1.0
ci Stoichiometry $(= \mathrm{s}i)$	ci specifies a connector. If the given ci is an input connector, then its consumption rate becomes $\mathrm{smass}(a, b, \mathrm{m}1, \ldots, \mathrm{m}k) \cdot \mathrm{s}i$. If c$i$ is an output connector, then si determines production rate.	1.0
Standard deviation $(= \sigma)$	Specifies the deviation in reaction rate; setting this parameter to 0 results in the same output as mass.	1.0

stochasticlognormalmass

This is similar to “mass” but includes lognormal noise (Table 5.4).

Table 5.4

Kinetic equation	$\mathrm{slogmass}(a, b, \mathrm{m}1, \ldots, \mathrm{m}k) = a \cdot b^k \cdot \mathrm{m}1 \cdots \mathrm{m}k \cdot e^{\varepsilon}$, $\varepsilon \sim N(0, \sigma^2)$	
Name	Coefficient Function	Default Value
Coefficient1 $(= a)$	One of the coefficients which determine reaction rate, the larger the value the faster the reaction.	0.1
Coefficient2 $(= b)$	One of the coefficients which determine reaction rate, the number of entities are linked.	1.0
ci Stoichiometry $(= \mathrm{s}i)$	ci specifies a connector. If the given ci is an input connector, then its consumption rate becomes $\mathrm{slogmass}(a, b, \mathrm{m}1, \ldots, \mathrm{m}k) \cdot \mathrm{s}i$. If c$i$ is an output connector, then si determines production rate.	1.0
Standard deviation $(= \sigma)$	Specifies the deviation in reaction rate; setting this parameter to 0 results in the same output as mass.	1.0

michaelismenten

This is the often-used Michaelis-Menten reaction equation. The K_m and $V_{\max}$ processes need to be set (Table 5.5). This process requires at least one enzyme with an association connector.

Table 5.5

Name	Coefficient Function	Default Value
K_m	Set K_m for the Michaelis-Menten equation.	0.1
$V_{\max}$	Set $V_{\max}$ for the Michaelis-Menten equation.	1.0

connectorrate

This kinetic style is used if the rate of consumption and production for all elements are proportional to some speed (Table 5.6).

Table 5.6

Name	Coefficient Function	Default Value
Rate ($= r$)	Sets the base consumption/production rate for all connectors.	0.1
ci Stoichiometry ($=$ si)	ci specifies a connector. If the given ci is an input connector, then its consumption rate becomes s$i \cdot r$. If ci is an output connector, then si determines production rate.	1.0

connectorcustom

This is used when the rates of consumption and production are varied (Table 5.7).

Table 5.7

Name	Coefficient Function	Default Value
ci Stoichiometry ($=$si)	ci specifies a connector. If the given ci is an input connector, then its consumption rate becomes si. If ci is an output connector, then si determines production rate.	0

custom

This is used if production and consumption rate for all entities can be shown in one equation (Table 5.8).

Table 5.8

Name	Coefficient Function	Default Value
Speed	Sets the rate of consumption/production for all elements.	1.0

5.3 Modeling Gene Regulatory Networks

As an example of a gene regulatory network, we model the system producing circadian rhythms in *Mus musculus* (mouse).

5.3.1 Biological Clocks and Circadian Rhythms

Living organisms are constantly aware of time. This is said to be because of an internal biological clock. Within the biological clock, there are various rhythms such as the circadian (24 hours), circatrigintan (30 days), circannual (1 year). These rhythms are often called tide cycle, daily cycle, monthly cycle, and annual cycles and thought to synchronize to various external rhythms.

This chapter will specifically cover the circadian rhythm and the gene regulatory network which produces the rhythms. The term *circadian* is formed from the Latin *circa* (around) and *dies* (1 day) and refers to any daily cycle.

Homo sapiens (human beings) normally operate around a 24-hour day. It is known that living in a dark room with no way to tell the time, the interval between eating and sleeping begins to widen and settles to a 25-hour cycle. This shows the existence of an internal clock mechanism. This 25-hour cycle is known as the *free run rhythm*. These free run rhythms vary from species to species. For humans, this is 25 hours, for rats 24.5 hours, and for mice 23.5 hours. The difference in cycles between species implies a genetic basis for these cycles.

In reality, however, the operating cycle is 24 hours and thus the free run rhythms are synchronized to the daily cycle of 24 hours. Although there are many environmental factors which cause synchronization, the strongest factor is considered to be the sunlight.

5.3.2 Gene Regulatory Network for Circadian Rhythms in Mice

Figure 5.32 shows the mechanism of the gene regulatory network for circadian rhythms in mice as currently known. The basic structure of this mechanism is simple.

First the protein BMAL, the product of the gene *Bmal*, binds the protein CLOCK produced by the gene *Clock* and makes a heterodimer. Then this heterodimer is transported to the nucleus and activates the transcription of the genes *Per*, *Cry*, and *Rev-Erb*.

The protein PER produced by the gene *Per* and the protein CRY produced by the gene *Cry* bind together to produce a heterodimer. This heterodimer also moves into the nucleus and inhibits the transcription of *Per*, *Cry*, and *Rev-Erb*. Thus by indirectly regulating both *Per* and *Cry* genes by themselves, they create a *negative feedback loop*.

The *Rev-Erb* gene produces the REV-ERB protein which moves into the nucleus and inhibits the transcription of the gene *Bmal*. Therefore *Bmal* and *Clock* also create a negative feedback loop through *Rev-Erb*. *Rev-Erb* is also inhibited by the PER/CRY heterodimer and thus *Rev-Erb* synchronizes these two feedback loops.

As can be seen, the murine circadian rhythms are made from two negative feedback loops which continually regulate each other.

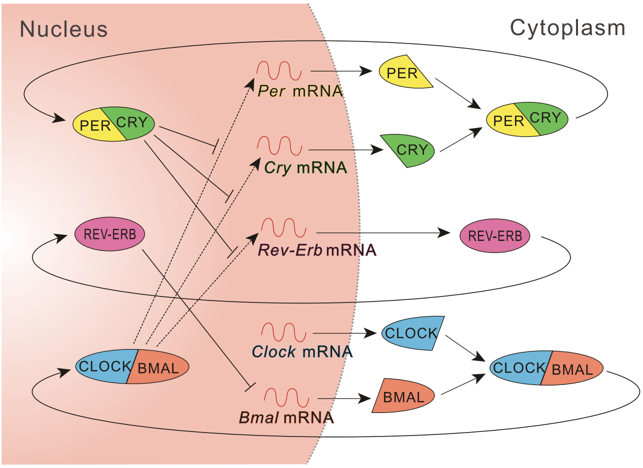


Fig. 5.32

5.3.3 Modeling Circadian Rhythms in Mice

In a similar way to Sections 5.1 and 5.2, we will model the system of murine circadian rhythms with a set of guided instructions and exercises. The goal is to turn Figure 5.32 into a model which can be simulated on Cell Illustrator.

5.3.3.1 Placement of Nucleus and Cytoplasm

In this model, we will deal with the gene regulatory network in the nucleus and cytoplasm. First, place the nucleus and cytoplasm as seen in Figure 5.33. Although the location and size of the nucleus will not affect the simulation, place them as seen in Figure 5.33. This is made easier using the "Edit Parts" → [cells] → [animal_cell_nucleus].

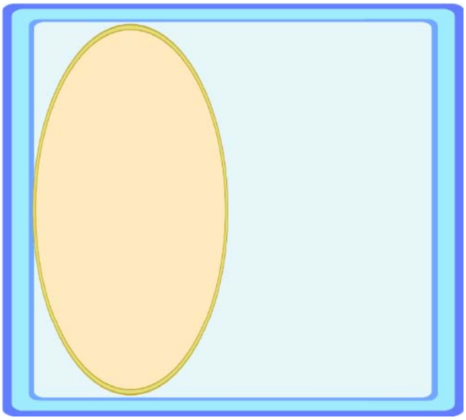


Fig. 5.33

5.3.3.2 Transcription and Translation of *Per*

The transcription and translation of the gene *Per* are known to be involved with the regulation of the murine circadian rhythm.

Exercise 5.3.1
Create a transcription process for *Per* inside the nucleus. Use for the mRNA of *Per* and set the transcription speed to 1.

Exercise 5.3.2
Add a translation process ◆ within the cytoplasm which translates 〰 of *Per*. Use 🔵 for the protein PER.

Exercise 5.3.3
As can be seen from the graph for PER, the translation speed for *Per* does not depend on the amount of the *Per* mRNA. Modify this model so that the *Per* translation speed depends on the concentration of *Per* mRNA by setting the rate to 1/5th of the mRNA concentration 〰 .

Then the result should look like Figure 5.34.

5.3.3.3 Transcription and Translation of *Cry*

In a similar way to *Per*, *Cry* is also transcribed and translated. From here on, entities of various colors will be used. The folder of “PrepareElements” in the CD-ROM has various illustrations for use in the file EntitiesForCircadianModel.xml.

Exercise 5.3.4
Create a process ◆ for *Cry*. Use 〰 for the mRNA of *Cry* and set the transcription speed to 1.

Exercise 5.3.5
Create a translation process ◆ for *Cry*. Use the red protein illustration 🔴 for CRY.

Exercise 5.3.6
This model keeps the *Cry* translation rate constant regardless of the *Cry* mRNA concentration. Modify the model in a similar manner to Exercise 5.3.3 to make the translation speed dependent on the *Cry* mRNA concentration.

Then the model should look like Figure 5.35.

5.3.3.4 Natural Degradation of *Per* and *Cry* mRNAs and PER and CRY

The model up to now contains the transcription and translation processes of *Per* and *Cry*. In this section, we add the processes which degrade their gene products.

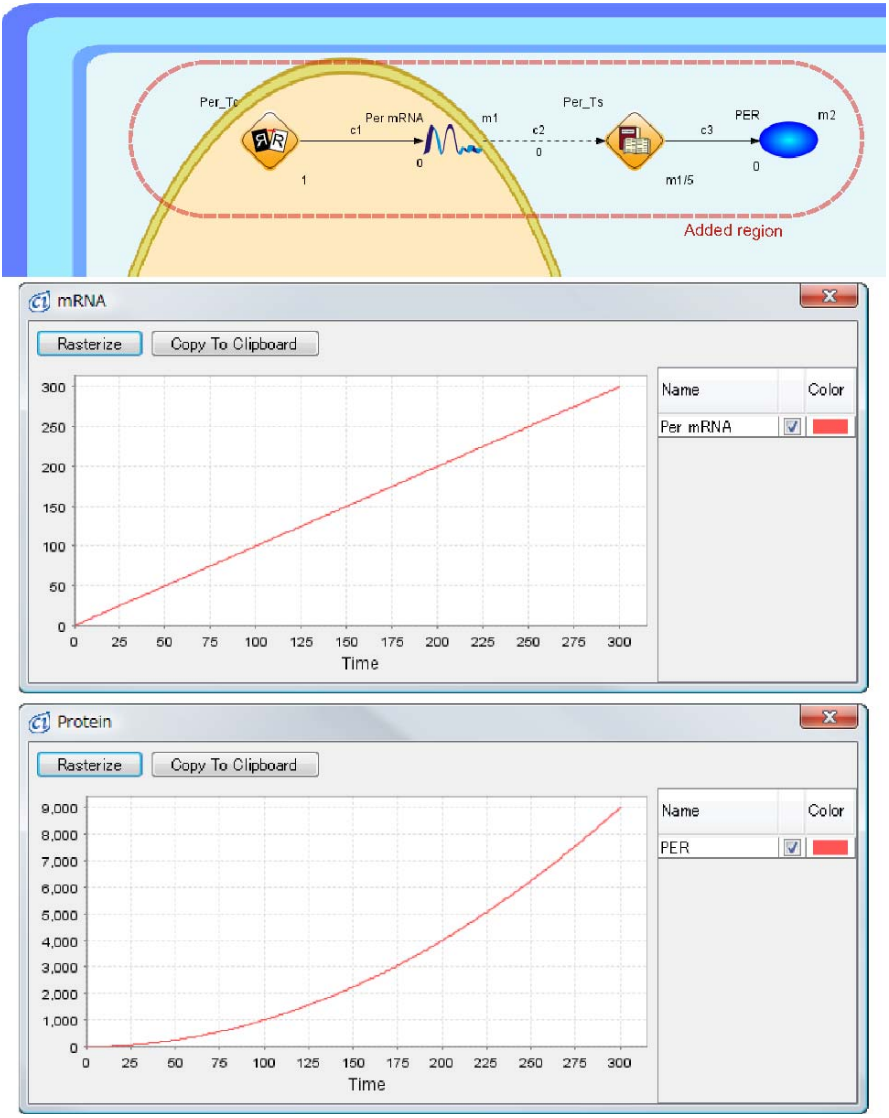
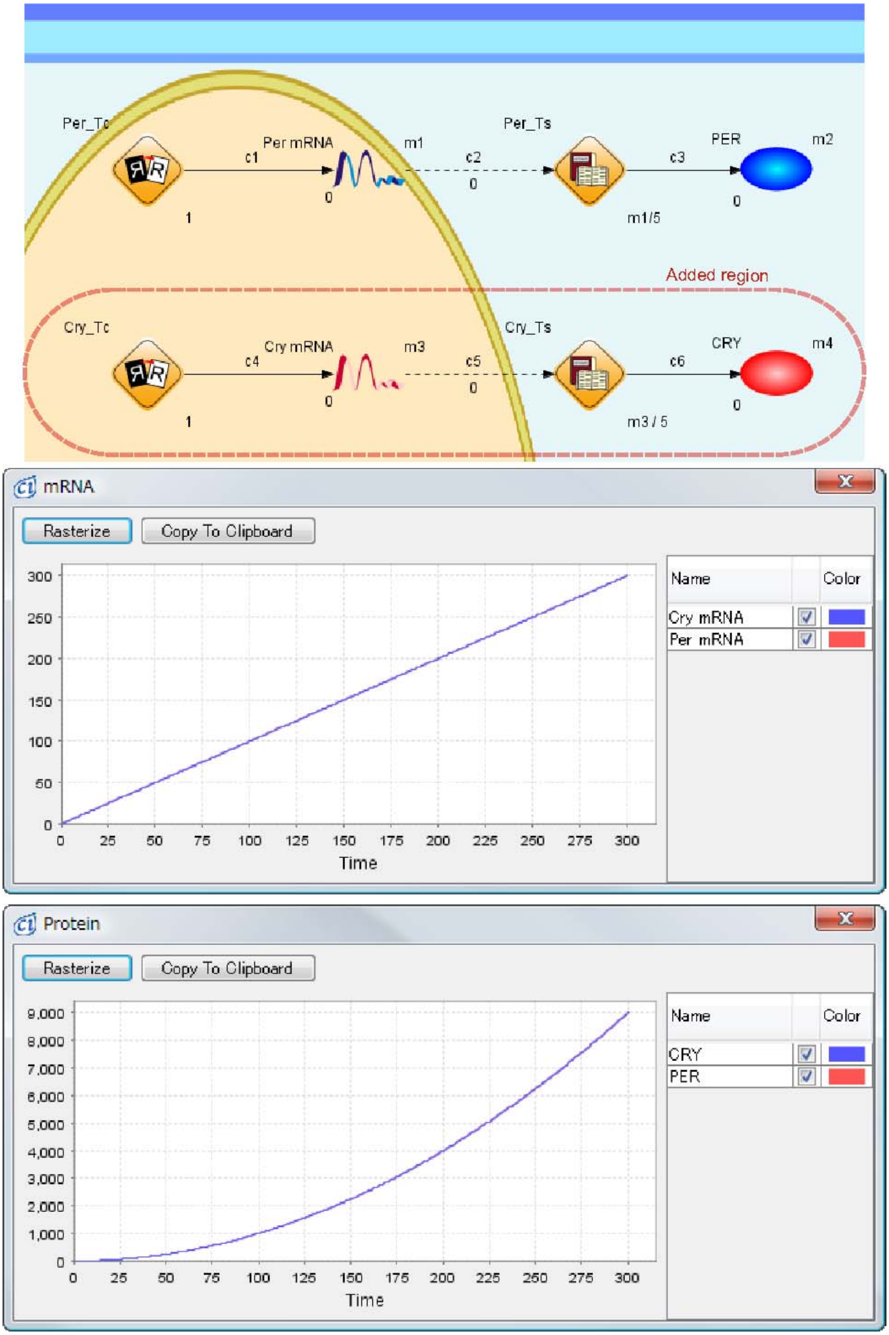

Per_Tc
Per mRNA
c1
m1
Per_Ts
c2
PER
c3
m2
1
0
0
m1/5
0
Added region
mRNA
Rasterize
Copy To Clipboard
300
250
200
150
100
50
0
0 25 50 75 100 125 150 175 200 225 250 275 300
Time
Name
Color
Per mRNA ☑ 🟥
Protein
Rasterize
Copy To Clipboard
9,000
8,000
7,000
6,000
5,000
4,000
3,000
2,000
1,000
0
0 25 50 75 100 125 150 175 200 225 250 275 300
Time
Name
Color
PER ☑ 🟥

Fig. 5.34



Name		Color
Cry mRNA	☑	🟦
Per mRNA	☑	🟥

Name		Color
CRY	☑	🟦
PER	☑	🟥

Fig. 5.35

Exercise 5.3.7
Add degradation processes to *Per* and *Cry* mRNAs and proteins PER and CRY. Use D for the degradation processes and set their speeds so that they depend on the concentrations of entities. Since the proteins are more stable than the mRNAs, set the speed of mRNA degradation to 1/5th of its concentration, and set the speed of protein degradation to 1/10th of its concentration.

