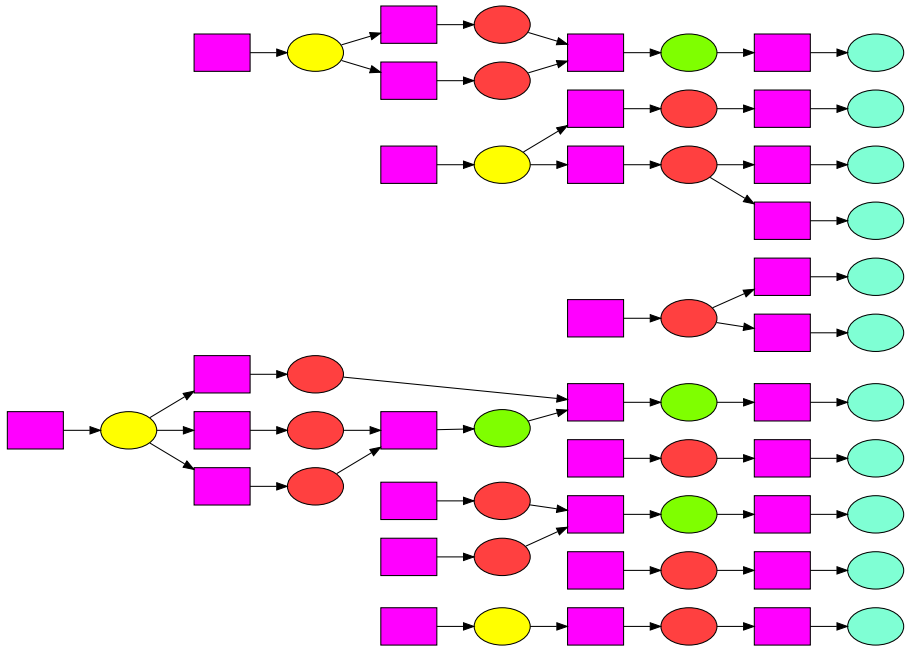
CHALMERS



A probabilistic model for genetic regulation of metabolic networks

Master's Thesis in Complex Adaptive Systems

JONATAN KALLUS
JOEL WILSSON

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, November 2012

A probabilistic model for genetic regulation of metabolic networks

JONATAN KALLUS
JOEL WILSSON

Examiner: DEVDATT DUBHASHI

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover: An illustration of a small part of the regulatory network for *M. tuberculosis*. The organism was mapped by Boshoff et al. [1].

Department of Computer Science and Engineering
Göteborg, Sweden November 2012

Abstract

Recent advancements in gene expression profiling and measurement of metabolic reaction rates have led to increased interest in predicting metabolic reaction rates. In this thesis we present a principled approach for using gene expression profiles to improve predictions of metabolic reaction rates. A probabilistic graphical model is presented, which addresses inherent weaknesses in the current state of the art method for data-driven reconstruction of regulatory-metabolic networks. Our model combines methods from systems biology and machine learning, and is shown to outperform the current state of the art on synthetic data. Results on real data from *S. cerevisiae* and *M. tuberculosis* are also presented.

Acknowledgements

We would like to thank our supervisor PhD student Vinay Jethava, for giving us much guidance in statistical methods, probabilistic modeling and computational approaches to biology. We would also like to thank Professor Devdatt Dubhashi, for additional guidance and support during our work. We are very grateful to both of them for providing us with the main ideas that inspired this thesis, and the many enjoyable discussions. Finally, we would like to thank PhD Intawat Nookaew, for giving us indispensable support within the field of biology and for providing us with useful datasets.

Jonatan Kallus and Joel Wilsson
Göteborg, November 2012

Contents

1 Introduction **1**
- 1.1 Problem . . . 1
- 1.2 Constraint based modeling . . . 2
- 1.3 Our approach . . . 2
- 1.4 Results . . . 3
- 1.5 Conclusion . . . 3
- 1.6 Organisation . . . 3

2 Background **4**
- 2.1 Biological background . . . 4
  - 2.1.1 Gene regulation . . . 4
  - 2.1.2 Metabolic reactions . . . 5
  - 2.1.3 Regulatory networks . . . 5
  - 2.1.4 Gene knockouts . . . 6
- 2.2 Conceptualization of genetic regulation and metabolic systems . . . 6
- 2.3 Probabilistic graphical models . . . 7
  - 2.3.1 Factor graphs . . . 9
  - 2.3.2 Belief propagation . . . 9
  - 2.3.3 The sum-product algorithm . . . 10
  - 2.3.4 Boolean logic in factor graphs . . . 12
  - 2.3.5 Graphical lasso . . . 14
- 2.4 Flux balance analysis . . . 14
- 2.5 Related work . . . 15
  - 2.5.1 Computational approaches for reconstruction of biological networks 16
  - 2.5.2 Regulatory FBA for *S. cerevisiae* . . . 17
  - 2.5.3 Probabilistic regulation of metabolism . . . 18

3 Model **21**
- 3.1 Factor graph formulation . . . 22
- 3.2 Discretization of gene expression data . . . 27

3.3 Integration with FBA 28
3.4 Implementation dependencies 28

4 Experiments **29**
4.1 Synthetic experiment 29
4.1.1 Model details 29
4.1.2 Data generation using Gibbs sampling 30
4.1.3 Results 30
4.2 Descriptions of datasets 32
4.2.1 *E. coli* 33
4.2.2 *M. tuberculosis* 33
4.2.3 *S. cerevisiae* 33
4.3 Results 34
4.3.1 *E. coli* 34
4.3.2 *M. tuberculosis* 35
4.3.3 *S. cerevisiae* 35

5 Conclusion and future work **41**

Bibliography **43**

1

Introduction

The past decade has seen an immense growth in publicly available datasets of gene expression data, such as M3D [2], SGD [3], GEO [4], and ASAP [5]. These datasets have enabled biologists to create genome-scale models of regulatory networks [6], which describe how the genes of an organism interact, and metabolic networks [7], which describe the chemical reactions required to sustain life.

These networks hold the promise of predicting metabolic reaction rates. Accurately modeling these processes is of primary importance to guide the search for viable drug targets and to diagnose metabolic disorders [8].

The structure of the regulatory network has been studied extensively for some organisms, such as *Escherichia coli* (a prokaryotic bacteria) and *Saccharomyces cerevisiae* (an eukaryotic fungi), although even in those cases the regulatory networks are still being updated and new discoveries are made. However, the nature of the interactions between the genes connected in these networks is less clear. Some interactions are stronger than others, but the available networks only contain information about the existence of the interactions and, in many cases, if the interaction is repressing or activating.

1.1 Problem

Gene expression levels and metabolic reaction rates are intimately connected. The genes encode proteins needed for metabolic reactions, and the gene expression level is a measure of how frequently the gene is being transcribed. That is, how much of the protein it is encoding is being produced. If a gene is completely repressed, the reaction rate of metabolic reactions that require the protein that the gene is encoding must be zero.

Which genes are being expressed depends on the environment. For example, genes that encode proteins required only for metabolic reactions that need oxygen will not be expressed in anaerobic conditions. Ideally, measurements of gene expressions would be exact, and there would be no need for a probabilistic approach. Unfortunately,

the measurements are noisy. There is also a problem with the discretization of the continuous valued measurements, so the gene expression values can not be accepted as truth. However, they provide important hints about the most likely combination of metabolic flux rates.

1.2 Constraint based modeling

Rather than trying to model the chemical processes themselves, for which only limited information is available, constraint based modeling methods are based on stoichiometric constraints, thermodynamic constraints, and enzymatic capacity constraints [9].

Flux Balance Analysis (FBA) [10] has been one of the most successful models of metabolism. In this model, the stoichiometric constraints are collected in a stoichiometric matrix $\mathbf{S}$, with one row for each metabolite and one column for each metabolic reaction. An appropriate objective function is also required, of the form

$$f(\mathbf{x}) = \mathbf{c}^\top \mathbf{x}.$$

It is often assumed that the organism will maximize the growth rate, that is, $\mathbf{c}$ is zero for all reactions except the reactions producing biomass.

Given minimum and maximum reaction rate constraints $\mathbf{a}$ and $\mathbf{b}$, respectively, FBA tries to find the reaction rates $\mathbf{x}$ by solving the optimization problem

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{subject to} \quad & \mathbf{S}\mathbf{x} = 0 \\ & a_i \leq x_i \leq b_i \end{aligned}$$

FBA does not use any of the available gene expression data which is now available. Several extensions of the model now exist, which make use of gene expression data, such as regulatory FBA (rFBA) [6], Gene Inactivity Moderated by Metabolism and Expression (GIMME) [11], and Probabilistic Regulation Of Metabolism (PROM) [8].