Exercise 5.3.8
Simulate the model and confirm that the values for the mRNAs of *Per* and *Cry* and the proteins PER and CRY converge to the steady-state values.

Then the model should look like Figure 5.36.

5.3.3.5 Interaction of Proteins PER and CRY

The proteins PER and CRY are known to bind and create a complex which is more stable than either PER or CRY.

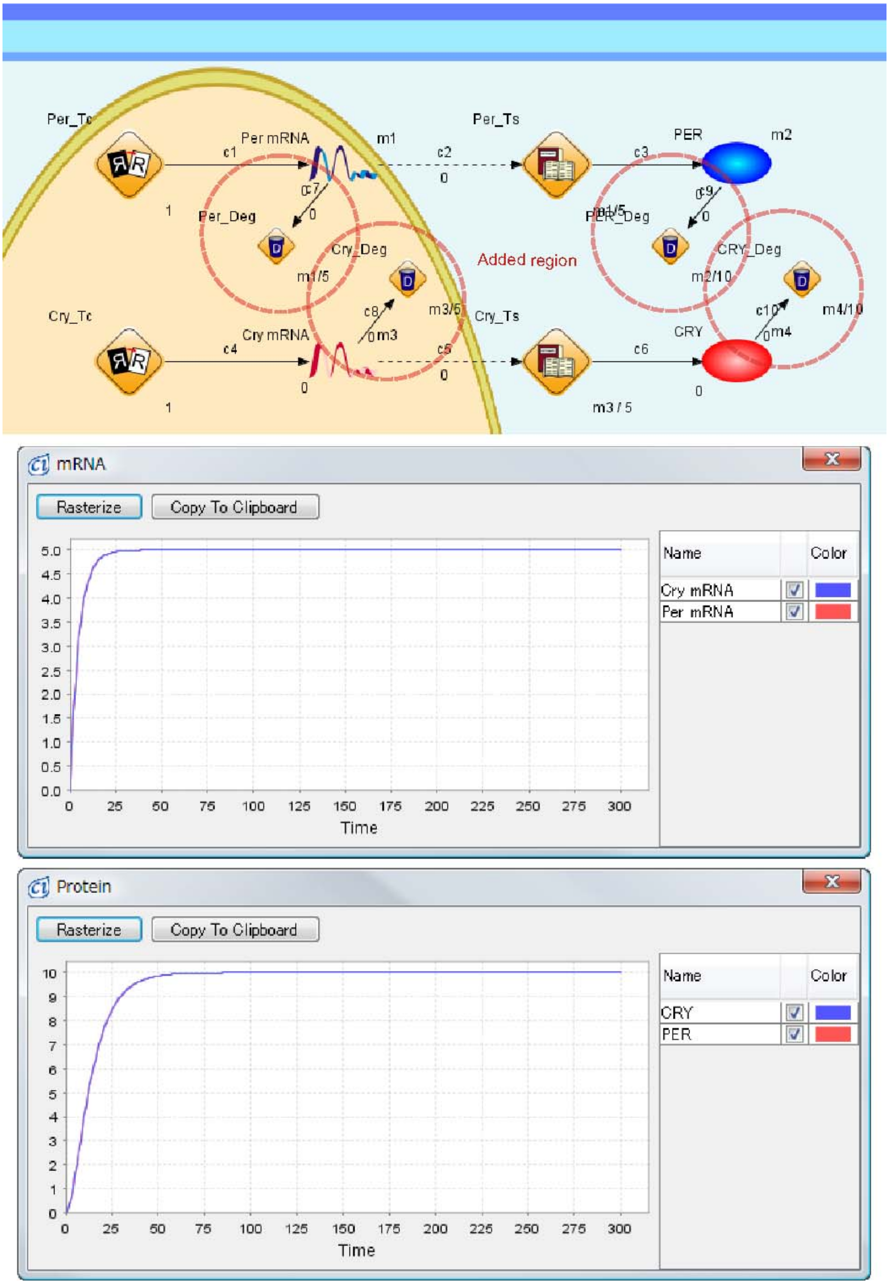
Exercise 5.3.9
Add a process ↪ which forms a complex PER/CRY ● from proteins PER and CRY. Set the speed to 1/10th of the product of the concentrations of PER and CRY. Add also a degradation process D for ● with a speed of 1/15th of its concentration.

Then the model should look like Figure 5.37.

Exercise 5.3.10
The complex PER/CRY ● of PER and CRY in the cytoplasm is known to be transported to the nucleus. Add this translocation process 🚚 to the model. Set the speed of this translocation to 1/10th of the concentration of the complex PER/CRY.

Exercise 5.3.11
Add a degradation process D for the complex PER/CRY ● in the nucleus. Considering the relative stability of PER/CRY, set the degradation speed to 1/15th of the concentration of PER/CRY.

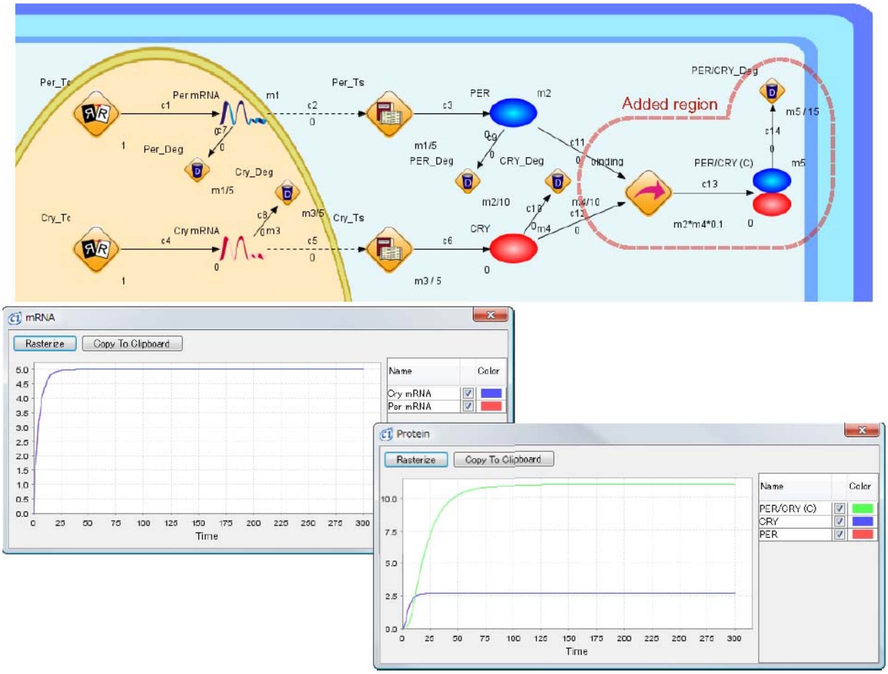
Then the model should look like Figure 5.38.



Name		Color
Cry mRNA	☑	🟦
Per mRNA	☑	🟥

Name		Color
CRY	☑	🟦
PER	☑	🟥

Fig. 5.36



Name	Color
Cry mRNA ☑	🟦
Per mRNA ☑	🟥

Name	Color
PER/CRY (C) ☑	🟩
CRY ☑	🟦
PER ☑	🟥

Fig. 5.37

5.3.3.6 Negative Feedback Loop of Genes *Per* and *Cry*

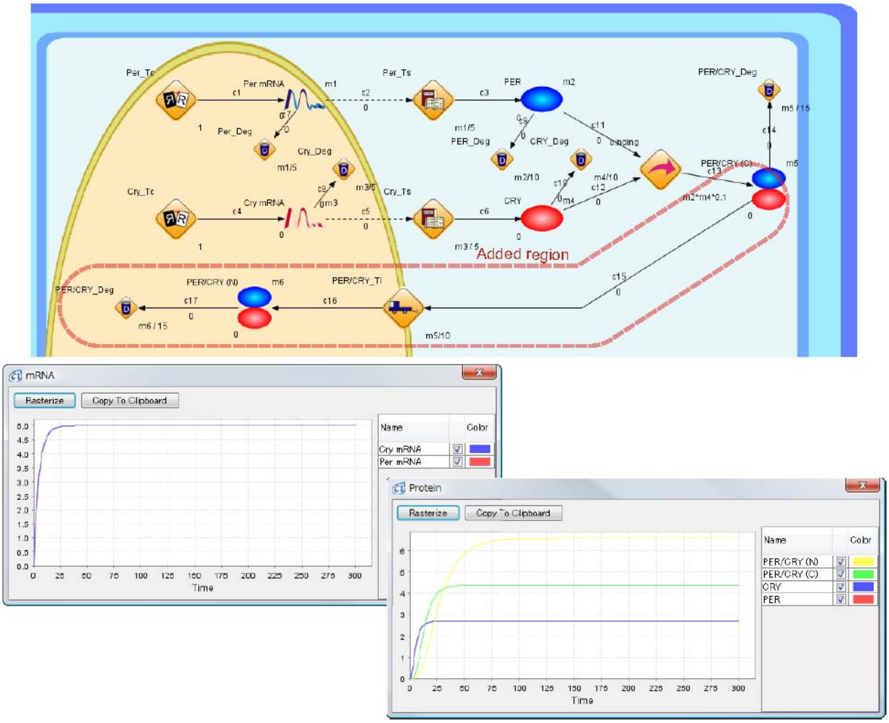
Once in the nucleus, it is known that the complex PER/CRY inhibits the transcription of both *Per* and *Cry*. In other words, the inhibition forms a negative feedback loop.

Exercise 5.3.12
Add inhibitory connectors ⊥ so that the complex PER/CRY in the nucleus inhibits the transcription processes of *Per* and *Cry*. Set the threshold value for each inhibitory connector to 1.5. Simulate and confirm that the behavior of the mRNA of *Per* shows oscillations.

Then the model should look like Figure 5.39.

5.3.3.7 Introducing *Rev-Erb* in the Network

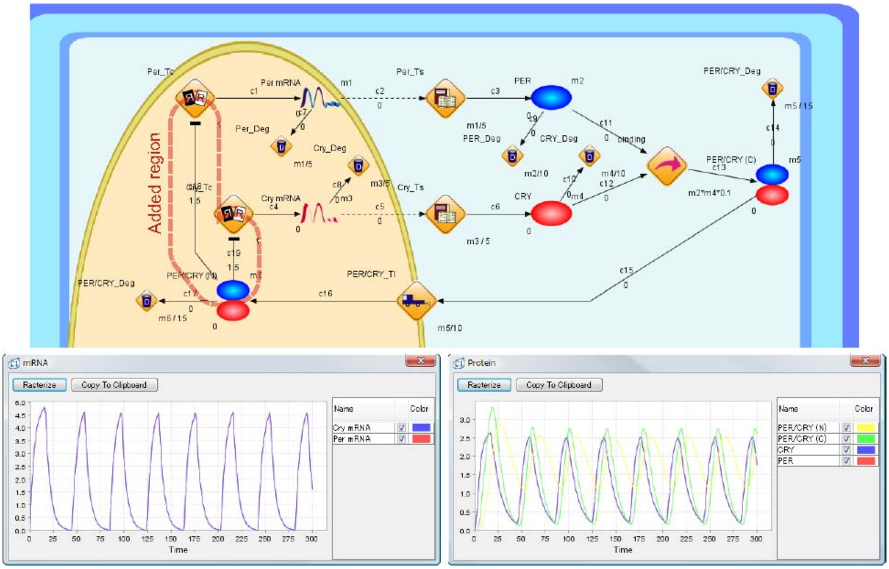
The murine circadian rhythms are regulated by two feedback loops; one made of *Per* and *Cry* and the other made of *Bmal* and *Clock*. The gene *Rev-Erb* plays a role to connect and synchronize these two loops.



Name	Color
Cry mRNA	☑ 🟦
Per mRNA	☑ 🟥

Name	Color
PER/CRY (N)	☑ 🟨
PER/CRY (C)	☑ 🟩
CRY	☑ 🟦
PER	☑ 🟥

Fig. 5.38



Name	Color
Cry mRNA	☑ 🟦
Per mRNA	☑ 🟥

Name	Color
PER/CRY (N)	☑ 🟨
PER/CRY (C)	☑ 🟩
CRY	☑ 🟦
PER	☑ 🟥

Fig. 5.39

Exercise 5.3.13
Add a transcription process ◆ for *Rev-Erb* and use ∿ for its mRNA. Add a translation process ◆ from mRNA ∿ of *Rev-Erb* to the protein REV-ERB 🟢. Add an inhibitory connector so that the complex PER/CRY 🔵🔴 inhibits the transcription of *Rev-Erb*. Set the threshold of this inhibitory connector to 1.4. Set the speed of translation from mRNA ∿ of *Rev-Erb* to 1/10th of its concentration. The speeds of other processes are the same as those considered for the *Cry* pathway.

Exercise 5.3.14
It is known that after translation the protein REV-ERB is transported to the nucleus. Add a translocation process ◆ to the model. Set the speed of translocation of REV-ERB to 1/10th of its concentration. Similarly, add a degradation process for REV-ERB in the nucleus and set the degradation speed to 1/10th of its concentration.

Then the model should look like Figure 5.40.

5.3.3.8 Transcription and Translation of *Bmal* and *Clock*

Similarly to the genes *Per* and *Cry*, the genes *Bmal* and *Clock* form a complex. In addition, *Bmal* is known to be inhibited by the protein REV-ERB in the nucleus.

Exercise 5.3.15
In the same way as *Per* and *Cry*, add a pathway comprised of the processes of transcription, translation, and degradation for the genes *Bmal* and *Clock*. Set their translation speeds to be proportional to the concentrations of mRNAs, and set the degradation speeds proportional to the concentrations (exactly the same as in the case for *Per* and *Cry*). Create an inhibitory connector with the threshold 0.8 from REV-ERB 🟢 in the nucleus to the transcription process ◆ of *Bmal*. Use ∿ for the *Bmal* mRNA, 🟠 for the protein BMAL, ∿ for the *Clock* mRNA, and 🟣 for the protein CLOCK.

Then the model should look like Figure 5.41.

5.3.3.9 Interaction of Proteins BMAL and CLOCK

The proteins BMAL and CLOCK are known to form a complex like the proteins PER and CRY. Similarly, the BMAL/CLOCK complex is known to be more stable than either BMAL or CLOCK.