1.3 Our approach

In this thesis we present a novel way of using gene expression data with FBA, by combining it with *belief propagation*, a method used in machine learning, to predict metabolic reaction rates. Belief propagation is introduced in chapter 2.

Our model has the idea of using the gene expression data to build a probabilistic model in common with PROM, but is able to account for more complex interactions between genes. Additionally, it is able to use the boolean rules of existing regulatory networks [12] in a probabilistic setting.

1.4 Results

Experiments, both on synthetic organisms and on real organisms, have been performed. Results show that our principled approach outperforms PROM on synthetic data, generated from a model based on generally accepted assumptions. In *E. coli* the possibility to learn the underlying structure of the regulatory network is evaluated. We find that a method called the graphical lasso is unable to reconstruct the network, and we conclude that knowledge from the literature about the network structure must be used in order to make predictions on real organisms. An experiment on *M. tuberculosis* show that our method performs similarly, but marginally worse, to the PROM method, when trying to predict lethality of gene knockouts. It has been shown [13] that the PROM algorithm has problems with real world data. *M. tuberculosis* is a case where our method does not improve the results.

Two experiments with *S. cerevisiae* are limited to study the regulatory system in isolation, predicting only gene activation and not growth rate or lethality. It is shown that our approach has significant advantages over PROM in learning the nature of connections between genes from gene expression data. Our method makes better predictions in some cases, and is able to make significant predictions in cases where the PROM method is unable to produce a result.

1.5 Conclusion

Using belief propagation with gene expression data to extend the basic FBA model has been shown to improve the accuracy of predictions of metabolic reactions on synthetic data. It also shows the ability to outperform state of the art methods on data from real organisms in some cases, when trying to predict target gene activation. Specifically, our approach has the ability to make predictions in cases where no predictions are made by other methods. Although the results for these hard to predict cases are far from perfect, they are significantly better than uninformed guesses.

1.6 Organisation

This section concludes the first chapter, which was meant to summarize the contents of this thesis. Chapter 2 gives a thorough overview of the biological terms that is needed to follow the rest of the thesis, and of the ideas that our model is based on. Chapter 3 goes through the method that we suggest for solving the problem defined above. Chapter 4 explains the experiments that have been carried out to evaluate our method and show their results. Lastly, chapter 5 concludes the thesis and suggests how this work can be developed further.

2

Background

This chapter starts by covering some basic terminology in biology, and then continues towards mathematical methods for modeling metabolism. It ends by covering regulatory flux balance analysis and probabilistic regulation of metabolism, the key ideas on which our work relies.

2.1 Biological background

Although this thesis is more about a mathematical model of biology than biology itself, it is important to have some understanding of the underlying biological phenomena. This section intends to provide some details about these phenomena, and some insight into the limitations of the current state of systems biology in general.

Biological systems are astoundingly complex, with many different feedback mechanisms, some of which are poorly understood. All models must must therefore make some simplifications to be useful.

2.1.1 Gene regulation

The DNA sequence of a gene is transcribed into RNA sequences by RNA polymerase.

A subset of the genes in an organism encode proteins. Furthermore, an even smaller subset of these genes can produce gene-regulatory proteins, called transcription factors, that regulate the activity of other protein-encoding genes. The transcription factors bind to target genes where they interact with RNA polymerase, either as activators or repressors, to increase or decrease the rate of transcription of the target gene. In this thesis, we will often simply use "transcription factor" to refer to "the gene transcribing the transcription factor," since the two are inextricably linked. Thus, if we refer to the "transcription factor level" it should be understood to mean "the gene expression level of the gene transcribing the transcription factor."

The rate of transcription of a gene is called the gene expression level, or simply the gene expression. Using microarray technology, gene expressions can be measured in the laboratory, and provide important information about the activity of the cell. In particular, gene expressions can be used to infer what metabolic reactions are likely to be taking place, since the proteins which are transcribed will enter the metabolic network.

2.1.2 Metabolic reactions

Metabolic reactions can be divided into two categories: anabolic reactions and catabolic reactions. Anabolic reactions combine simpler substances to form more complicated ones. For example, for a cell to grow it needs to produce biomass, which requires many different building blocks and consumes energy. Catabolic reactions, on the other hand, break down bigger molecules into their components, and can produce energy to be used in anabolic reactions. The molecules which take part in metabolic reactions are called metabolites.

It is often assumed that the organism will try to optimize its metabolism to produce the maximum amount of biomass. Experiments show that this is often a good approximation in controlled settings with simple organisms placed in a stable environment. However, in more complex organisms, such as mammals, the function of a cell depend on the type of tissue to which it belongs.

2.1.3 Regulatory networks

An organism needs to adapt its metabolism to the environment. In the case of yeast, the metabolic reactions that require oxygen can only take place in aerobic environments. If the oxygen in the environment disappears, the cells need to adjust their metabolism to use different metabolic reactions, which require different kinds of proteins.

The key mechanism behind this regulatory process are the genes controlling transcription factors. Their gene expression levels depend on the metabolites found in the environment, and will therefore change in response to environmental changes. This will in turn affect the target genes controlled by these transcription factors, ensuring that the proteins transcribed will be suited to the new environmental conditions. These connections between the environment and genes, and between genes encoding transcription factors and target genes, form a regulatory network.

This process of regulation is made more complicated by the fact that some transcription factors can affect the gene expression levels of genes encoding other transcription factors. These feedback mechanisms need to be taken into account by our model.

Furthermore, while regulation at the transcriptional level is the most important one, there are also post-transcriptional regulation processes. For example, the DNA double helix in the cell nucleus is enclosed by proteins termed histones, which form a complex called chromatin. This chromatin structure can be altered, so that it is loosened from the DNA, which can expose additional DNA regions for transcription and, consequently, increase the transcription rate.

The significance of post-transcriptional regulation seems to be greater than previously thought, especially in determining the function of cells in different tissues. In this thesis we have only considered simple organisms where all cells have the same function, and our model does not take any post-transcriptional processes into account.

2.1.4 Gene knockouts

One of the most important tools for finding out which target genes are being regulated by different transcription factors are gene knockout experiments. By constructing a mutated version of a gene and introducing it into an embryonic stem cell, the gene will be recombined with this mutated version. The resulting gene will, instead of being transcribed into its usual protein, either produce a nonfunctional protein, or not be transcribed at all.

By conducting experiments with knockout strains and comparing the gene expression levels with the unmodified strain under identical conditions, it is possible to see which target genes were affected by the knockout, indicating a relation between the knocked out transcription factor gene and the target genes.

2.2 Conceptualization of genetic regulation and metabolic systems

In computational systems biology the complex process in which a part of DNA sequence, a gene, facilitate the creation of proteins which, in turn, affect the functionality of other genes or cell functions, is often simplistically modeled as a gene that is active or not. Abstractly, genes are thought of as entities that can be either turned on or turned off. We adopt this abstraction in the remainder of this thesis. Further, when this abstraction has been made, it is straight forward to think of these abstract genes as nodes in a graph. The graph then captures gene interactions.

Continuing this procedure of abstraction, the metabolic reactions can be thought of as forming a network, where one reaction's output is used as input in another metabolic reaction. Only some reactions take up metabolites from the environment. In the steady state of no growth, the metabolites entering the cell must be equal to the metabolites leaving the cell. As we will see in the following sections, this network of metabolic dependencies is often captured with a matrix, called the stoichiometric matrix. To exemplify, let a metabolic system consist of two reactions called 1 and 2, and four metabolites called A, B, C and D. With the reactions being

$$A \leftrightarrow B + C$$
$$B + 2C \leftrightarrow D$$

the system can be described by the stoichiometric matrix

	1	2
A	-1	
B	1	-1
C	1	-2
D		1

which captures the information that in order for reaction 2 to create D, reaction 1 must create B and C. Assuming that the goal is to produce D, it also shows that metabolite C will be limiting in this system, while B will be available in excess. Section 2.4 will explain how this representation is useful.

Following Chandrasekaran et al. [8] it makes sense to think about metabolic reactions as being probabilistically turn on or off. That is, instead of thinking of a reaction as either ongoing or stopped, it makes sense to think of it as being ongoing with a certain probability. Since all biological experiments are conducted not on a single cell, but on large colonies of cells, a reaction may be active in some fraction of cells and stopped in the others.

Further, immense studies of organisms have been performed to find out which genes that respond to which environmental changes and how. This information is conceptualized as boolean rules. So if, for example, a gene activates only when oxygen and glucose is available in the environment, this information is captured as

```
Gene A := oxygen AND glucose
```

Similarly, the way in which some metabolic reactions only proceed when certain genes are active is captured with boolean rules.

2.3 Probabilistic graphical models

Our model is based on Bayesian statistics, the core of which is Bayes' theorem,

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

$P(A|B)$ is called the posterior, $P(B)$ the evidence and $P(A)$ the prior. Noting that for all A, the value of $P(B)$ is fixed, we can write

$$P(A|B) \propto P(B|A)P(A).$$

It is easy to extend this rule to more general statements about probability, which allows us to factorize probabilistic statements using joint probability distributions. For example, we have that $P(A, B, C) \propto P(A|B, C)P(B, C) \propto P(A|B, C)P(B|C)P(C)$.

Table 2.1: Examples of joint probability tables.

TF_1	G_1	P
0	0	1.0
1	0	0.0
0	1	0.0
1	1	1.0

TF_2	TF_3	G_2	P
0	0	0	0.25
1	0	0	0.05
0	1	0	0.05
1	1	0	0.0
0	0	1	0.05
1	0	1	0.15
0	1	1	0.10
1	1	1	0.35

G_1	G_2	R_1	P
0	0	0	1.0
1	0	0	1.0
0	1	0	1.0
1	1	0	0.0
0	0	1	0.0
1	0	1	0.0
0	1	1	0.0
1	1	1	1.0

These factorizations can be represented as graphs, where the variables are nodes of the graph and the dependencies created by the conditional probabilities are directed edges.

Consider the following scenario: a reaction rate R depends on proteins transcribed by G_1 and G_2. These are in turn controlled by a single transcription factor, TF_1. Given the probability that TF_1 is turned on, what is the probability of reaction R_1 having flux equal to its maximum rate? To answer this question, we need to know something about the relationships between the transcription factor and the genes, and between the genes and the reaction. The factorization of the joint probability distribution in this case is

$$P(TF_1, G_1, G_2, R1) \propto P(R_1|G_1, G_2)P(G_1|TF_1)P(G_2|TF_1)P(TF_1),$$

and we need to have $P(G_1|TF_1)$, $P(G_2|TF_1)$ and $P(R_1|G_1, G_2)$ available.

These functions can be summarized in joint probability tables, which can be used to compute the probability that one variable is in a certain state, conditional on known values of all the other variables. In this example, we will use the values shown in table 2.1. From these tables we can see that $P(G_2 = 1|TF_1 = 1) = 1.0$, and so on. Note that the probabilities in these tables do not need to be normalized, since normalization can always be done at the end.

The probabilities for R_1 in this case is a boolean `AND` rule. The reaction can thus only go ahead at full speed if both G_1 and G_2 are turned on. Using these tables, and setting $P(TF_1 = 1) = 1$, we can compute the conditional probability $P(R_1|TF_1 = 1)$.

$$P(G_1 = 1|TF_1 = 1) = 1.0$$

$$P(G_2 = 1|TF_1 = 1) = 0.8$$

$$P(R_1 = 1|TF_1 = 1) = \frac{P(R_1 = 1, TF_1 = 1)}{P(TF_1 = 1)} = P(R_1 = 1, TF_1 = 1) \tag{2.1}$$

$$= \sum_{x,y \in 0,1} P(R_1 = 1|G_1 = x, G_2 = y, TF_1 = 1) P(G_1 = x, G_2 = y, TF_1 = 1) \tag{2.2}$$

$$= \sum_{x,y \in 0,1} P(R_1 = 1|G_1 = x, G_2 = y, TF_1 = 1) P(G_1 = x|TF_1 = 1) P(G_2 = y|TF_1 = 1) \tag{2.3}$$

$$= \sum_{x,y \in 0,1} P(R_1 = 1|G_1 = x, G_2 = y) P(G_1 = x|TF_1 = 1) P(G_2 = y|TF_1 = 1) \tag{2.4}$$

$$= P(R_1 = 1|G_1 = 1, G_2 = 1) P(G_1 = 1|TF_1 = 1) P(G_2 = 1|TF_1 = 1) \tag{2.5}$$

$$= 1.0 \cdot 0.7 \cdot 1.0 = 0.7. \tag{2.6}$$

By putting the variables as nodes in a graph and drawing directed edges according to the dependencies, we get the graph shown in figure 2.1. Such directed acyclic graphs are an important subclass of probabilistic graphical models, called Bayesian networks.

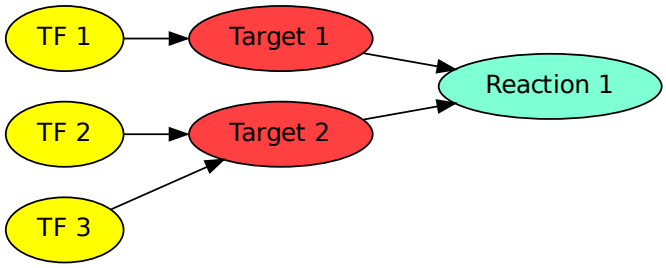


Figure 2.1: Bayesian network of the regulatory network in section 2.3.

2.3.1 Factor graphs

A Bayesian network can be converted into a bipartite graph called a *factor graph*, where additional nodes are inserted in-between all connected nodes in the Bayesian network. These *factor nodes* encode the joint probability tables describing the dependencies. Factor nodes connected only to a single variable can be used to model prior beliefs; its probability table is then just the probabilities of the variable's possible states.

2.3.2 Belief propagation

Belief propagation is an algorithm that can perform exact inference on Bayesian networks [14]. It has been shown to be a special case of a more general algorithm, called the *sum-product algorithm*.

The sum-product algorithm is the reason for building factor graphs in the first place. It can be formulated as a message-passing algorithm on a factor graph, where messages are sent between the nodes.

2.3.3 The sum-product algorithm

Remarkably, it has been shown that many famous algorithms, such as Kalman filters, the Fast Fourier Transform, turbo codes, and belief propagation are instances of the sum-product algorithm [15]. As its name implies, the algorithm has two parts, involving sums and products. For messages sent from variable nodes to factor nodes, we have

$$\mu_{x\to f}(x) = \prod_{h\in n(x)f} \mu_{h\to x}(x)$$

and for messages sent from a factor node to a variable node,

$$\mu_{f\to x}(x) = \sum_{x} \left(f(x) \prod_{y\in n(f)x} \mu_{y\to f}(y) \right)$$

where $n(f)$ is the set of arguments of the function f associated with the factor node.

Function f is defined by the joint probability tables that we have specified. A node sends a message to a neighbor after receiving messages from all its other neighbors, and the message sent is a summary of those received messages. Nodes with only one neighbor, like the factors encoding our prior probabilities in the example above, can thus send a message at the start of the algorithm. Also, no sums or products are needed by nodes having only two neighbors, as they will simply apply their function f to the message received from one neighbor and send the result to the other neighbor.

Finally, variable nodes with only one neighbor will send an identity message to their neighbor at the start of the algorithm.

As an example, using the network above, if we are absolutely certain that TF_1 is active, TF_2 is inactive, and have no information about TF_3, the following messages would be sent, as shown in figure 2.2.

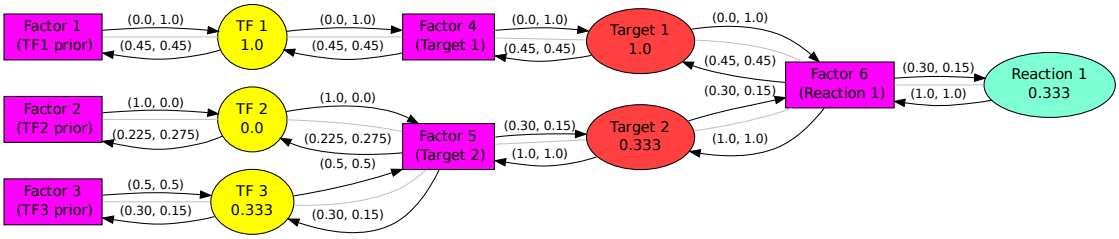


Figure 2.2: Messages sent, and probabilities computed, by the sum-product algorithm.

1. Messages are sent from the factor nodes encoding our priors to the variable nodes of TF_1, TF_2, and TF_3. Since each transcription factor is a binary variable, being either on or off, the messages will contain two values, representing the probability of either states. If we are absolutely sure that TF_1 is on, TF_2 is off, and we have no information about the state of TF_3, the messages sent from the factor nodes would be $(0.0, 1.0)$, $(1.0, 0.0)$, and $(0.5, 0.5)$ for TF_1, TF_2 and TF_3, respectively. Additionally, an identity message of $(1, 1)$ is sent from the leaf variable node R_1.