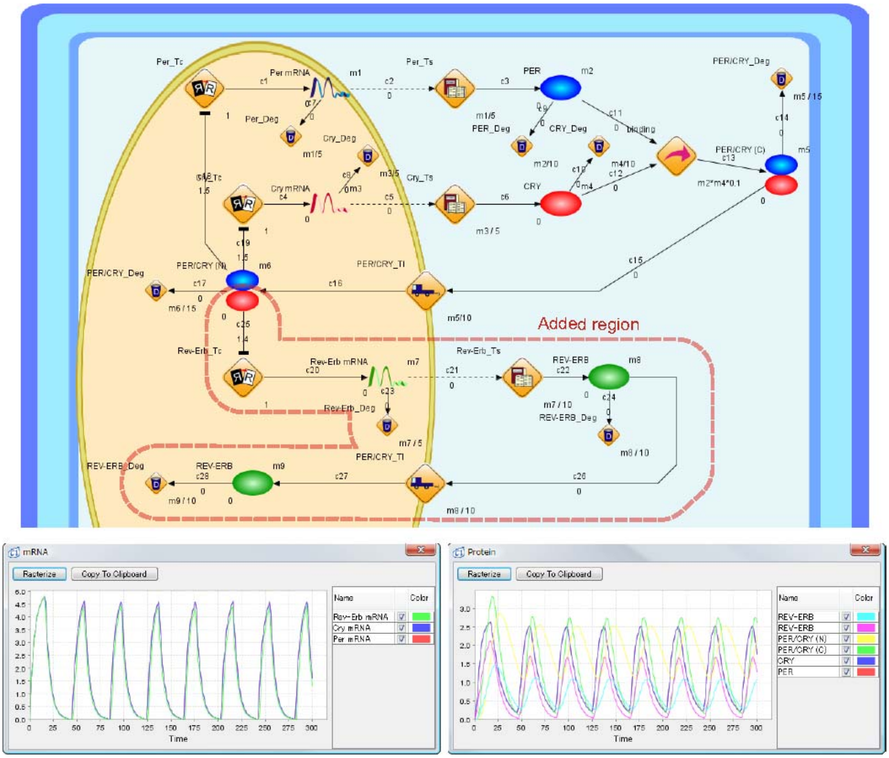
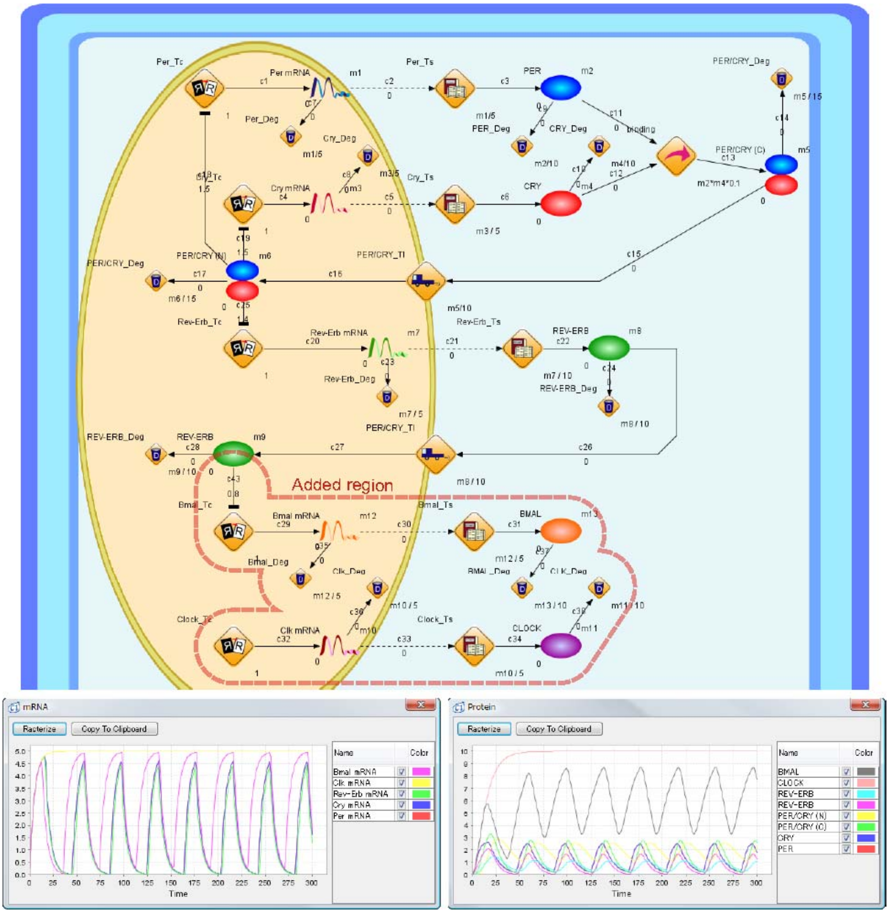


Fig. 5.40

Exercise 5.3.16
Add a process ◆ which creates a complex BMAL/CLOCK ● of BMAL and CLOCK. Set the speed of this process to 1/10th of the product of the concentrations of BMAL and CLOCK. Add also a degradation process D for ● with a degradation rate 1/15th of its concentration.

Exercise 5.3.17
The complex BMAL/CLOCK ● is transported to the nucleus. Add this process to the model, keeping the translocation speed identical to that of the PER/CRY pathway. Add also a degradation process for the complex BMAL/CLOCK in the same operation as the complex PER/CRY.

Then the model should look like Figure 5.42.



Name		Color
Bmal mRNA	☑	🟪
Clk mRNA	☑	🟨
Rev-Erb mRNA	☑	🟩
Cry mRNA	☑	🟦
Per mRNA	☑	🟥

Name		Color
BMAL	☑	⬛
CLOCK	☑	🟥
REV-ERB	☑	🟦
REV-ERB	☑	🟪
PER/CRY (N)	☑	🟨
PER/CRY (C)	☑	🟩
CRY	☑	🟦
PER	☑	🟥

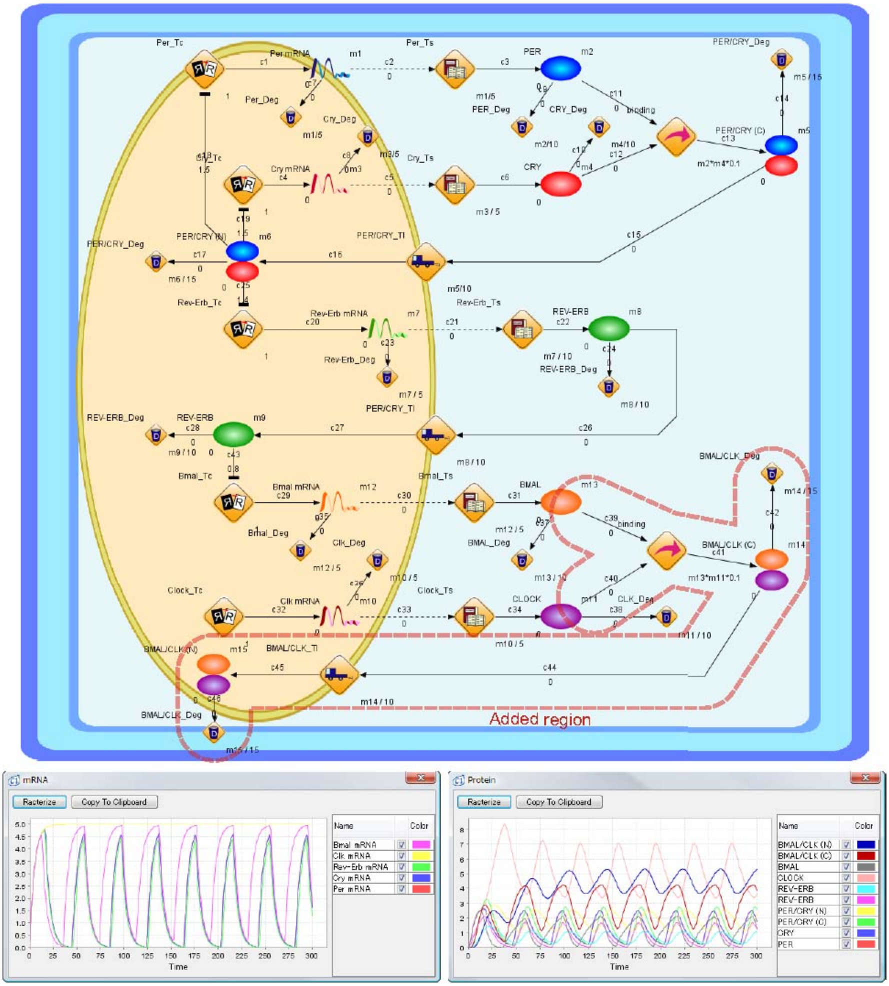
Fig. 5.41

5.3.3.10 Transcriptional Activation by BMAL/CLOCK Complex

The complex BMAL/CLOCK in the nucleus is known to act as a transcription factor. In other words, the complex BMAL/CLOCK binds the upstream sites of the genes *Per*, *Cry*, and *Rev-Erb* and activates the transcription of these genes.

> **Exercise 5.3.18**
> Modify the model so that the complex BMAL/CLOCK activates the transcription of the genes *Per*, *Cry*, and *Rev-Erb*. For each of these three genes, add an association connector ⇣ with a threshold value of 0.5.

Then the model should look like Figure 5.43.



Name		Color
Bmal mRNA	☑	🟪
Clk mRNA	☑	🟨
Rev-Erb mRNA	☑	🟩
Cry mRNA	☑	🟦
Per mRNA	☑	🟥

Name		Color
BMAL/CLK (N)	☑	🟦
BMAL/CLK (C)	☑	🟥
BMAL	☑	⬛
CLOCK	☑	🟥
REV-ERB	☑	🟦
REV-ERB	☑	🟪
PER/CRY (N)	☑	🟨
PER/CRY (C)	☑	🟩
CRY	☑	🟦
PER	☑	🟥

Fig. 5.42

5.3.4 Creating Hypothesis by Simulation

The model in Figure 5.43 is created based on the known biological facts regarding gene regulation but the parameters for the speeds and thresholds are hand-tuned to reflect the facts. The simulation result in Figure 5.43 seems to show circadian rhythms in a correct way, but is inconsistent with known biological observation. This inconsistency concerns the relationship between *Cry* and *Bmal*.

The plot for the *Cry* mRNA has peaks that almost coincide with those of *Bmal*. However, the experiments with murine circadian rhythms showed that the peaks

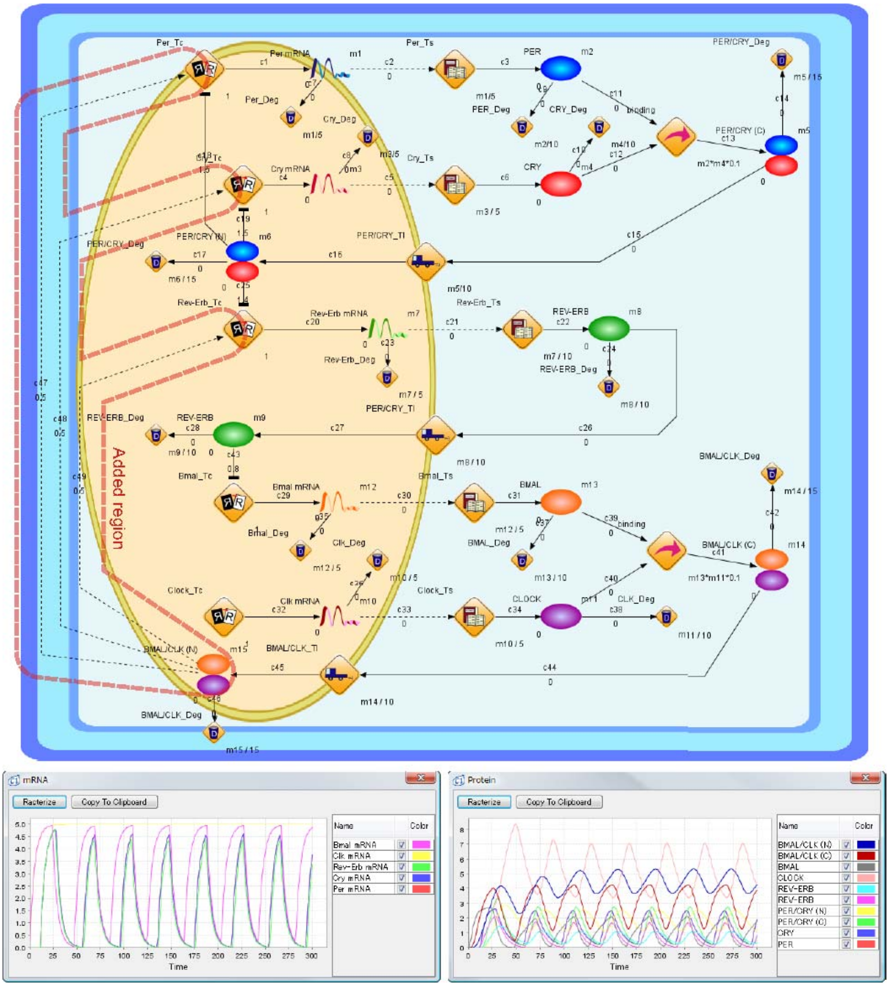


Fig. 5.43

for *Bmal* are located about midway between those for *Per*, which differs from the simulation result.

In order to solve this inconsistency, create a hypothesis that PER/CRY activates *Bmal*. Although this is unconfirmed in mice, an analogous relationship has been found in *Drosophila*. The genes *Cry* and *Bmal* of mouse are similar to the genes *Tim* and *dClk* in *Drosophila*, respectively. It is known that the complex PER/TIM of PER and TIM activates the gene *dClk* in *Drosophila*.

Exercise 5.3.19

By using an association connector, modify the model so that the complex PER/CRY in the nucleus activates the transcription of *Bmal* as mentioned above for *Drosophila*. Set the threshold of the association connector to 1.6.

Then the model and results should look like Figure 5.44.

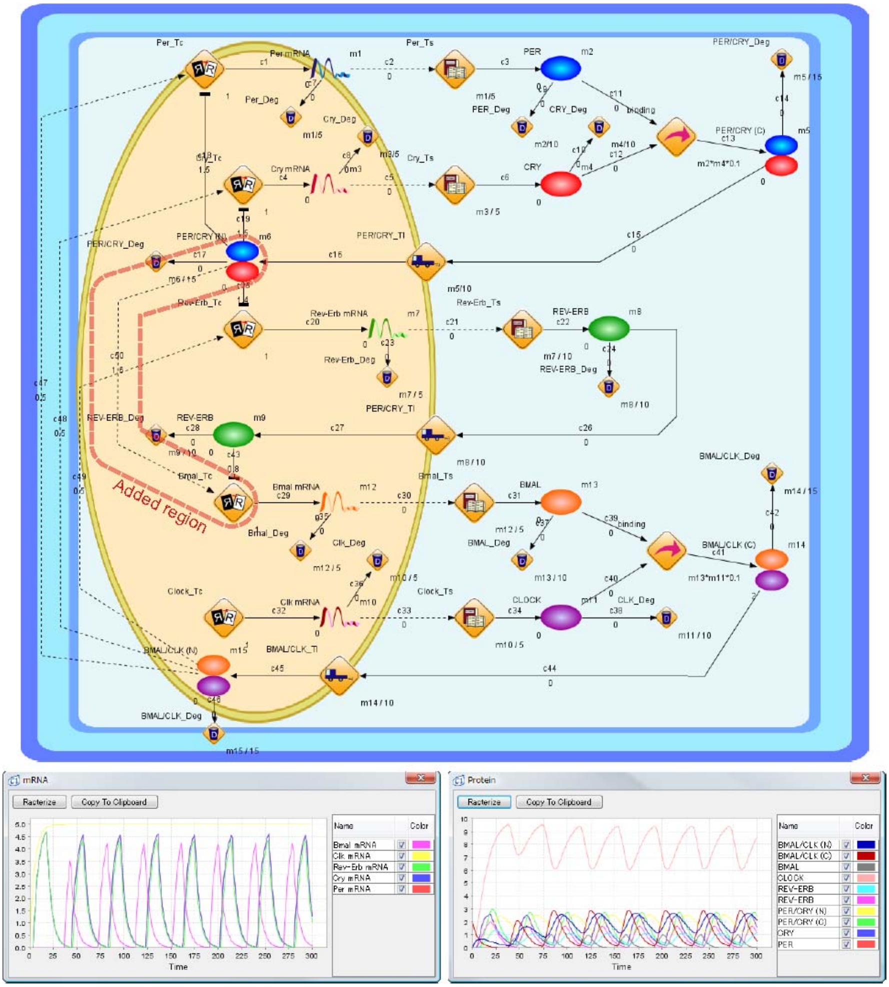


Fig. 5.44

Compared to Figure 5.43, the result in Figure 5.44 shows that the *Bmal* mRNA peaks are in between the *Cry* mRNA peaks. The threshold value of 1.6 was selected within the range of concentration of PER/CRY. Adjusting the parameters and adding an association connector from PER/CRY to the transcription process of *Bmal* successfully brought the model closer to reality. In other words, the hypothesis "PER/CRY activates *Bmal*" might be correct.

Independently, a gene called *Ror* was discovered which activates the *Bmal* transcription but is inhibited by PER/CRY. In other words, PER/CRY is indirectly inhibiting *Bmal* which is opposite to the previous hypothesis. Regardless, simulating with a model using *Ror* and its protein ROR results in the peaks of *Bmal* placed in between those of *Cry*. This simulation result suggests an insight that the role of *Ror* is to shift *Bmal* so that the peaks are in between those of *Cry*.

A model for the *Drosophila* circadian rhythms mentioned above can also easily be constructed on Cell Illustrator. The modeling method is basically the same as described in this chapter.

Figure 5.45 shows the genes and their network which regulate the *Drosophila* circadian rhythms. Though it is similar to the gene regulatory network of mice, there are some differences.

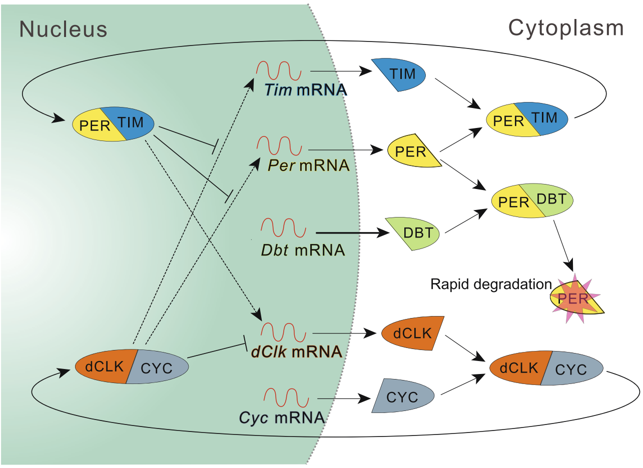


Fig. 5.45

In the case of *Drosophila*, there are five genes involved in the circadian rhythm: *Per*, *Tim*, *dClk*, *Cyc*, and *Dbt*. These correspond to the murine genes *Per*, *Cyc*, *Rev-Erb*, *Clock*, and *Bmal*, respectively.

Drosophila proteins PER and TIM form a heterodimer (PER/TIM) similar to that of PER/CRY. While the protein TIM is expressed in both the nucleus and cytoplasm, the protein PER is expressed only in the cytoplasm. In addition, the complex PER/TIM is known to be more stable than either PER or TIM as is the case for PER/CRY in mice. It is also suggested that PER/TIM formed in the cytoplasm can be transported to the nucleus.

The genes *Cyc* and *dClk* code transcription factors with DNA binding domains. These two genes are known to be homologs of the murine genes *Clock* and *Bmal*.

Similarly to PER and TIM, dCLK and CYC form a heterodimer dCLK/CYC which is translocated to the nucleus. In mice, the proteins CLOCK and BMAL form the complex CLOCK/BMAL that activates *Per*, *Cry*, and *Rev-Erb*. Similarly, dCLK/CYC activates the transcription of *Per* and *Tim*. Furthermore, it is known that dCLK/CYC inhibits the transcription of *dClk* itself, while PER/TIM activates it. PER/TIM is known to inhibit dCLK/CYC from activating *Per* and *Tim*. From the fact that when PER and TIM concentrations are high, dCLK and CYC concentrations are low, the initial values of 10 will be used for PER and TIM.

Actually, there is one gene, *Dbt* (abbreviation for *double-time*), that was left out. From various biological experiments, the following facts are known about the protein DBT:

1) DBT physically binds to PER.
2) DBT phosphorylates PER.
3) Phosphorylation makes PER unstable and causes its degradation.
4) DBT localizes in the cytoplasm.

Adding the gene *Dbt* to the model effectively speeds PER degradation and decreases the amount of PER/TIM in the nucleus. Since PER/TIM in the nucleus inhibits the transcription of *Per* and *Tim*, this effect becomes weaker. As a result, adding *Dbt* to the model increases the period of the circadian rhythm by one cycle. The details are similar to the murine model, which is shown in Figure 5.46. This model for *Drosophila* circadian rhythms is also given in the CD-ROM.

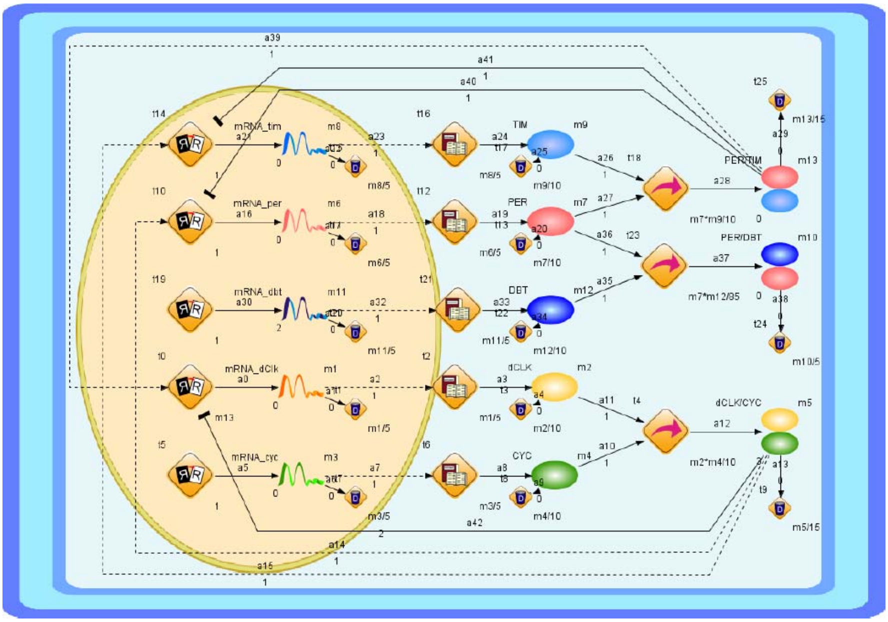


Fig. 5.46

The genes *Dbts* and *Dbtl* (“s” for short, “l” for long) are mutations of *Dbt*. Genotype *Dbts* decreases the time for locomotion cycle compared to the wild type, while *Dbtl* increased the time. In addition, *Dbts* decreases the period of the *Per* mRNA expression compared to the wild type, while *Dbtl* makes this longer. It is considered that this is caused by a change in affinity between proteins DBT and PER.

Exercise 5.3.20
Modify the model in Figure 5.46 by changing the affinity of DBT and PER and confirm that this affects the circadian rhythms.

Many hypotheses will probably come up while watching the behaviors of the models simulated on Cell Illustrator. Since we can see the effects of such changes visually by simulation, these hypotheses may suggest the next biological experiments to design.

Three mutants are known for the gene *Per*: (i) *PerS*, which decreases the period for metamorphosis, (ii) *PerL*, which increases it, and (iii) *Per0* which has no period.

Exercise 5.3.21
Consider the above three mutants. For each mutant, create a hypothesis how the mutant differs from the wild type in the model and modify the model to include the hypothesis. Confirm or reject the hypothesis by simulating the model.

We observed through the models and their simulation how the various functions of the genes and proteins regulate the circadian rhythms in mice and *Drosophila*. In addition, this chapter showed that the model can help in not only confirming known facts but also creating new hypotheses.

5.4 Summary

In this chapter, we created a part of the signal transduction pathway model of EGFR activated by EGF stimulation, and also considered the effects of an inhibitory agent AG1478. A whole EGFR pathway model is shown in [14]. Furthermore, in a similar way, we can create a signal transduction pathway model for apoptosis induced by Fas ligand [8], or pathways involving p53 [12, 13].

We modeled the glycolysis metabolic pathway to show the steps in creating a typical metabolic pathway model. It is possible to connect the glycolytic pathway to the gene regulatory network [10] of the *lac* operon. In addition, it is known that the speed of the glycolysis pathway is mainly regulated by the concentrations of the enzymes whose reactions are irreversible, and their activations are regulated negatively by ATP and positively by AMP. Insulin is known to increase these enzymes while glucagon decreases them. It is a good exercise to add these facts and molecules to the model.

We modeled a gene regulatory network comprised of complex feedback loops involving five genes. Although it might be easy to draw the network, it turned out that the resulting behaviors are far from easy to understand without simulation. For more details, see [16]. Other regulatory networks are the genetic switches model of λ phage [6, 7] and the miRNA regulated network model [15].

Readers are also invited to visit the Cell System Markup Language (CSML) website:

http://www.csml.org/

Far more complex pathways are shown together with explanations. Moreover, any models described with SBML or CellML can be converted for CSML models so that they can be edited and simulated on Cell Illustrator. Including these converted models, over 300 Cell Illustrator models are available for download from this website. In addition, Cell Illustrator equips a tool called *BioPACS* which combines data from KEGG and BRENDA to automatically create models for Cell Illustrator. Using and editing these models, Cell Illustrator allows one to modify them by using experimental data and new biological facts, rather than starting from scratch.

Lastly, many signal transduction pathway models based on the concept of Petri net are provided at the following Petri Net Pathways website:

http://genome.ib.sci.yamaguchi-u.ac.jp/~pnp/

This will also help us in creating models with Cell Illustrator.

Chapter 6
Computational Platform for Systems Biology

Chapters 4 and 5 covered the systematic method to model and simulate pathways. In this chapter, we first introduce a method for visualizing and analyzing large-scale gene networks, and then discuss further functionalities required for the research and development in Systems Biology.

6.1 Gene Network of Yeast

In the previous chapters, towards systems understanding of biology, we introduced a method for creating detailed models mainly based on our biological knowledge. Another method for creating pathways involves generating networks by computation from large-scale experimental data such as gene expression *microarray data*. The current microarray technology has made it possible to measure the gene expression states of tens of thousands of genes.

Computational methods based on computer science and statistics have made it possible to predict the overall gene networks from various measurements of the system states by using computational methods such as *Bayesian network methods* combined with nonparametric regression [17, 18]. Although these generated models are still rough, they can serve as the global map of the relationship between genes with which we can walk around on the gene networks and further biological investigations on the networks will be incubated. In particular, this gene network method is anticipated as a strong strategy for drug target discovery [19–23].

In the next section, we will use the version of Cell Illustrator Standard to analyze a gene network of a small size as an example.

Cell Illustrator comes with a gene network of *yeast* (*Saccharomyces cerevisiae*) (included on the CD-ROM). Microarray measurements of a few hundred genes including transcription factors are behind this gene network of yeast.

Microarray measurements were taken after knocking out each gene. Forcibly halting the expression of a gene such as a transcription factor causes changes in the expressions of the genes under its regulation. This change cascades downwards and causes even more changes. In other words, the gene expression data here reflect the complex gene network at work.

This might be better explained as an analogy. Yeast has approximately 6000 genes. Therefore, consider some city (cell) with 6000 people (genes). This city has various companies, factories, power plants, roads, and communication networks, much like a cell. People do a variety of activities in the city. One day, a person M in this city suddenly dies. M was running a company called G and the management decisions for this company were made by M. Then the people working at the company have no idea what to do, the company stops functioning, and goes bankrupt. As a result, there are those who lose their jobs and income while others find new jobs. On the other hand, company G had a competitor called H. H will record a higher profit, and the company may give out bonuses or hire more people. If it was possible to view the bank accounts of all persons in the city (view the expression levels of all genes with a microarray), then by killing a certain person (knocking out a certain gene), it would be possible to see who profits and who does not (microarray analysis). Then by assuming it was possible to reset the city to its original state (bringing back the dead), kill another person, and measure changes, you would have the human version of yeast microarray analysis. Of course the experiment is not as morbid as the analogy. The experiment sometimes simply adds shocks such as “Making the environment hotter to see if air conditioners sell better”. In other words, this is similar to adding a stimulus such as “killing a person” or “making the city hotter” and seeing the resulting states (bank accounts) of everyone.

The yeast network included on the CD-ROM uses microarray data involving a few hundred gene knockouts and drug responses developed in the research [19, 20]. By using a supercomputer system, we could compute networks of more than 1000 genes (Figure 6.1). Of the 1000 genes, 30 of those are included in this network, where a directed graph shows the relationship as to how the genes are expressed.

6.2 Computational Analysis of Gene Network

We will explain some functions required for the analysis of gene networks.

6.2.1 Displaying Gene Network

Cell illustrator has two modes: *detailed network mode* and *gene network mode*. In Chapters 4 and 5, all editing was done in the detailed network mode. In order to analyze gene networks, Cell Illustrator should be switched to the gene net mode. As seen in Figure 6.2, switch by selecting [View] → [Gene Net Mode] from the menu bar. Afterwards, simply open the yeast gene network through [File] → [Open] (file located at Models\Chapter6\Yeast30repress.xml on the CD-ROM).