2. The transcription factor nodes will simply pass on the messages received to the factor nodes connected to the genes.

3. The factor nodes between the transcription factors and the target genes will apply their functions, specified by the joint probability tables, to the incoming messages. The factor node between TF_1 and G_1 will simply pass on the message, while the factor node between the transcription factors and G_2 will first use the sum-product algorithm above to compute its message to G_2:

$$\mu_{f_5 \to G_1} = \sum_{TF_2, TF_3 \in \{0,1\}} f_5(TF_2, TF_3, G_1) \mu_{TF_2 \to f_5}(TF_2) \mu_{TF_3 \to f_5}(TF_3) \tag{2.7}$$

$$= \begin{pmatrix} 0.25 \cdot 1.0 \cdot 0.5 \\ 0.05 \cdot 1.0 \cdot 0.5 \end{pmatrix} + \begin{pmatrix} 0.05 \cdot 0.0 \cdot 0.5 \\ 0.15 \cdot 0.0 \cdot 0.5 \end{pmatrix} \tag{2.8}$$

$$+ \begin{pmatrix} 0.05 \cdot 1.0 \cdot 0.5 \\ 0.10 \cdot 1.0 \cdot 0.5 \end{pmatrix} + \begin{pmatrix} 0.0 \cdot 0.0 \cdot 0.5 \\ 0.35 \cdot 0.0 \cdot 0.5 \end{pmatrix} = \begin{pmatrix} 0.30 \\ 0.15 \end{pmatrix} \tag{2.9}$$

4. The variable nodes G_1 and G_2 will simply pass on the messages received from the factor nodes to their right towards the factor node on the left.

5. The factor node between the gene variables and R_1 will also apply the sum-product algorithm messages from G_1 and G_2 to compute the message to be sent to R_1. Similarly, a message to G_1 will be computed using messages from G_2 and R_1, and a message to G_2 will be computed using messages from G_1 and R_1.

$$\mu_{f_6 \to R_1} = \sum_{G_1, G_2 \in \{0,1\}} f_6(G_1, G_2, R_1) \mu_{G_1 \to f_6}(G_1) \mu_{G_2 \to f_6}(G_2) \tag{2.10}$$

$$= \begin{pmatrix} 1.0 \cdot 0.0 \cdot 0.30 \\ 0.0 \cdot 0.0 \cdot 0.30 \end{pmatrix} + \begin{pmatrix} 1.0 \cdot 1.0 \cdot 0.30 \\ 0.0 \cdot 1.0 \cdot 0.30 \end{pmatrix} \tag{2.11}$$

$$+ \begin{pmatrix} 1.0 \cdot 0.0 \cdot 0.15 \\ 0.0 \cdot 0.0 \cdot 0.15 \end{pmatrix} + \begin{pmatrix} 0.0 \cdot 1.0 \cdot 0.15 \\ 1.0 \cdot 1.0 \cdot 0.15 \end{pmatrix} = \begin{pmatrix} 0.30 \\ 0.15 \end{pmatrix} \tag{2.12}$$

$$\mu_{f_6 \to G_1} = \sum_{G_2, R_1 \in \{0,1\}} f_6(G_1, G_2, R_1) \mu_{G_2 \to f_6}(G_2) \mu_{R_1 \to f_6}(R_1) \tag{2.13}$$

$$= \begin{pmatrix} 1.0 \cdot 0.30 \cdot 1.0 \\ 1.0 \cdot 0.30 \cdot 1.0 \end{pmatrix} + \begin{pmatrix} 1.0 \cdot 0.15 \cdot 1.0 \\ 0.0 \cdot 0.15 \cdot 1.0 \end{pmatrix} \tag{2.14}$$

$$+ \begin{pmatrix} 0.0 \cdot 0.30 \cdot 1.0 \\ 0.0 \cdot 0.30 \cdot 1.0 \end{pmatrix} + \begin{pmatrix} 0.0 \cdot 0.15 \cdot 1.0 \\ 1.0 \cdot 0.15 \cdot 1.0 \end{pmatrix} = \begin{pmatrix} 0.45 \\ 0.45 \end{pmatrix} \tag{2.15}$$

$$\mu_{f_6 \to G_1} = \sum_{G_1, R_1 \in \{0,1\}} f_6(G_1, G_2, R_1) \mu_{G_1 \to f_6}(G_1) \mu_{R_1 \to f_6}(R_1) \tag{2.16}$$

$$= \begin{pmatrix} 1.0 \cdot 0.0 \cdot 1.0 \\ 1.0 \cdot 0.0 \cdot 1.0 \end{pmatrix} + \begin{pmatrix} 1.0 \cdot 1.0 \cdot 1.0 \\ 0.0 \cdot 1.0 \cdot 1.0 \end{pmatrix} \tag{2.17}$$

$$+ \begin{pmatrix} 0.0 \cdot 0.0 \cdot 1.0 \\ 0.0 \cdot 0.0 \cdot 1.0 \end{pmatrix} + \begin{pmatrix} 0.0 \cdot 1.0 \cdot 1.0 \\ 1.0 \cdot 1.0 \cdot 1.0 \end{pmatrix} = \begin{pmatrix} 1.0 \\ 1.0 \end{pmatrix} \tag{2.18}$$

6. The messages received by the variable nodes G_1 and G_2 are sent unchanged to f_4 and f_5, respectively.

7. Messages $\mu_{f_4 \to TF_1}$, $\mu_{f_5 \to TF_2}$ and $\mu_{f_5 \to TF_3}$ are computed similarly.

8. Finally, messages received by the transcription factor nodes from the right are sent unchanged to their prior factors.

For each variable node, the probabilities of its states can be computed as the product of any two messages sent in opposite directions over an edge connected to the variable node, and normalizing the result so that the sum of the probabilities over the variable's different states is equal to one.

Hence, the probability for TF_3 being expressed is

$$P(TF_1 = 1) = (0.5 \cdot 0.3)/(0.5 \cdot 0.3 + 0.5 \cdot 0.15) \approx 0.667.$$

The probabilities for other genes being expressed, and for the reaction rate being the maximum rate possible, are shown in figure 2.2 inside the variable nodes.

Note that the probability for TF_3 is different from its prior of 0.5, because with the given joint distributions and priors for TF_1 and TF_2 it is more likely to be suppressed than to be expressed.

2.3.4 Boolean logic in factor graphs

In metabolic networks there are often boolean rules specifying the interactions between genes and metabolic reactions. For example, in the consensus yeast model [16], the gene YDR297W is controlled by the boolean expression

```
(YBR126C and YDR074W and (YML100W or YMR261C))
```

As demonstrated above, belief propagation gives us a structured way of calculating the probability of YDR297W being on given uncertain observations of the genes in this boolean rule. This is done by connecting the genes involved in the boolean rule to a factor node, which is also connected to YDR297W, and specifying the appropriate joint probability table representing the boolean rule. The rule above can thus be implemented by the factor graph shown in figure 2.3. More details about reasoning under uncertainty using boolean logic rules can be found in [17].

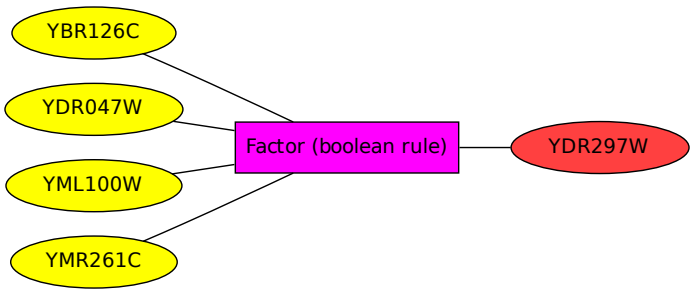


Figure 2.3: Factor graph of a boolean logic rule involving several arguments, using one factor node.

Just like the joint probability tables for factors connected to many transcription factors grow exponentially in size, so do the joint probability tables of factor nodes representing the boolean rules. However, the values of different subrules, such as `YBR126C and YDR074W` and `YML100W or YMR261C`, are independent. Therefore, without any loss of precision, we can transform the boolean rule to the network shown in figure 2.4, by introducing dummy variables nodes representing the values of the subrules, and then combine these values to get the probability of the boolean expression being true.

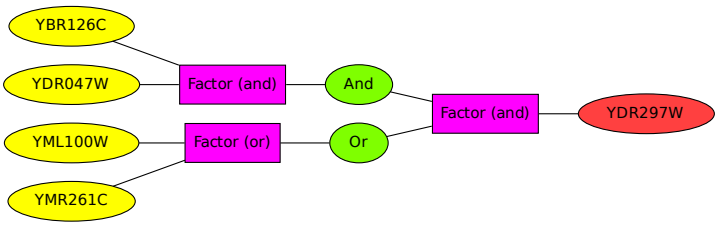


Figure 2.4: The same boolean rule as in figure 2.3, represented by a factor graph containing one factor node for each subrule.

This conversion is not unique, since `A and B and C` can be written as `(A and B) and C` or `A and (B and C)`. In the first case, there would be a dummy variable node representing the probability of `A and B` being true, while in the second case there would be a dummy variable node representing the probability of `B and C` being true. However, and again due to the independence between different subrules, both graphs will compute the same value for the same inputs.

2.3.5 Graphical lasso

The *graphical lasso* [18] is an algorithm used to estimate sparse graphs by using an l_1-regularization term (called a "lasso" penalty, for least absolute shrinkage and selection operator; see [19]) applied to the inverse covariance matrix. This is relevant for regulatory networks, since these consist of sparse graphs. The sparsity is enforced by the penalty term.

Briefly, the graphical lasso is a fast algorithm to maximize

$$\log\det\Theta - \mathrm{tr}(S\Theta) - \rho||\Theta||_1$$

over all non-negative definite matrices Σ, where ρ is the strength of the lasso penalty, $\Theta = \Sigma^{-1}$ where Σ is the covariance for multivariate variables of dimension p, and S is the empirical covariance matrix for N observations of the variables.

The graphical lasso is based on the models for static reconstruction that are discussed in section 2.5.1. We use the gene expression profiles to compute the empirical covariance matrix S. The resulting Σ is then the estimated regulatory network.

In section 4.3.1 we present an experiment that showed that the results found using the graphical lasso on the metabolic regulatory network of *E. coli* were poor, by comparing them to knowledge from experiments. Therefore, and since some knowledge about which genes that are interacting in the regulatory networks of the model organisms is available, our model does not use the methods presented here. Instead, our model is based on the methods presented by Chandrasekaran et al presented in section 2.5.3.

2.4 Flux balance analysis

Flux balance analysis (FBA) is a mathematical approach for analyzing the flow of metabolites through a metabolic network [10]. FBA is a subgroup of constraint-based reconstruction and analysis (COBRA) methods. A metabolic network is represented as a graph with metabolites (substances) as nodes and co-occurrence in metabolic reactions as edges. By assuming that the system is in a steady state, the constraint that all metabolite concentrations should be constant can be imposed. In other words, each metabolite must be consumed at a rate that is equal to the rate it is produced. In addition, the minimum and maximum allowable fluxes are constrained for each reaction. The assumption of steady state makes this approach unnatural for dynamic (time-varying) systems, although attempts to extend it to capture dynamic systems have been made.

Mathematically the system is described using basic linear algebra. All reactions are captured in a stoichiometric matrix **S** with each row representing one metabolite and each column representing one reaction. Fluxes through each reaction is represented by the vector **v**. The first and second constraints mentioned in the previous paragraph are imposed by

$$\mathbf{Sv} = 0$$
$$a_i < v_i < b_i$$

where **a** and **b** are the upper and lower reaction rate bounds. The intent of the model is to predict all or some of the reaction rates given internal constraints (stoichiometric matrix) and external constraints (bounds on reaction rates), thus **v** holds the prediction of the model. With these constraints imposed **v** is constrained to an allowable solution space, but **v** is not uniquely determined. In FBA, an objective function $\mathbf{Z} = \mathbf{c}^T\mathbf{v}$ is chosen, making **v** uniquely determined and findable by solving a linear optimization problem. It is common to use growth rate as the objective function, by letting **c** be one in the element corresponding to the biomass equation and zero in all other elements.

Several attempts have been made to enhance the FBA model with gene expression data and knowledge about the regulatory network (that is, the influence of gene expression via enzymes on metabolic reactions). Covert et al. [6] propose an extension of FBA where boolean rules are defined to further constrain the allowable solution space. Boolean rules are defined, that require certain enzymes or regulatory proteins to be present/absent for certain reactions to take place.

They argue for the validity of a binary classification of reactions as either active or inactive and controlled via boolean logic equations by stating that transcriptional regulation generally happens at a time scale that is an order of magnitude slower than the time scales of metabolism. On the other hand Lewis et al. [20] write that the use of binary rules are one of the main weaknesses of this model since they assume binary responses for all transcription-regulatory interactions, when in fact transcription regulation ranges from binary to continuous in real biological systems.

Covert et al. [6] also propose a dynamic FBA model by assuming a quasi-steady-state at each time step and letting the presence of enzymes or regulatory proteins vary over time.

2.5 Related work

Study of genetic and metabolic systems is a major subject within the fields of computational biology, systems biology and machine learning. It combines decades of manual study of model organisms in biological laboratories with statistical methods for understanding the big amount of data that is recorded by modern gene sequencers and DNA microarrays. This chapter presents a few of the most influential concepts within the subject. First, the way that these systems are modeled is described. Thereafter an

overview of methods for understanding genetic networks is presented. The two last sections describe the implementation of regulatory FBA for *S. cerevisiae* and the current state-of-the-art method for data-driven reconstruction of regulatory networks, respectively.

2.5.1 Computational approaches for reconstruction of biological networks

From a computational viewpoint, the main difficulty for inference of gene regulatory networks is the relatively small number of observations compared to the number of unknown variables [21]. This section will provide an overview of some key methods for reconstruction of genetic networks by using observations of gene expression levels. Reconstruction aims at finding which genes that interact and the strength of their interaction.

The Gaussian Graphic Model (GGM) is intended for reconstruction of static networks and assigns high weights to edges that connect two genes with highly correlated observations. GGM models gene expressions as being drawn from a multivariate Gaussian distribution. Then its covariance matrix is the inverse of the edge weight matrix. This model leads to a closed form expression for the log-likelihood of observations. The edge weight matrix that minimizes the log-likelihood of the observed gene expression levels gives the resulting reconstruction. Since the minimization is equivalent to a convex optimization problem, solutions can be found in a reasonable amount of time even for networks consisting of thousands of nodes. Because of noise in gene expression data it is often better to use discretized values of expression levels and edge weights. In the discretized case, this model is equivalent to the Ising model. [22]

There are two main problems with both methods mentioned in the previous paragraph. First, since the number of genes is high and the number of observations is low, they fail to give significant estimations of the interaction strengths. Second, it is known that the interaction network of genes is sparse, yet they give non-sparse solutions. Least absolute shrinkage and selection operator (the "lasso")[19], or l_1-regularization, mitigates these problems. Essentially, it adds a term to the closed form expression from GGM which punishes non-sparsity. This leads to the graphical lasso which was described in section 2.3.5. [22]

All models discussed so far in this section handles the observations of gene expression profiles as independent and identically distributed samples. These models have no concept of time and cannot capture changes in the network over time. Methods for reconstruction of time-varying regulatory networks are divided into two categories, optimization-based methods and model-based methods. Optimization-based methods enhance the l_1-regularization methods previously mentioned by asserting that big changes of the interaction network in short time periods have a low probability. Specifically, a kernel function is introduced which models how slowly the network changes. Model-based methods assume that the probability of network changes depends on motifs for the network, such as decreasing density, avoiding fast changes and increasing transitivity, or, as in NETGEM [23], on functional roles of the genes. [22]

The combination of dynamic reconstruction of regulatory network with metabolic

network modeling lies outside the scope of this thesis, and no such models have been presented to our knowledge.

2.5.2 Regulatory FBA for *S. cerevisiae*

In section 2.4 the theory of regulatory flux balance analysis (rFBA) was discussed. This section discusses the first implementation of rFBA for a model of *S. cerevisiae*. In their paper [24] Herrgård et al. present a model that allows for integration of boolean rules for three kinds of regulatory networks with gene expression data. Boolean rules are compiled from primary literature for external metabolite concentrations affecting TF gene expression (and some non-TF gene expressions), genes affecting genes and genes affecting metabolic reaction rate. This division into three layers makes it possible to compare model predictions with experimental data both for gene expression predictions and growth rate predictions. Predicted gene expressions and growth rates are compared with experimental data for 10 different strains and for several environmental conditions. In addition to implementing rFBA for an eukaryotic cell, a method for using mispredictions to refine the boolean rules is proposed.