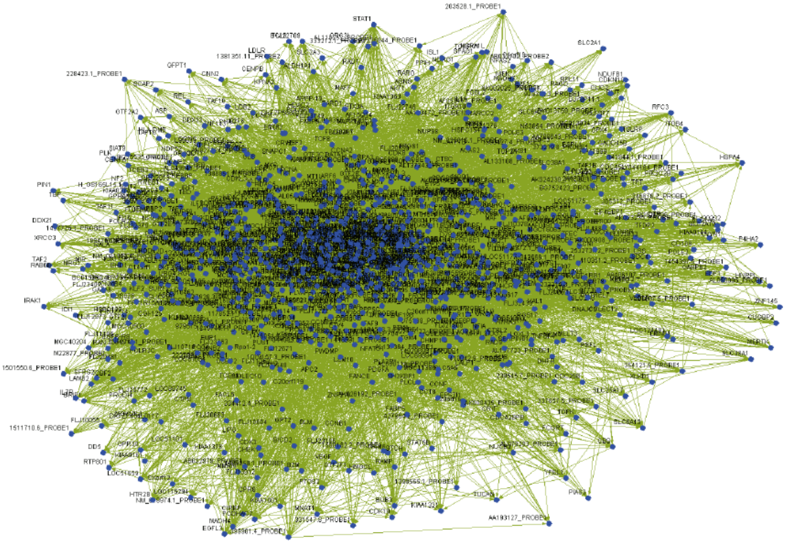

203528.1_PROBE1
STAT1
SLC2A1
LDLR
1381351.11_PROBE2
QFPT1
CNN2
CENPB
228423.1_PROBE1
SCAP2
ISL1
NDUFB1
RFC3
HSPA4
PIN1
DDX21
XRCC3
TAF2
IRAK1
P4HA2
ZNF145
MGC40204
M22877_PROBE1
1501550.6_PROBE1
FLJ10055
1511710.6_PROBE1
DDS
RTP801
LOC51629
HTR2B
NM_006974.1_PROBE1
MADH4
EGFL3
AA193127_PROBE1
PIAS1
TUCPN1
KIAA1211
CDK11

Fig. 6.1

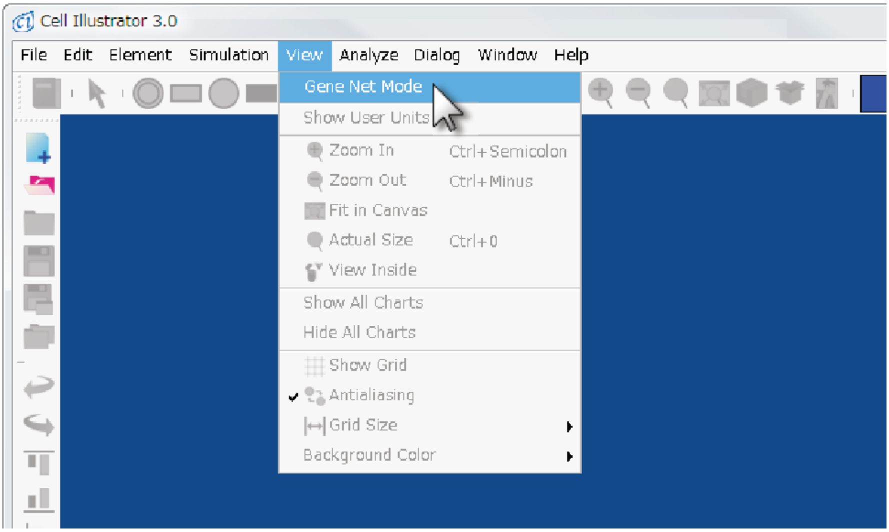

Cell Illustrator 3.0
File
Edit
Element
Simulation
View
Analyze
Dialog
Window
Help
Gene Net Mode
Show User Units
Zoom In Ctrl+Semicolon
Zoom Out Ctrl+Minus
Fit in Canvas
Actual Size Ctrl+0
View Inside
Show All Charts
Hide All Charts
Show Grid
✓ Antialiasing
Grid Size
Background Color

Fig. 6.2

6.2.2 Layout of Gene Networks

Figure 6.3 shows a circular layout of the network. Since this makes the regulatory relationships hard to see, first change this layout.

Click on the right tool bar and open the Optimize Layout Dialog. Using the "Import" button, load the current pathway into the dialog box. Here the layout can be changed by selecting the "CCL Sugiyama" button (Figure 6.4) (Note: the results of the layout will differ each time, and results may not be identical to Figure 6.4.) Then use the "Enlarge" button to increase the spacing between elements. Pressing the "Export" button applies the changes to the canvas (Figure 6.5).

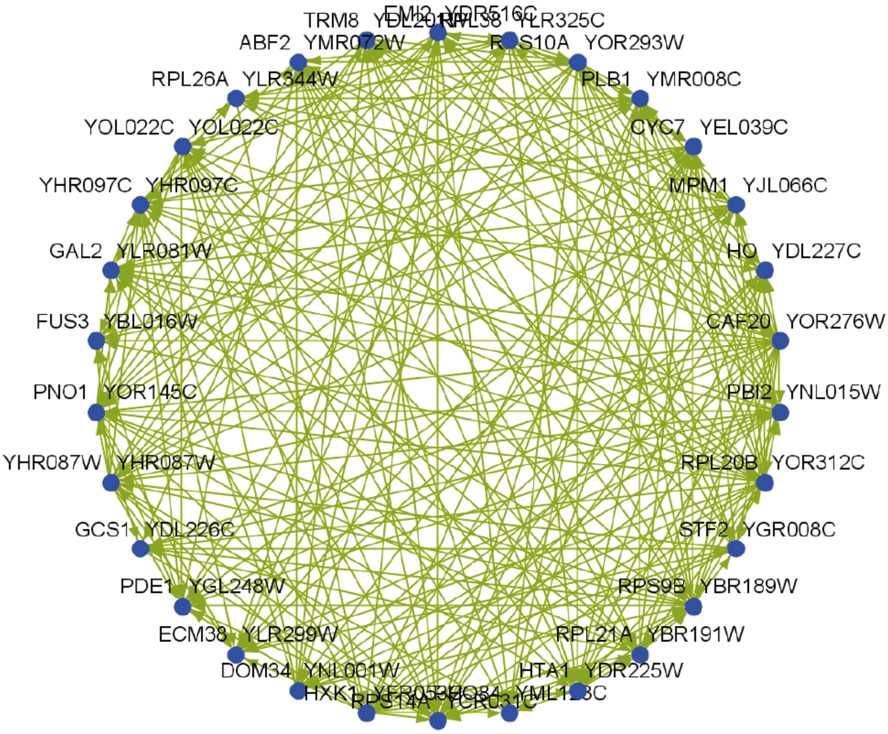


Fig. 6.3

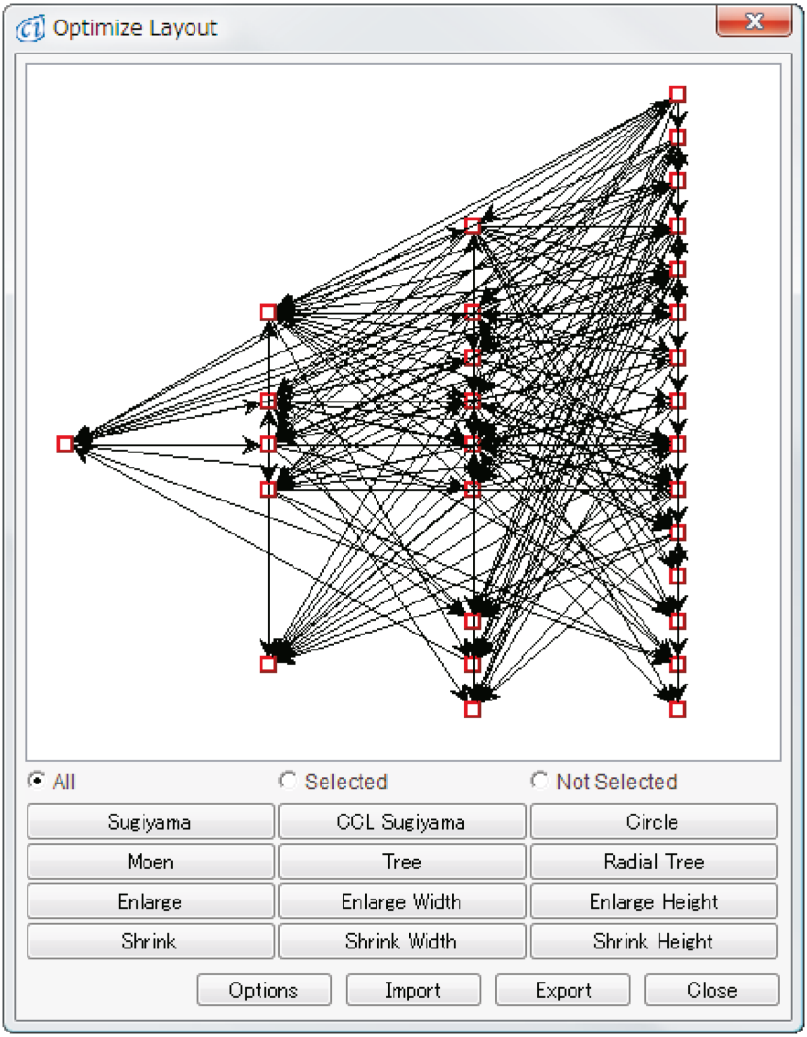


Fig. 6.4

6.2.3 Pathway Search Function

Although Figure 6.5 shows the pathway in a semireadable fashion, it is difficult to grasp the relationships visually because of the number of arrows. Searching and isolating a specific subnetwork can make the view much more intuitive.

Click the 🔭 to display the Pathway Search Dialog. Then enter the parameters in Figure 6.6 and press the "Execute" button. This searches for paths of any direction less than two links (edges) away between ABF2 and STF2. The Pathway Search Results Dialog shows six such subnetworks (Figure 6.7). Highlighting any one of the

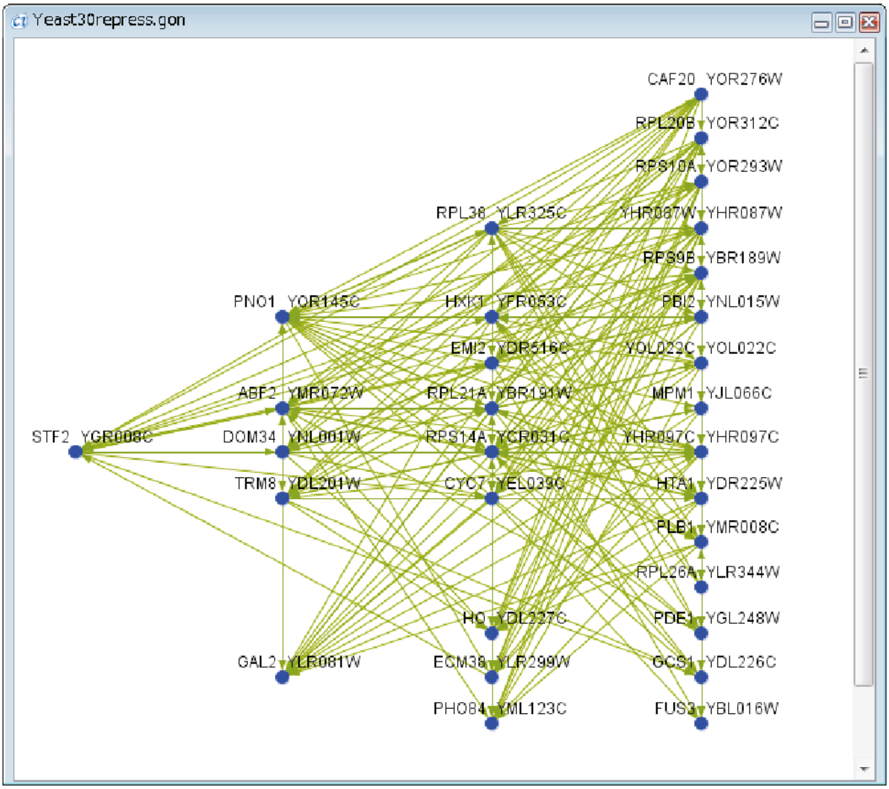
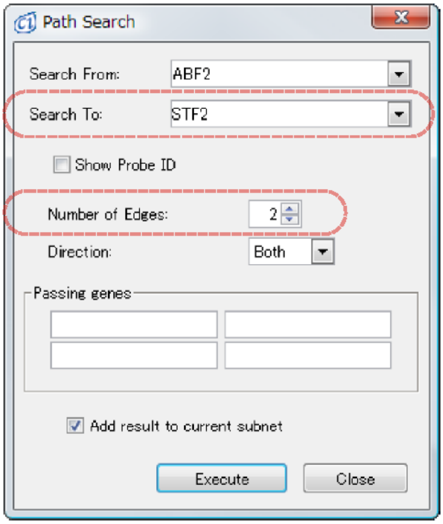


Fig. 6.5

search results shows a subnetwork highlighted in red as seen in Figure 6.8. Selecting multiple rows using the Shift and Ctrl keys shows the combined subnetwork.

6.2.4 Extracting Subnetworks

The subnetwork found in Section 6.2.3 can be extracted onto a new canvas. By selecting [Analyze] → [Extract Subnet], the subnetwork should appear on the canvas as seen in Figure 6.9. By saving this network, the subnetwork of interest can be called up at any time. Save the extracted subnetwork by selecting [File] → [Save As] for Section 6.2.4 as Extract1.xml. Call this extracted subnetwork the subnetwork P1.



Search From: ABF2

Search To: STF2

☐ Show Probe ID

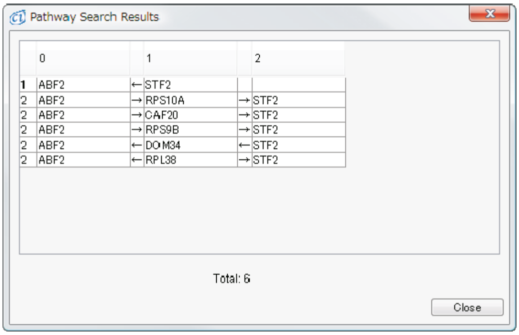
Number of Edges: 2

Direction: Both

Passing genes

☑ Add result to current subnet

Fig. 6.6



	0		1		2
1	ABF2	←	STF2		
2	ABF2	→	RPS10A	→	STF2
2	ABF2	→	CAF20	→	STF2
2	ABF2	→	RPS9B	→	STF2
2	ABF2	←	DOM34	←	STF2
2	ABF2	←	RPL38	→	STF2

Total: 6

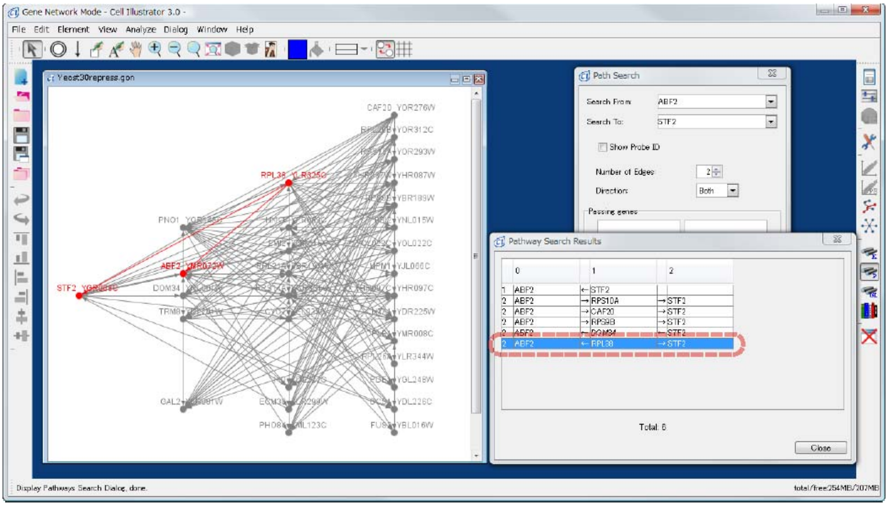
Fig. 6.7

6.2.5 Comparing Two Subnetworks

By comparing subnetworks extracted with various conditions, it is possible to grasp the genes which are co-regulated and which are not. For example, comparing the networks before and after EGF stimulation would isolate the regulation mechanisms unique to EGF.

Two subnetworks, P1 created in Section 6.2.4 which contains bidirectional links between ABF2 and STF2 and P2 which contain only links from STF2 to ABF2, are compared using four methods in Table 6.1.