The model for the regulatory network that is used and presented is called iMH805/775, using the naming convention MH for Markus Herrgård, 805 is the number of genes accounted for and 775 is the number of regulatory interactions in the model. The iMH805/775 model is combined with the stoichiometric matrix for yeast using the framework of rFBA. In entirety, a model consisting of three layers is composed, each layer is connected with the next via a set of boolean rules. It takes 67 extracellular and 15 intracellular metabolite concentrations as input signals. The first layer consists of 55 transcription factor genes which can be activated by the input signals via boolean rules, but also activated or repressed by each other. The middle layer consists of 348 metabolic genes (genes that have been found to influence metabolism). Primarily, these genes are controlled by the first layer via boolean rules, but some genes are instead controlled directly by the input signals. Boolean rules between the first and middle layer were found by Harbison et al. [12]. Direct connections from input signals to metabolic genes are not biologically viable; these connections are explained as capturers of unknown TF activity. To be clear, out of the 805 genes in the model 55 are transcription factors, 348 are metabolic genes that are used and the rest are metabolic genes that are not used in this model. The third layer is the metabolic reactions. They are controlled by the middle layer via boolean rules, primarily compiled by Duarte et al. [25]. In entirety, the N metabolic reaction are controlled by composite boolean functions

$$y_i = f_i(X^{\text{ext}}) \tag{2.19}$$

where $i = \{1,...,N\}$ and X^{ext} is a boolean vector of 82 input signals. In addition to the 82 input signals, there may also be rules that depend on whether specific intracellular fluxes have non-zeros values. As proposed by rFBA, the upper and lower bounds of the reaction rate of reaction i is set to zero if $f_i(X^{\text{ext}}) = 0$. Unlike Covert et al. [6], only two rFBA time steps are performed in each experiment. First one with all reactions

activated to find intracellular fluxes to use as input, and then one to find the output from the model.

The novel contribution by Herrgård et al., in addition to being the first implementation of rFBA for an eukaryotic cell, is the proposal of a method for systematic expansion of the regulatory network by comparing experimental gene expressions with predicted gene expressions. The initial model predicts few gene expressions that are not seen in vivo (few false positives), but it fails to predict many interactions from observed gene expression changes (many false negatives). In each experiment, false negatives that were strongly supported in experimental measurements were chosen as candidate new target genes. Then the initial network from *i*MH805/775 was combined with a provisional network from [12], forming a directed graph with genes as nodes and edges labeled as either up- or downregulating. In this combined network, label-coherent paths of length shorter than 6 from TF genes to candidate target genes was found using breadth first search. The found paths were manually integrated into the model if they were not in conflict with existing rules and they increased the overall predictive power. The extended model is called *i*MH805/837, since it was extended with 62 additional regulatory interactions.

In the refinement of the model 10 simulations are conducted and compared with gene expression data for 10 different strains. For the growth rate prediction 132 simulations are conducted (12 different carbon sources combined with the wild type strain and 10 TF knockout strains) and compared with in vivo experiments conducted by the authors.

Flux balance analysis depends heavily on the rates that a cell strain is able to consume key external metabolites. Unfortunately, these rates were not experimentally measured. Therefore, these parameters were estimated to make the model fit growth rates from the experiments. A genetic algorithm was used for this purpose.

Regulatory FBA for *S. cerevisiae* shows impressive results in growth rate prediction, and a data-driven method for model improvement is presented. On the other hand, these results are impossible to reproduce in any less studied organism. The presented method relies on decades of manual study of regulatory and metabolic systems in *S. cerevisiae*, the entire model was built by studying a vast amount of literature to compose it into a coherent set of rules. To be able to model other organisms, without spending decades on researching that specific organism, much more emphasis must be put on data-driven model construction. A related issue is that rFBA does not allow for uncertainty. Binary states and binary responses works for an extremely well studied organism, but for data-driven approaches various levels of certainty must be handled.

2.5.3 Probabilistic regulation of metabolism

Noting the shortcomings of Regulatory FBA, rFBA, Chandrasekaran et al. [8] propose a probabilistic model to replace the boolean rules with probabilities. An advantage of this method, compared to rFBA, is that it is data-driven and allows for non-binary regulation. Rather than turning a protein on or off based on gene expression, it changes the bounds for reactions that depend on the protein.

The method presented is called probabilistic regulation of metabolism (PROM). It requires a genome-scale metabolic network, a regulatory network structure, abundant gene

expression data, and additional interactions involving enzyme regulation by metabolites and proteins.

Using a large amount of data (measurements for 1875 growth phenotypes in the case of *E. coli*), marginal probabilities involving a target gene and a transcription factor, such as $P(G_1 = 1|TF_1 = 0)$, where both genes and transcription factors (TFs) are binary variables, are estimated. This is done simply by counting

$$p = P(G_1 = 1|TF_1 = 0) = \frac{N(G_1 = 1, TF_1 = 0)}{N(TF_1 = 0)}. \tag{2.20}$$

The purpose of PROM is to investigate and predict the effects of various transcription factor knockouts. Hence, a regulatory network is required to determine which target genes are affected by a knockout. For each of these genes, the reaction bounds are changed based on the probabilities computed as above. Note that the regulatory network contains boolean rules for how target genes are controlled by a transcription factor, and can therefore contain a rule such as `TF_1 or TF_2` for a reaction R_1. If only a single knockout is done at a time, the flux bounds for R_1 would never be changed. This is because all transcription factors except for the one that was knocked out are considered to be expressed.

The authors argue that the flux through the reaction regulated by gene 1 would not exceed the maximum flux possible through the reaction (V_{max}) if gene 1 is expressed, and would be zero if gene 1 is not expressed. The genes affected by the knockout are sequentially processed to determine reaction bounds for each reaction controlled by the gene. In [8] it is implicitly assumed that the reaction is only regulated by one gene, the activity of which alone determines the flux bounds, independent of all other genes. Although not stated in the paper, the implementation uses the minimum/maximum value of the upper/lower flux bounds over all genes associated with a reaction.

The probabilities for each gene being expressed are used to change the flux bounds for reactions controlled by the gene. The bounds specified in the genome-scale metabolic models are usually very loose, and growth is constrained by setting the flux rates for external metabolites according to the environment. To get the minimum and maximum flux bounds of a reaction under the specific conditions, flux variability analysis (FVA) [26] is used. It is these lower/upper bounds which are modified using the maximum/minimum probabilities of the associated genes.

FVA is a simple variation of FBA, where the objective function is changed to minimize or maximize the flux rate through the reaction being considered.

Simply modifying the bounds given by FVA can lead to an optimization problem with no solution. Non-negative slack variables are therefore subtracted/added to all lower/upper flux bounds, to allow the reaction rates to be below or above the bounds given by FVA and PROM. The slack variables are included in the objective function, and given a negative weight, to discourage them from being non-zero. This relaxed problem will always have an optimal solution, although that optimal solution may be zero. In PROM, the slack penalty is denoted by κ and set to 1.

The results in [8] are slightly better than rFBA as presented by [6]. This is claimed to be a significant improvement, since Covert et al. had to manually add boolean rules to

the regulatory network to get their results. On the other hand, the PROM algorithm did not require any changes to the regulatory network, but used only probabilities derived from high-throughput data.

PROM converts the gene expression profiles into discrete values (genes are either on/active or off/inactive) by considering genes with expression values above the 0.33 quantile as on, and the rest as being off. Thus, in each profile, 33% of the genes are off.

Equation 2.20 reveals two limitations of PROM that our model addresses; predictions are only made locally in the regulatory network and target genes are predicted independently of each other. Predictions are local in the sense that the probability of a gene being turned on is only predicted if the gene is directly connected to the knocked out transcription factor. This weakness is also related to the fact that PROM does not handle an arbitrary number of knockouts at the same time. (Although not mentioned in [8], the implementation shows that double knockouts can be handled, but in a simplistic fashion that does not take transcription factor interaction into account.) The algorithm implicitly assumes that transcription factors that are not knocked out are active, and that target genes that are not directly connected to the knocked out transcription factor are also active. The next chapter presents a model where *beliefs* can *propagate* through the network, so that predictions, although with less certainty, can be made for genes further away from knocked out genes, and, in fact, for the entire network. The other limitation, independent prediction of target gene activity, means that although predictions are made for several target genes that are connected to the same transcription factor, PROM does not take into account the possibility of activities of target genes being correlated with each other.