First, the network created in Section 6.2.4 is still selected in red, so release using [Analyze] → [Clear Subnet]. Next, extract the network P2 using the parameters shown in Figure 6.6 while setting the direction to reverse (Figure 6.10). Changing


Gene Network Mode - Cell Illustrator 3.0 -
File Edit Element View Analyze Dialog Window Help
Yeast30repress.gon
Path Search
Pathway Search Results
Close

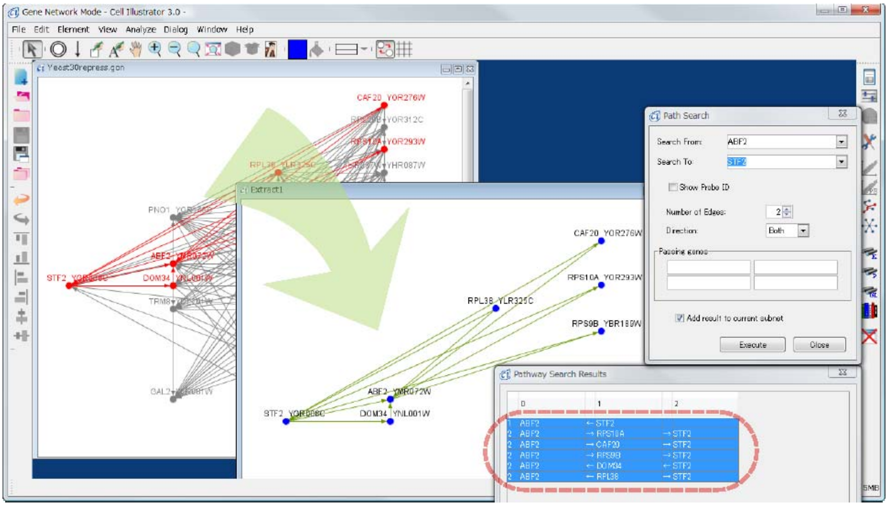
Search From: ABF2
Search To: STF2
□ Show Probe ID
Number of Edges: 2
Direction: Both
Passing genes

	0	1	2
1	ABF2	←STF2	
2	ABF2	→RPS10A	→STF2
2	ABF2	→CAF20	→STF2
2	ABF2	→RPS9B	→STF2
2	ABF2	←DOM34	←STF2
2	ABF2	←RPL38	→STF2

Total: 6

Display Pathways Search Dialog, done.
total/free:254MB/207MB

Fig. 6.8



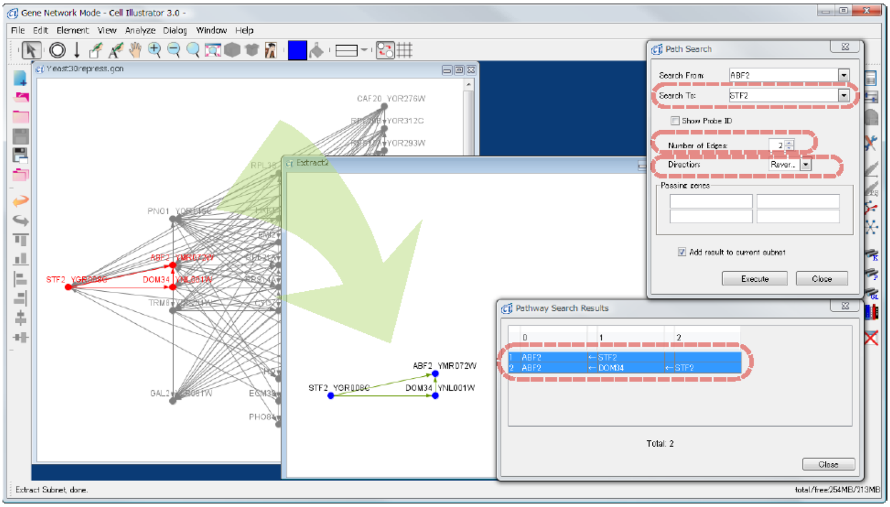
Search From: ABF2
Search To: STF2
□ Show Probe ID
Number of Edges: 2
Direction: Both
Passing genes
☑ Add result to current subnet

	0	1	2
1	ABF2	← STF2	
2	ABF2	→ RPS10A	→ STF2
2	ABF2	→ CAF20	→ STF2
2	ABF2	→ RPS9B	→ STF2
2	ABF2	← DOM34	← STF2
2	ABF2	← RPL38	→ STF2

Fig. 6.9

this option extracts only the networks which go from STF2 to ABF2. Save this new network to a different file (Extract2.xml).

Since we have now two networks to compare, close all files using . Then by using [File] → [Open], open both P1 and P2. Finally, select [Analyze] → [Compare] to display six images as shown in Figure 6.11. These canvases correspond to the four methods of comparison and the two original pathways P1 (left bottom) and P2 (right bottom).



Search From: ABF2

Search To: STF2

□ Show Probe ID

Number of Edges: 2

Direction: Rever..

Passing genes

☑ Add result to current subnet

0	1	2
1 ABF2	←STF2	
2 ABF2	←DOM34	←STF2

Total: 2

Fig. 6.10

Table 6.1

No.	Comparison Method	Explanation
1	P1\P2	The result of subtracting the elements of P2 from P1. In the example, this would remove all paths which are directed from STF2 to ABF2 in P2 from P1.
2	P2\P1	The result of subtracting the elements of P1 from P2. In the example, this would remove all paths in P1 from paths heading from ABF2 to STF2 in P2 (resulting in an empty set).
3	P1∩P2	Extracts the subnetwork which is common to both P1 and P2. In the example, since all paths from ABF2 to STF2 in P2 are also included in P1, the result would be P2.
4	P1∪P2	The union of subnetworks P1 and P2. In the example, since all paths in P2 are included in P1, the result would be P1.

These comparisons allow us quick identification of common and unique pathways.

In this section we explained the basic analysis of gene networks using Cell Illustrator. The features covered in this section can also be used in the detailed network mode used in Chapters 4 and 5. The measurement and computational technology for analyzing gene networks is being improved considerably, and the analysis tools such as Cell Illustrator will thus have more extensive functionalities.

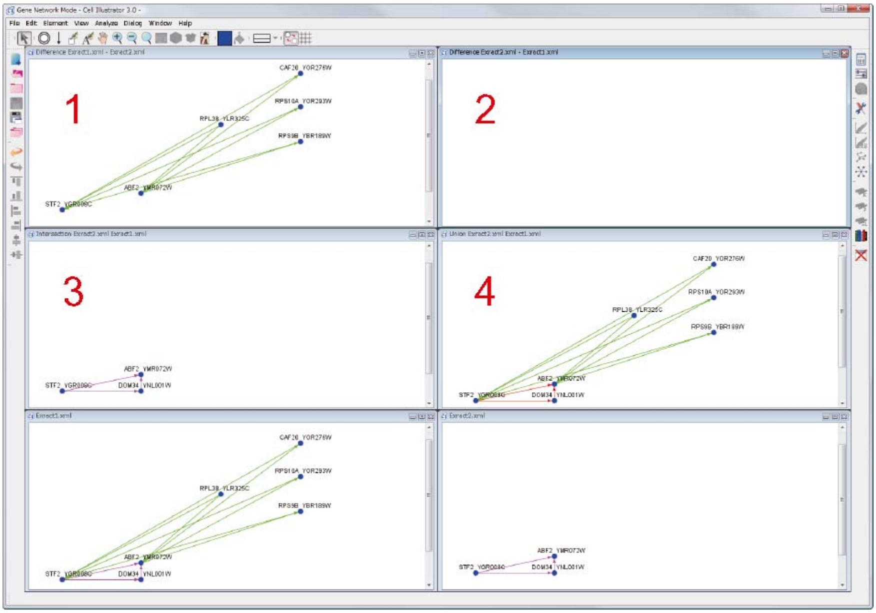


Fig. 6.11

6.3 Further Functionalities for Systems Biology

In this section, we consider computational capabilities which will accelerate the research and development in Systems Biology. For this purpose, we briefly introduce the online version *Cell Illustrator Online 4.0* (CIO4.0). This product is the next version of Cell Illustrator 3.0 and more functionalities are newly introduced or enhanced to meet various requirements from Systems Biology. One of the features of CIO4.0 is that the application launches via the Internet by using Java Web Start technology and the user can always employ the latest functionalities in CIO4.0. The application can be evaluated at the following websites:

http://cionline.bioillustrator.com/
http://cionline.hgc.jp/

6.3.1 Languages for Pathways: CSML 3.0 and CSO

As we have seen in Chapter 2, various pathway databases are built up with their own XML formats. Therefore, it is important to have a strong XML which can cover most of them for data import without loss of information while keeping the biological meaning. CSML 3.0 is a highly optimized XML format for biological pathway modeling and simulation that almost achieved this objective. This XML format has a full compatibility with the pathway modeling and simulation ontology format Cell System Ontology (CSO) (Section 2.3.6). CI4.0 fully supports CSML3.0 though CI3.0 supports CSML1.9. With CSML3.0, other modeling and simulation XML formats such as SBML (see Section 2.3.4) and CellML (see Section 2.3.3) can be imported to CIO4.0. The ontology format BioPAX (see Section 2.3.5) can also be imported to CIO4.0. Figure 6.12 shows these functionalities.

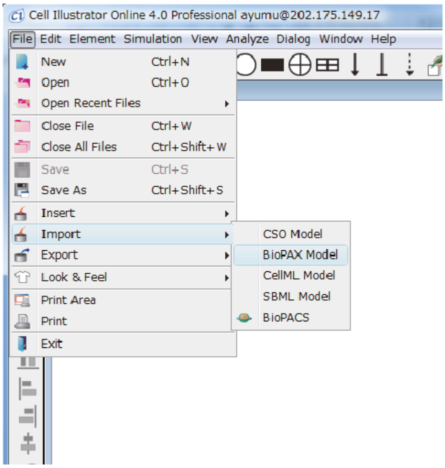


Fig. 6.12

6.3.2 *SaaS Technology*

Various types of analysis requests are coming from the R&D in Systems Biology. Some of them require a huge supercomputer system for such as optimal parameter search mentioned later in Section 6.3.3, very expensive/large databases to own, or very specific analysis focused on a specific research topic. A single software cannot cover all capabilities inside, and the software customization is very expensive or impossible to cope with. Thus, inevitably, we need Software as a Service (SaaS) technology for the computational platform for Systems Biology.

SaaS is a software application delivery model which is usually associated with software business and is considered as a low-cost way to obtain the same merits without the associated complexity and high initial cost as licensing.

SaaS technology is introduced in CIO4.0 and modules are serviced on the servers introduced at the beginning of Section 6.3. Clients can select the desired modules from them on their demands. The following eight modules including the beta version are serviced on the server side as of March 2008:

1) TRANSPATH Search Module
2) Project Management Module
3) TRANSPATH Pathway Library Module
4) High-performance Simulation Module
5) Pathway Model to Multiple Program Languages Export Module (Java, Fortran, C++, C, Perl, and Python are supported)
6) Pathway Parameter Search Module
7) CSML to SVG Module (beta)
8) CSML to HTML Module (beta)

6.3.3 Pathway Parameter Search

The problem of parameter search for dynamic pathway models is one of the most important topics in Systems Biology. Some challenges have been made for automatic parameter estimation for HFPNe models by using a technology called *data assimilation* which "blends" simulation models and observational data "rationally" [14, 27]. This data assimilation method is more suited for a high-performance computing system with Peta FLOPS computing ability. These efforts and developments are anticipated to create groundbreaking modeling platforms for Systems Biology.

Although the data assimilation technology is still in the stage of research and challenge, a "Pathway Parameter Search Module" is provided for CIO4.0 for a small size parameter search. For a pathway model, this module executes the user-specified multiple initial conditions at once and displays the result with 2D or 3D plots. By using this method, the user can create a highly tuned model that fits the observed result. Figure 6.13 shows a snapshot of this pathway search module. We can see how the systems behavior will change according to the changes in parameters.

6.3.4 Much Faster Simulation

Thorough parameter search requires much faster simulation. Since the original simulation engine in Cell Illustrator is a script-based one, CIO4.0 is enhanced with a "High-performance Simulation Module" whose performance is 10 to 100 times faster than the original engine. A comparison is shown in Table 6.2, where the pathway models used in comparison can be downloaded from http://www.csml.org/.

6.3.5 Exporting Pathway Models to Programming Languages

If the models described with CSML could not be used by other software applications, the efforts for making such models would be less appreciated. In order to

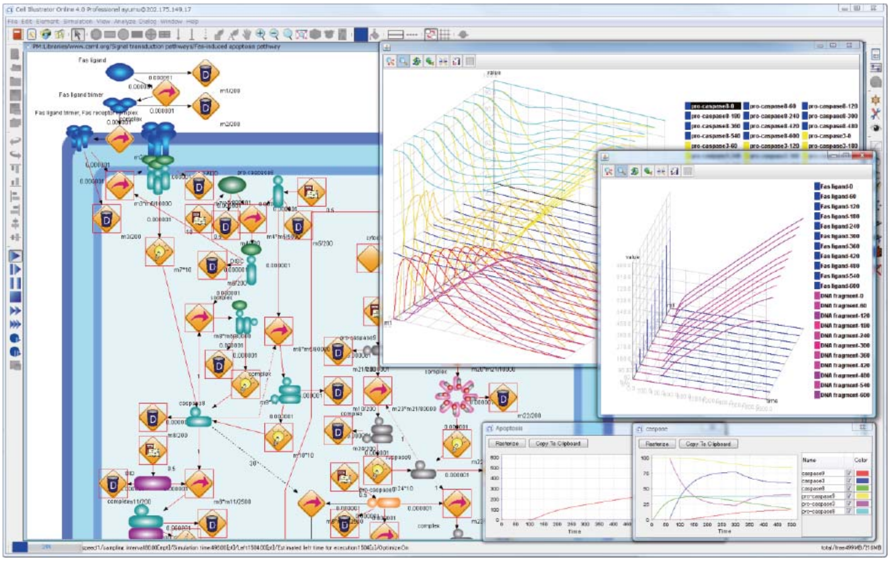


Fig. 6.13

Table 6.2

Pathway Model	Original Engine	Module Engine
Circadian rhythms in *Mus musculus* (20000 [pt])	37 sec	4 sec
Fas-induced apoptosis pathway (20000 [pt])	66 sec	5 sec
p53 CDK-dependent phosphorylation pathway (1000 [pt])	17 sec	5 sec
p53 ARF-dependent stabilization pathway (1000 [pt])	3 sec	Less than 1 sec
Glycolytic pathway and *lac* operon of *E. coli* (10000 [pt])	76 sec	5 sec

@Intel Core 2 (3.0 G Hz)

cope with this situation, CIO4.0 is provided with a “Pathway Model to Multiple Program Languages Export Module”. This module exports the model on CIO4.0 to native programming languages. As of March 2008, Java, Fortran, C++, C, Perl, and Python are supported (Figure 6.14). By using this module, users can seamlessly integrate their own applications to the model developed on CIO.

6.3.6 Pathway Layout Algorithms

Pathway layout is a key to better understanding of pathways. CI3.0 has the 14 basic pathway layout algorithms. In CI4.0, seven more layout algorithms are provided. Especially, highly customized layout algorithms for biological pathways, and three

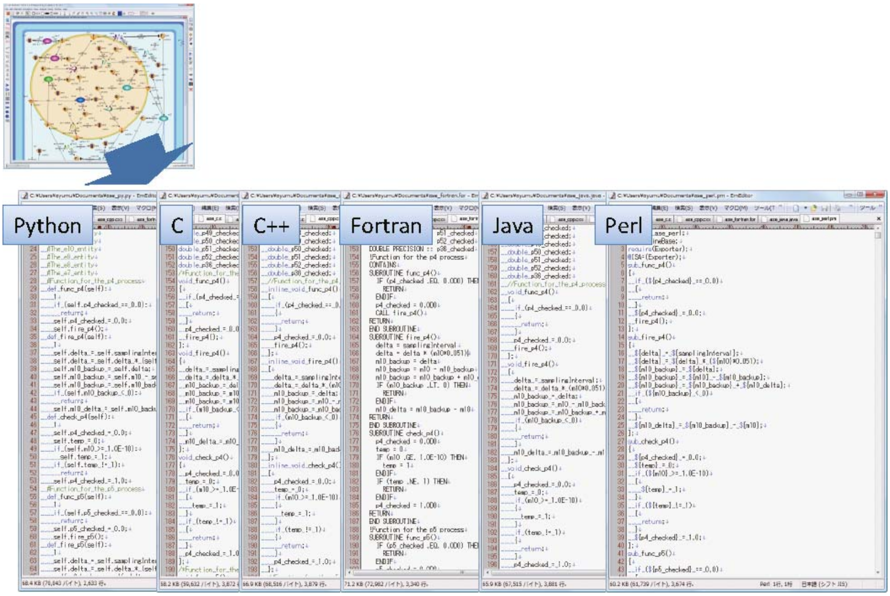


Fig. 6.14

grid layout algorithms (BLK, CB, and SCCB) are included. Figure 6.15 shows an example of layout. Biological knowledge represented with Cell System Ontology (CSO) is utilized in these algorithms so that biological entities and processes are arranged at appropriate subcellular locations.

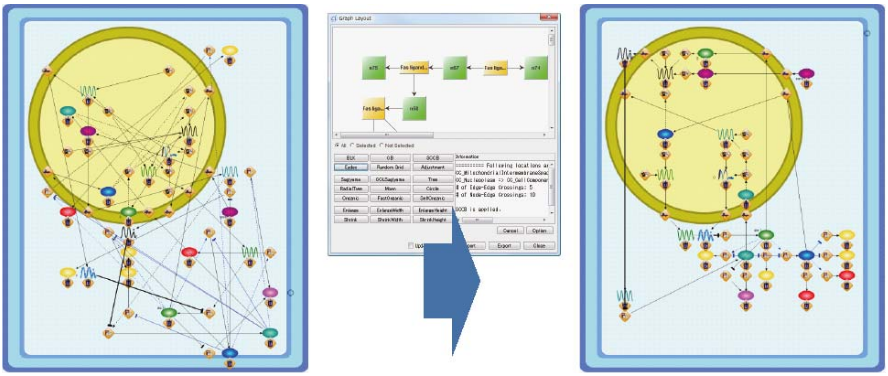
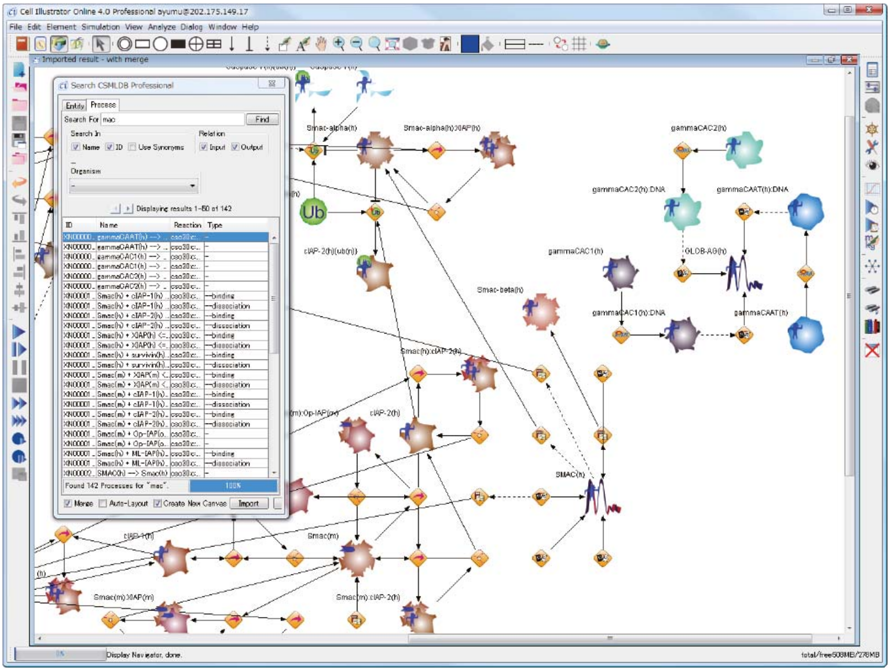


Fig. 6.15

6.3.7 Pathway Database Management System

A database management system (DBMS) with GUI is also an essential functionality. CIO4.0 has a DBMS for CSML which allows various searches via GUI such as “TRANSPATH Search Module”. As of March 2008, TRANSPATH is fully supported [26]. This means that all pathway information in TRANSPATH is converted to the data in CSML format so that pathway search results can be displayed on CIO4.0 as shown in Figure 6.16. Moreover, with “TRANSPATH Pathway Library Module”, more than 1000 well-established biological pathways in TRANSPATH (signal transduction pathways and gene regulatory networks) can also be used on CIO4.0 in a way that all pathways can be loaded, edited, saved, and simulated on the user’s own terms (Figure 6.17).

“Project Management Module” is also a useful environment where each project on the server can be shared by other permitted users (read, write, or both permissions) and public pathway models such as those in http://www.csml.org/ can be accessed via the GUI of the module (Figure 6.18).



Search For: mac

Search In: ☑ Name ☑ ID □ Use Synonyms

Relation: ☑ Input ☑ Output

Organism: –

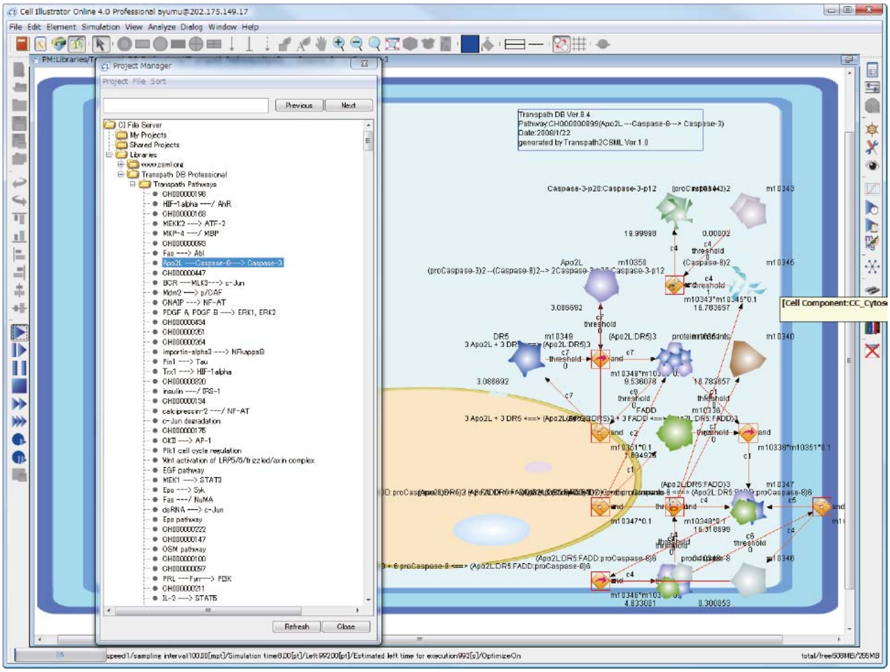
Displaying results 1–50 of 142

ID	Name	Reaction	Type
XN00000..	gammaCAAT(h) --> ..	cso30:c:..	–
XN00000..	gammaCAAT(h) --> ..	cso30:c:..	–
XN00000..	gammaCAC1(h) --> ..	cso30:c:..	–
XN00000..	gammaCAC1(h) --> ..	cso30:c:..	–
XN00000..	gammaCAC2(h) --> ..	cso30:c:..	–
XN00000..	gammaCAC2(h) --> ..	cso30:c:..	–
XN00001 ..	Smac(h) + cIAP-1(h) ..	cso30:c:..	--binding
XN00001 ..	Smac(h) + cIAP-1(h) ..	cso30:c:..	--dissociation
XN00001 ..	Smac(h) + cIAP-2(h) ..	cso30:c:..	--binding
XN00001 ..	Smac(h) + cIAP-2(h) ..	cso30:c:..	--dissociation
XN00001 ..	Smac(h) + XIAP(h) <=..	cso30:c:..	--binding
XN00001 ..	Smac(h) + XIAP(h) <=..	cso30:c:..	--dissociation
XN00001 ..	Smac(h) + survivin(h)..	cso30:c:..	--binding
XN00001 ..	Smac(h) + survivin(h)..	cso30:c:..	--dissociation
XN00001 ..	Smac(m) + XIAP(m) <..	cso30:c:..	--binding
XN00001 ..	Smac(m) + XIAP(m) <..	cso30:c:..	--dissociation
XN00001 ..	Smac(m) + cIAP-1(h)..	cso30:c:..	--binding
XN00001 ..	Smac(m) + cIAP-1(h)..	cso30:c:..	--dissociation
XN00001 ..	Smac(m) + cIAP-2(h)..	cso30:c:..	--binding
XN00001 ..	Smac(m) + cIAP-2(h)..	cso30:c:..	--dissociation
XN00001 ..	Smac(m) + Op-IAP(o..	cso30:c:..	–
XN00001 ..	Smac(m) + Op-IAP(o..	cso30:c:..	–
XN00001 ..	Smac(h) + ML-IAP(h)..	cso30:c:..	--binding
XN00001 ..	Smac(h) + ML-IAP(h)..	cso30:c:..	--dissociation
XN00002..	SMAC(h) --> Smac(h)	cso30:c:..	–

Found 142 Processes for "mac". 100%

☑ Merge □ Auto-Layout ☑ Create New Canvas

Fig. 6.16



C] File Server
My Projects
Shared Projects
Libraries
www.csml.org
Transpath DB Professional
Transpath Pathways
CH000000198
HIF-1alpha ---/ AhR
CH000000168
MEKK2 ---> ATF-2
MKP-4 ---/ MBP
CH000000093
Fas ---> Abl
Apo2L ---Caspase-8---> Caspase-3
CH000000447
BCR ---MLK3---> c-Jun
Mdm2 ---> p/CAF
CNAIP ---> NF-AT
PDGF A, PDGF B ---> ERK1, ERK2
CH000000434
CH000000261
CH000000264
importin-alpha3 ---> NFkappaB
Pin1 ---> Tau
Trx1 ---> HIF-1alpha
CH000000820
insulin ---/ IRS-1
CH000000134
calcipressin-2 ---/ NF-AT
c-Jun degradation
CH000000176
CKII ---> AP-1
Plk1 cell cycle regulation
Wnt activation of LRP5/6/frizzled/axin complex
EGF pathway
MEK1 ---> STAT3
Epo ---> Syk
Fas ---/ NuMA
dsRNA ---> c-Jun
Epo pathway
CH000000222
CH000000147
OSM pathway
CH000000100
CH000000097
PRL ---Fyn---> PI3K
CH000000211
IL-2 ---> STAT5

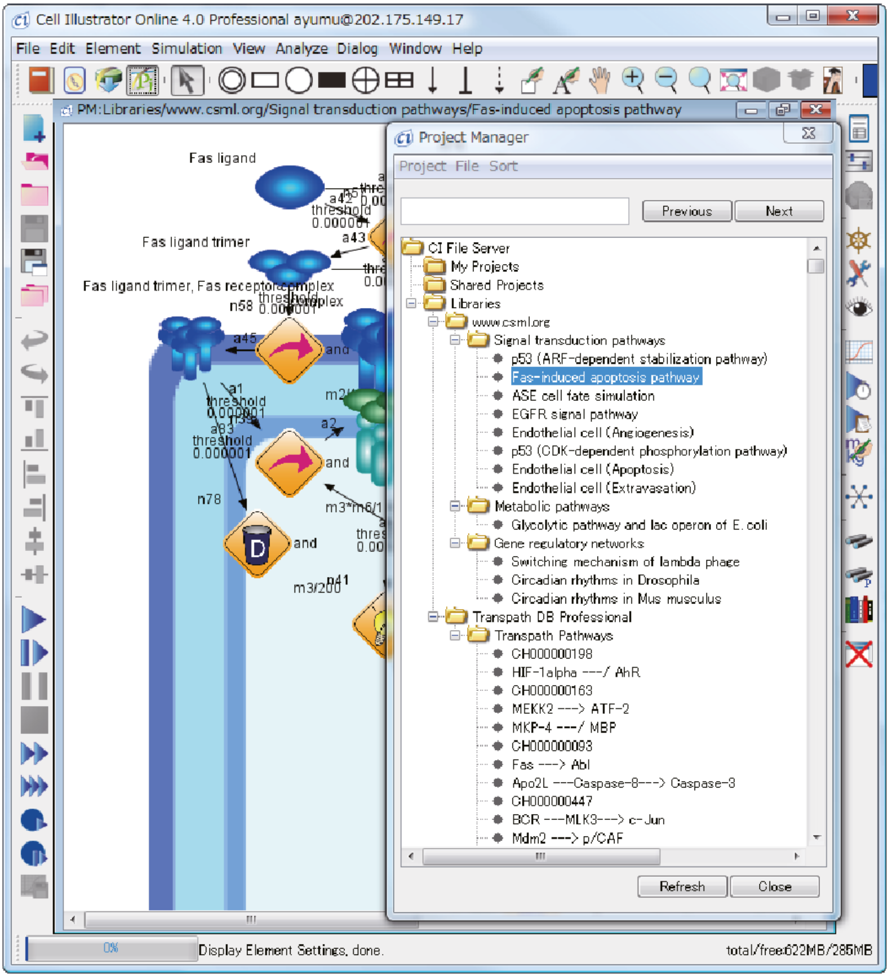
Transpath DB Ver.8.4
Pathway:CH000000899(Apo2L---Caspase-8---> Caspase-3)
Date:2008/1/22
generated by Transpath2CSML Ver.1.0

0%
speed1/sampling interval100.00[mpt]/Simulation time8.00[pt]/Left99200[pt]/Estimated left time for execution992[s]/OptimizeOn
total/free508MB/255MB

Fig. 6.17

6.3.8 More Visually: Automatic Generation of Icons

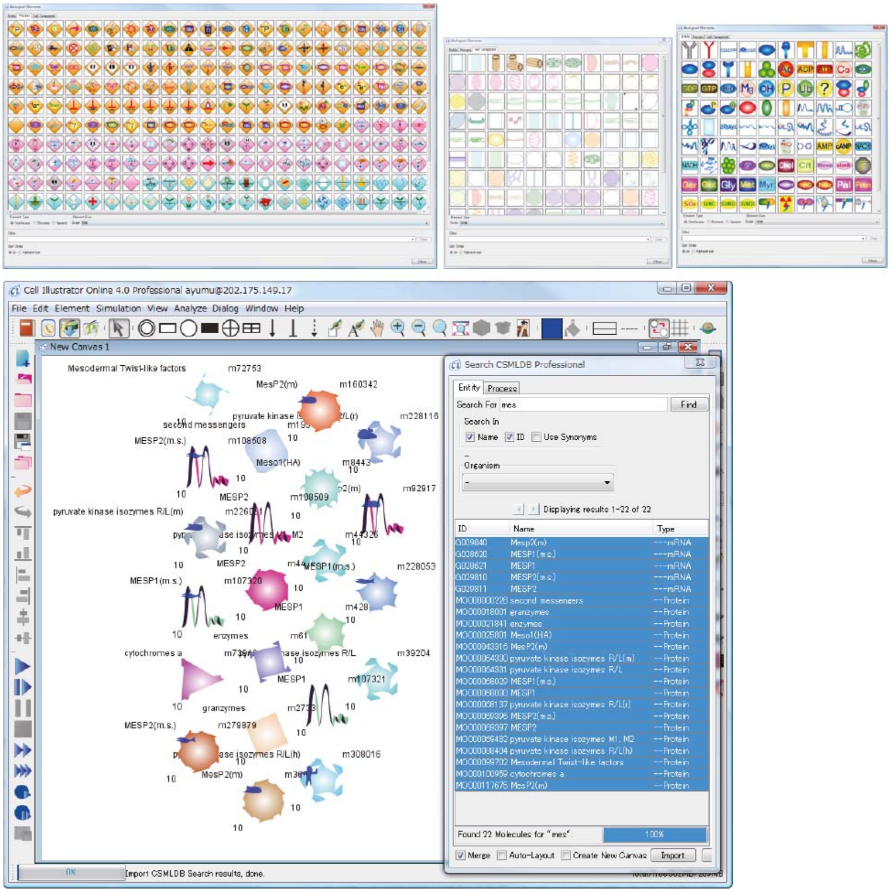
Biologically intuitive and human-friendly icons are very appreciated in practice. CI3.0 has about 350 handmade icons. However, when we use pathway databases such as TRANSPATH, more than 100,000 biologically intuitive but unique icons are necessary for graphical drawing of pathways. The number is beyond the capacity of handmade or pseudo-handmade level. Thus the functionality is needed to automatically generate icons which reflect biological meanings attached to the biological objects. This functionality was developed for CIO4.0 so that CIO4.0 can fully use TRANSPATH in addition to handmade icons (Processes: 92; Entities: 275; Cell Components: 114) for more detailed pathway modeling. Figure 6.19 shows these handmade icons (up) and automatically generated icons for TRANSPATH (down).