3

Model

Probabilistic and data-driven modeling of genetic networks is well studied [22][23] and we summarize the subject in section 2.5.1. Several books [27] and papers [20] have been written about modeling of metabolic systems with constraint-based linear optimization. Complex models of the metabolic systems of the model organisms exist and are able to make good phenotype predictions [28][24]. It is known that it is the genetic network that regulate metabolic reactions in response to environmental conditions [6] and successful attempts have been made for the well studied model organisms to model this regulation manually by immense literature studies [24]. Yet, to our knowledge, PROM [8] is the only attempt to combine probabilistic and data-driven modeling of genetic networks with constraint-based modeling of metabolic systems as an optimization problem. In section 2.5.3 we argue that the approach of [8] makes assumptions of independence that are unrealistic, unnecessary and limiting to the model's predictive ability. Our main contribution in this thesis is to take a principled approach to the combined modeling of regulatory and metabolic systems by modeling the regulation of metabolism as a factor graph. By using this framework, which is a type of graphical model well studied in probability theory [29], we can handle environmental, intergenetic and regulatory interactions in a probabilistic and unified way. This framework also allows us to use sets of gene expression data in concert with knowledge from literature in a more effective way than PROM.

In regulatory-metabolic models, organisms are conceptually modeled as a set of rules for how transcription factor genes react to environmental conditions, a network for how genes interact, a set of rules for how target genes control metabolic reactions and, lastly, a network of metabolic reactions. Figure 3.1 shows these four steps as the topmost part of the figure. In systems biology, these four parts are represented as boolean rules, a graph and a stoichiometric matrix, the second row of figure 3.1. As mentioned in the previous paragraph, we propose the first three parts to be modeled jointly as a factor graph, the output of which can be fed to the optimization problem defined in FBA.

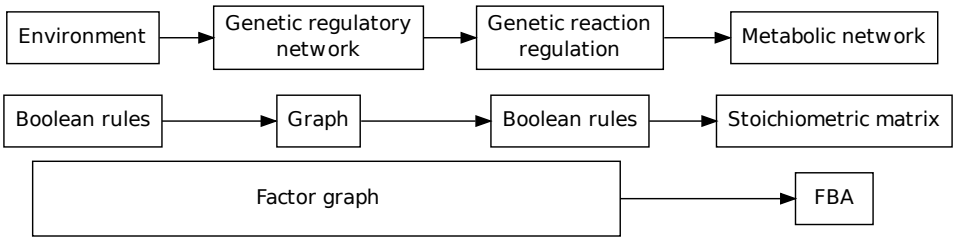


Figure 3.1: An overview of the implementation of our method. Information flows through the model along the direction of the arrows. Each column shows three corresponding levels; domain, representation and implementation.

The following sections will describe how the problem is formulated as a factor graph and how the result of this formulation is integrated with the constraint-based linear optimization problem.

3.1 Factor graph formulation

Before taking the step of formulating the problem as a factor graph, we define an ordinary graph of nodes and pairwise edges, that describes the regulatory system of an organism. This graph includes knowledge of how genes respond to changes in the environment, how genes influence each other and how genes control metabolic reactions.

It is known that there is a response in genetic activity to which substances that are available in the cell's environment. Herrgård et al. have described this response in a structured fashion in [24]. The description consists of a set of boolean rules for how genes are active or repressed based on the availability or absence of substances in the environment, as described in section 2.2. We capture these boolean rules in the graph by letting each substance and each gene be a node. Additionally, nodes representing the logic operators ("and" and "or") are added to the network. This allows for the construction of parts of the graph which describe boolean rules, see figure 3.2.

At this point some of the transcription factors are already added to our graph. Lists of pairs of interacting genes exist for the model organisms. The addition of such interaction lists to the graph is straight forward. Each gene in the list is added to the graph as a node, and nodes that represent genes which interact with each other are connected with an edge. This is exemplified in figure 3.3.

Similarly to how the environment's impact is captured, the control of metabolic reactions by target genes is also often described by boolean rules and such descriptions exist for many organisms [24] [9] [8]. Each metabolic reaction which is genetically controlled is added as a node to the graph. As previously, nodes representing logic operators are also added and connected to target genes. See figure 3.4. This concludes the creation of

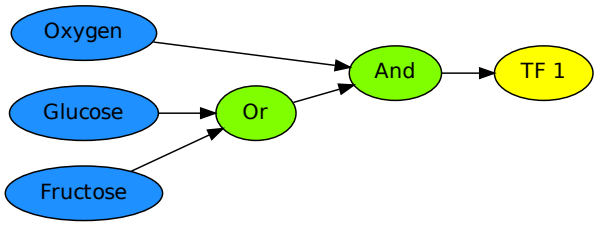


Figure 3.2: Example of how a boolean rule is represented as a graph. The boolean rule represented here is `TF_1 := (Glucose or Fructose) and Oxygen`. That is, the transcription factor gene is active if the right hand side of the expression is true.

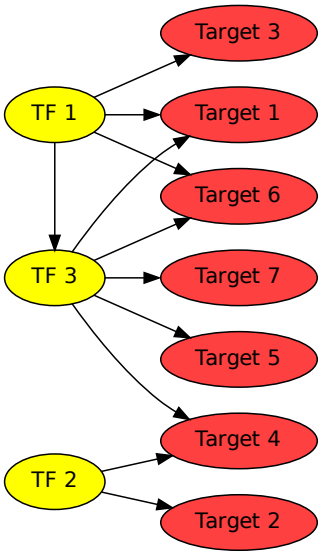


Figure 3.3: A synthetic genetic network consisting of three transcription factor genes and seven target genes.

the graph.

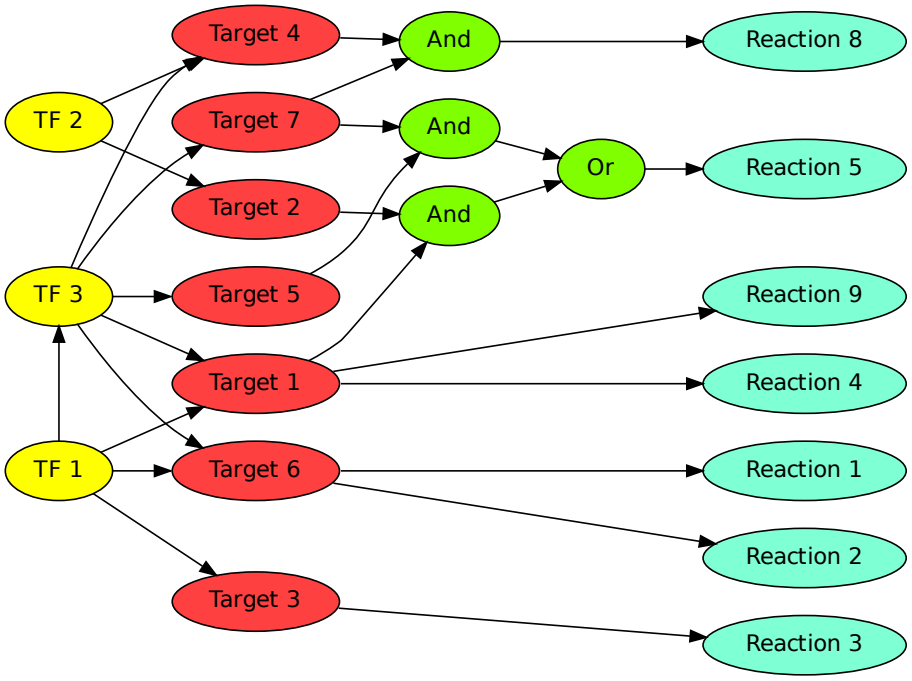


Figure 3.4: A synthetic genetic and regulatory network consisting of three transcription factor genes, seven target genes and seven metabolic reactions. The metabolic reactions are controlled with boolean rules.

We will now describe the conversion of this graph to a factor graph. Factor graphs in general are described in section 2.3.1. As mentioned there, factor graphs are bipartite graphs, and the original nodes can only be connected via so called factor nodes. We propose two ways of creating the factor graph, we will call them the Bayesian network approximation and the Bethe approximation.