Cell Illustrator Online 4.0 Professional ayumu@202.175.149.17
File Edit Element Simulation View Analyze Dialog Window Help
PM:Libraries/www.csml.org/Signal transduction pathways/Fas-induced apoptosis pathway
Fas ligand
Fas ligand trimer
Fas ligand trimer, Fas receptor complex
a42
threshold
0.000001
a43
n58
a45
and
a1
threshold
0.000001
a83
threshold
0.000001
n78
a2
and
D
and
m3/200
m3*m6/1
Project Manager
Project File Sort
Previous
Next
Refresh
Close
0%
total/free:622MB/285MB

CI File Server
- My Projects
- Shared Projects
- Libraries
  - www.csml.org
    - Signal transduction pathways
      - p53 (ARF-dependent stabilization pathway)
      - Fas-induced apoptosis pathway
      - ASE cell fate simulation
      - EGFR signal pathway
      - Endothelial cell (Angiogenesis)
      - p53 (CDK-dependent phosphorylation pathway)
      - Endothelial cell (Apoptosis)
      - Endothelial cell (Extravasation)
    - Metabolic pathways
      - Glycolytic pathway and lac operon of E. coli
    - Gene regulatory networks
      - Switching mechanism of lambda phage
      - Circadian rhythms in Drosophila
      - Circadian rhythms in Mus musculus
  - Transpath DB Professional
    - Transpath Pathways
      - CH000000198
      - HIF-1alpha ---/ AhR
      - CH000000163
      - MEKK2 ---> ATF-2
      - MKP-4 ---/ MBP
      - CH000000093
      - Fas ---> Abl
      - Apo2L ---Caspase-8---> Caspase-3
      - CH000000447
      - BCR ---MLK3---> c-Jun
      - Mdm2 ---> p/CAF

Display Element Settings, done.

Fig. 6.18


Cell Illustrator Online 4.0 Professional ayumu@202.175.149.17
File Edit Element Simulation View Analyze Dialog Window Help
New Canvas 1
Mesodermal Twist-like factors
MesP2(m)
second messengers
MESP2(m.s.)
Meso1(HA)
MESP2
pyruvate kinase isozymes R/L(m)
MESP1(m.s.)
MESP1
enzymes
cytochromes a
granzymes
MesP2(m)
Search CSMLDB Professional
Entity
Process
Find
Import
OK

Search For mes

Search In

☑ Name ☑ ID ☐ Use Synonyms

Organism -

Displaying results 1-22 of 22

ID	Name	Type
G009840	Mesp2(m)	---mRNA
G028620	MESP1(m.s.)	---mRNA
G028621	MESP1	---mRNA
G029810	MESP2(m.s.)	---mRNA
G029811	MESP2	---mRNA
MO000000228	second messengers	--Protein
MO000018001	granzymes	--Protein
MO000021841	enzymes	--Protein
MO000025801	Meso1(HA)	--Protein
MO000043316	MesP2(m)	--Protein
MO000064330	pyruvate kinase isozymes R/L(m)	--Protein
MO000064331	pyruvate kinase isozymes R/L	--Protein
MO000068029	MESP1(m.s.)	--Protein
MO000068030	MESP1	--Protein
MO000068137	pyruvate kinase isozymes R/L(r)	--Protein
MO000069396	MESP2(m.s.)	--Protein
MO000069397	MESP2	--Protein
MO000069482	pyruvate kinase isozymes M1, M2	--Protein
MO000098404	pyruvate kinase isozymes R/L(h)	--Protein
MO000099702	Mesodermal Twist-like factors	--Protein
MO000100959	cytochromes a	--Protein
MO000117675	MesP2(m)	--Protein

Found 22 Molecules for "mes". 100%

☑ Merge ☐ Auto-Layout ☐ Create New Canvas

Import CSMLDB Search results, done.

Fig. 6.19

Bibliographic Notes

The development of Cell Illustrator, introduced in Chapter 4, began in 1999 with the intent of "creating Systems Biology software that people in biology labs can use without any computer science skills". Cell Illustrator originally began under the code name *Genomic Object Net* (http://www.genomicobject.net/). We proposed the Hybrid Functional Petri Net with extension (HFPNe) in 2003 [1]. Simultaneously, we also proposed a method for building pathway models based on HFPNe [2].

[1] Nagasaki M, Doi A, Matsuno H, Miyano S. Genomic Object Net: I. A platform for modeling and simulating biopathways. *Applied Bioinformatics*. **2**:181–184, 2003.

[2] Doi A, Nagasaki M, Fujita S, Matsuno H, Miyano S. Genomic Object Net: II. Modeling biopathways by hybrid functional Petri net with extension. *Applied Bioinformatics*. **2**:185–188, 2003.

The details of HFPNe, which is currently the architecture for Cell Illustrator, are defined in [3]. In addition, we also explained in [4] the details of the tool BioPACS in Section 5.4 as well as tools such as Cell Animator (Genomic Object Net Visualizer). Later on, Cell System Markup Language 3.0 and Cell System Ontology 3.0, mentioned in Chapter 2, were developed. Various improvements and enhancements to the GUI and backend were made while creating models like those seen in Chapter 5 and led to the current version of Cell Illustrator.

[3] Nagasaki M, Doi A, Matsuno H, Miyano S. A versatile Petri net based architecture for modeling and simulation of complex biological processes. *Genome Informatics*. **15**(1):180–197, 2005. http://www.jsbi.org/modules/journal/index.php/IBSB04/IBSB04F020.pdf

[4] Nagasaki M, Doi A, Matsuno H, Miyano S. Computational modeling of biological processes with Petri net-based architecture. *Bioinformatics Technologies* (Yi-Ping Phoebe Chen, Ed.). Springer. pp.179–242, 2005.

In Chapter 4, we mentioned that Petri net is the basis of Cell Illustrator. Usually, Petri net theory uses terms such as *place*, *transition*, and *arc* instead of the terms *entity*, *process*, and *connector* used in this book. The history of Petri net goes back to Carl Adam Petri who defined this concept in 1962 in his doctoral thesis. Since then, the theory of Petri net has been deeply studied and has grown up. Many software applications have been developed based on Petri net in systems sciences and engineering but not the biological sciences. The first attempt to use Petri net in the biological sciences was made by Reddy et al. [5] in 1993 that used the concept of discrete Petri net for modeling a metabolic pathway.

[5] Reddy VN, Mavrovouniotis ML, Liebman MN. Petri net representations in metabolic pathways. *Proceedings of International Conference on Intelligent Systems for Molecular Biology*. AAAI Press. **1**:328–336, 1993.

There are many papers which directly describe the *Drosophila* circadian rhythm in terms of differential equations. In this case, the models use continuous values for each event. In biological systems, however, there are also some events which may be better considered as discrete events such as the on-off switching of genes. The *hybrid Petri net* is a concept which allows us to combine both continuous and discrete models. The first paper which attempted to use the hybrid Petri net is the paper [6] that successfully modeled the λ-phage genetic switches. A Japanese explanation is also available [7].

[6] Matsuno H, Doi A, Nagasaki M, Miyano S. Hybrid Petri net representation of gene regulatory network. *Pacific Symposium on Biocomputing*. **5**:338–349, 2000.
http://psb.stanford.edu/psb-online/proceedings/psb00/matsuno.pdf

[7] Matsuno H, Drath R, Miyano S. Simulating gene networks with hybrid Petri net. *Advances in Systems Biology* (H. Kitano, Ed.). Springer. pp.165–176, 2001. (in Japanese)

This was a straightforward application of the hybrid Petri net. Later on, while modeling various biological pathways, there appeared some difficulties that prevented us from intuitive understanding of pathways though they could be solved in a mathematical sense. We defined the *Hybrid Functional Petri Net* (HFPN) in order to overcome these problems. Using this, the gene regulatory network of *Drosophila* circadian rhythms and the Fas-induced apoptosis signaling pathway were modeled intuitively [8].

[8] Matsuno H, Tanaka Y, Aoshima H, Doi A, Matsui M, Miyano S. Biopathways representation and simulation on hybrid functional Petri net. *In Silico Biology*. **3**(3):389–404, 2003.
http://www.bioinfo.de/isb/2003/03/0032/

Further, there are a wide range of models that can be created using HFPNs; the following papers show some models:

[9] Matsuno H, Murakami R, Yamane R, Yamasaki N, Fujita S, Yoshimori H, Miyano S. Boundary formation by notch signaling in *Drosophila* multicellular systems: experimental observations and gene network modeling by Genomic

Object Net. *Pacific Symposium on Biocomputing*. **8**:152–163, 2003. (Multicellular gene regulatory network in development)
http://psb.stanford.edu/psb-online/proceedings/psb03/matsuno.pdf

[10] Doi A, Fujita S, Matsuno H, Nagasaki M, Miyano S. Constructing biological pathway models with hybrid functional Petri nets. *In Silico Biology*. **4**(3):271–291, 2004. (Glycolysis with *lac* operon gene regulation)
http://www.bioinfo.de/isb/2004/04/0023/

[11] Matsui M, Fujita S, Suzuki S, Matsuno H, Miyano S. Simulated cell division processes of the *Xenopus* cell cycle pathway by Genomic Object Net. *J. Integrative Bioinformatics*. 0001, 2004. (Gene regulation of the cell cycle)
http://www.jsbi.org/journal/GIW03/GIW03P047.pdf

[12] Doi A, Nagasaki M, Matsuno H, Miyano S. Simulation-based validation of the p53 transcriptional activity with hybrid functional Petri net. *In Silico Biology*. **6**(1-2):1–13, 2006. (Model of p53 transcriptional activity)
http://www.bioinfo.de/isb/2006/06/0001/

[13] Doi A, Nagasaki M, Ueno K, Matsuno H, Miyano S. A combined pathway to simulate CDK-dependent phosphorylation and ARF-dependent stabilization for p53 transcriptional activity. *Genome Informatics*. **17**(1):112–123, 2006. (Model of p53 transcriptional activity)
http://www.jsbi.org/journal/IBSB06/IBSB06F007.pdf

[14] Tasaki S, Nagasaki M, Oyama M, Hata H, Ueno K, Yoshida R, Higuchi T, Sugano S, Miyano S. Modeling and estimation of dynamic EGFR pathway by data assimilation approach using time series proteomic data. *Genome Informatics*. **17**(2):226–238, 2006. (EGFR signaling pathway)
http://www.jsbi.org/journal/GIW06/GIW06F032.pdf

[15] Saito A, Nagasaki M, Doi A, Ueno K, Miyano S. Cell fate simulation model of gustatory neurons with microRNAs double-negative feedback loop by hybrid function Petri net with extension. *Genome Informatics*. **17**(1):100–111, 2006. (Cell fate determination by miRNA)
http://www.jsbi.org/journal/IBSB06/IBSB06F005.pdf

[16] Matsuno H, Inouye ST, Okitsu Y, Fujii Y, Miyano S. A new regulatory interaction suggested by simulations for circadian genetic control mechanism in mammals. *J. Bioinformatics and Computational Biology*. **4**(1):139–153, 2006. (Gene regulatory network of murine circadian rhythm)

However, even this HFPN cannot directly handle sequences and complex data structures. The aforementioned HFPNe solves these problems by introducing *objects*. At the current stage of development of Cell Illustrator, it requires some skills in programming. The specifics of creating a model using HFPNe are dealt with in [3, 4].

In Chapter 6, the gene network of yeast was shown. There have been many papers published on computational methods for computing gene networks from microarray data. The method used for generating this network is based on [17, 18].

[17] Imoto S, Goto T, Miyano S. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric re-

gression. *Pacific Symposium on Biocomputing*. **7**:175–186, 2002.
http://psb.stanford.edu/psb-online/proceedings/psb02/imoto.pdf

[18] Imoto S, Higuchi T, Goto T, Tashiro K, Kuhara S, Miyano S. Combining microarrays and biological knowledge for estimating gene networks via Bayesian networks. *J. Bioinformatics and Computational Biology*. **2**(1):77–98, 2004.

Then in Section 6.1, we explained an intuitive method for organizing and analyzing gene networks. This technology in conjunction with gene network prediction methods is currently being used effectively for drug discovery processes. Results such as finding antibiotic targets using yeast [19, 20] and searching for hyperlipidemia triggers in a large-scale human gene network [21, 22] are encouraging. Further developments are expected to produce even better results.

In addition, the drug discovery approach outlined in [23] is possible in Cell Illustrator through the aforementioned gene network analysis.

[19] Imoto S, Savoie CJ, Aburatani S, Kim S, Tashiro K, Kuhara S, Miyano S. Use of gene networks for identifying and validating drug targets. *J. Bioinformatics and Computational Biology*. **1**(3):459–474, 2003.

[20] Savoie CJ, Aburatani S, Watanabe S, Eguchi Y, Muta S, Imoto S, Miyano S, Kuhara S, Tashiro K. Use of gene networks from full genome microarray libraries to identify functionally relevant drug-affected genes and gene regulation cascades. *DNA Research*. **10**(1):19–25, 2003.

[21] Imoto S, Tamada Y, Araki H, Yasuda K, Print CG, Charnock-Jones SD, Sanders D, Savoie CJ, Tashiro K, Kuhara S, Miyano S. Computational strategy for discovering druggable gene networks from genome-wide RNA expression profiles. *Pacific Symposium on Biocomputing*. **11**:559–571, 2006.
http://helix-web.stanford.edu/psb06/imoto.pdf

[22] Imoto S, Tamada Y, Savoie CJ, Miyano S. Analysis of gene networks for drug target discovery and validation. *Target Discovery and Validation*, Volume 1. (J. Walker and M. Sioud, Eds.). Humana Press. pp.33–56, 2006.

[23] Miyano S, Savoie CJ. Applications of bioinformatics to drug discovery. *Experimental Medicine*. **20**(18):2632–2637, 2002. (in Japanese)

Cell System Markup Language and Cell System Ontology are the basis of Cell Illustrator. These topics are discussed in [24–26].

[24] Jeong E, Nagasaki M, Miyano S. Conversion from BioPAX to CSO for system dynamics and visualization of biological pathway. *Genome Informatics*. **18**:225–236, 2007.
http://www.jsbi.org/modules/journal1/index.php/IBSB07/IBSB07022.pdf

[25] Jeong E, Nagasaki M, Saito A, Miyano S. Cell System Ontology: Representation for modeling, visualizing, and simulating biological pathways. *In Silico Biology*. **7**:0055, 2007.
http://www.bioinfo.de/isb/2007/07/0055/

[26] Nagasaki M, Saito A, Li C, Jeong E, Miyano S. Systematic reconstruction of TRANSPATH data into Cell System Markup Language. *BMC Systems Biology*. **2**:53, 2008.

Pathway parameter search is challenged in [14, 27].

[27] Nagasaki M, Yamaguchi R, Yoshida R, Imoto S, Doi A, Tamada Y, Matsuno H, Miyano S, Higuchi T. Genomic data assimilation for estimating hybrid functional Petri net from time-course gene expression data. *Genome Informatics*. **17**(1):46–61, 2006.
http://www.jsbi.org/journal/IBSB06/IBSB06F008.pdf

Pathway layout algorithms using biological knowledge are devised in [28, 29].

[28] Kato M, Nagasaki M, Doi A, Miyano S. Automatic drawing of biological networks using cross cost and subcomponent data. *Genome Informatics*. **16**(2):22–31, 2005.
http://www.jsbi.org/modules/journal1/index.php/GIW05/GIW05F037.pdf

[29] Kojima K, Nagasaki M, Jeong E, Kato M, Miyano S. An efficient grid layout algorithm for biological networks utilizing various biological attributes. *BMC Bioinformatics*. **8**:76, 2007.
http://www.biomedcentral.com/1471-2105/8/76

Index