In the Bayesian network approximation we remove all connections between transcription factor genes. By doing so we get a directed graph without cycles whose nodes represent random variables, a Bayesian network. This network can then be converted to a factor graph by inserting one factor node for each target gene, and connecting the factor node to its target and to all transcription factors that were connected to the target in the original graph. See the left part of figure 3.5. Associated to each factor node is a table, which holds information about the interaction strengths between all genes that are connected via the factor node, and if the interaction is of a repressing or activating character. The information in the factor matrices is learnt from gene expression profiles that are given as input when constructing the factor graph. If, for example, a factor node

connects three genes, then its factor table will contain the number of times that each combination of genes being on or off is seen in the gene expression profiles. The factor tables can be thought of as containing counts or probabilities, they are normalized so the results are equivalent. Table 3.1 shows an example of a factor table for the Bayesian network approximation. This approach loses information from the original graph about connections between transcription factors. But if the gene expression profiles show a strong connection between transcription factors that are connected to the same factor node, these interactions are captured by the factor table. A problem with the Bayesian network approximation is that factor tables can grow very large.

Table 3.1: Example of a factor table in the Bayesian network approximation. Each row shows one configuration of the three genes, 1 meaning that the gene is on and 0 meaning that the gene is off.

TF_1	TF_2	G_1	Count
0	0	0	2
1	0	0	5
0	1	0	10
1	1	0	0
0	0	1	3
1	0	1	11
0	1	1	5
1	1	1	5

In the Bethe approximation the original graph is instead converted to a factor graph by simply inserting a factor node between each pair of connected nodes. This avoids large factor tables, in this approach all factor tables consists of four rows, the four possible combinations of the two genes that are connected to it. This approximation is described by Yedida in [30].

The nodes in the original graph that represent boolean logic operators demand a different treatment than other nodes when converting the graph to a factor graph. Instead of using gene expression data to attain the strength and character of a connection, as done for connections between genes, the information about connections is already contained in boolean rules. By replacing each boolean rule, and all logic operator nodes associated with it, with a factor node the rule is transitioned into the factor graph framework. The boolean logic is contained in the factor table as exemplified by table 3.2. Here the same problem as in the Bayesian network approximation arises; tables grow exponentially with the size of their boolean rules. This problem is easily solved by inserting dummy nodes as described in section 2.3.4.

It is possible to use this factor graph formulation even when only parts of the regulatory system is to be modeled. If, for example, the gene interactions are to be examined in isolation, only genes are added to the initial graph which is then converted to a fac-

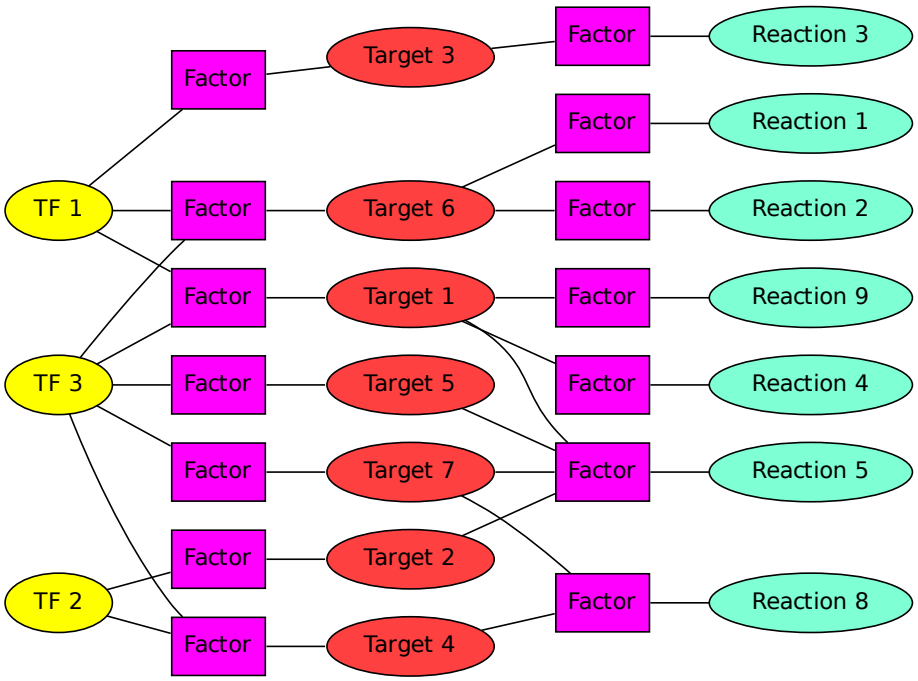


Figure 3.5: A Bayesian network factor graph corresponding to the network in figure 3.4. Connections between transcription factor genes, up- or downregulation and boolean rules are all captured within the factor nodes.

tor graph. Indeed, because of the lack of data for genetic and phenotypic responses to environmental changes, the environmental part of the graph will be excluded in our experiments. This lack of data is discussed further in section 4.3.3. Thus, the abilities of our method's modeling of the environment have not been evaluated.

One important difference from other data-driven methods for modeling gene interactions, such as [31], is that, unlike Friedman et al., we are not trying to learn the structure of the network, since high-confidence regulatory networks already exist for *S. cerevisiae* [12], *E. coli* [32] and *M. tuberculosis* [33]. We are therefore only interested in learning the nature of the connections in this given network structure, such as whether a transcription factor is repressing its target or activating it, and how strong its effect is.

When the factor graph, including all factor tables, has been constructed it can be used to compute probabilities for the nodes in the network to be active. The nodes whose state is known, in our cases a subset of the environment nodes or a subset of the transcription factor nodes, are given to the factor graph. Running the sum-product algorithm then calculates probabilities for all nodes in the network. States of the known

Table 3.2: Example of how a boolean rule is represented as a factor table. The example shows the table for the rule `tf2 := glucose OR fructose`. For environmental substances 1 means present and 0 means absent.

glucose	fructose	TF_2	Count
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

nodes can be imposed on the network either by setting the probability to exactly zero or one, we call this "clamping", or by adding a new factor node and connecting it to the known node, thus imposing a prior probability of that node to be on or off.

In the case of using the input to clamp nodes in the network, the factor graph can be forced into a state that has a zero probability, causing the sum-product algorithm to fail. States have zero probability if they correspond to a combination in a factor table that never has been seen in the training data. To alleviate this, Laplace priors need to be added to the factor tables. That is, a small value is added to all rows in all factor tables to avoid that any combination has a probability that is exactly zero. We use a Laplace prior with the value $n/1000$, where n is the number of gene expression profiles that we learn from. The impact of changing this value was negligible, but the Laplace prior prevented the sum-product algorithm from failing for some inputs.

After imposing the input, by clamping or imposing prior probabilities, and running the sum-product algorithm, there is a value for each node representing that node's probability of being active. Thus, given a configuration of availability of substances in the environment or of transcription factor activity, a set of probabilities for target genes to be active or for metabolic reactions to be genetically activated is returned.

3.2 Discretization of gene expression data

Gene expression profiles contain the expression levels of genes as real numbers. In order to learn the factor tables discussed above, we need discrete gene expression, genes are either more expressed than usually or less expressed than usually. The discretization of gene expression profiles is a well studied problem, and a range of methods exists. Following Nookaew's recommendation [34], we chose the simplest possible method for discretization; thresholding. Since we are only interested in finding the strength of gene

interactions, and if the interaction is activating or repressing, it makes sense to threshold the expression levels for each gene with regards to the mean expression level of that gene in all profiles in the dataset. Thus, for each set of gene expression profiles, the mean value was computed for each gene. Then the gene was classified as active in the profiles where the gene's expression level was above the mean, and otherwise it was classified as repressed.

3.3 Integration with FBA

To complete the model, the factor graph is integrated with the FBA framework described in section 2.4. This is done in the same way as in the PROM method and is described in section 2.5.3. The probabilities of metabolic reactions to be active are computed with the factor graph and then integrated with FBA by limiting the maximum reaction rates based on these probabilities.

3.4 Implementation dependencies

Our implementation uses libDAI [35], a software library for discrete approximate inference in graphical models, for constructing factor graphs and running the sum-product algorithm.

The openCOBRA project provides a software toolbox for constraint-based reconstruction and analysis in systems biology [36]. Our implementation uses the openCOBRA toolbox for performing flux balance analysis. OpenCOBRA supports the systems biology markup language (SBML), which is a machine-readable format for representing biological models such as metabolic pathways [37]. We use SBML to import metabolic models for the model organisms. The GNU linear programming kit (GLPK)[38] was used to solve the resulting optimization problems, unless stated otherwise in the experiment description. In one experiment, MOSEK [39] was used in addition to GLPK.

4

Experiments

We performed experiments using both synthetic data and gene expression profiles from publically available databases. In this chapter we describe how the synthetic data was generated, the datasets used, and the results of the experiments.

4.1 Synthetic experiment

In this experiment we construct a synthetic organism and use it to compare methods for predicting probabilities for genes to be on for different gene knockouts and for predicting metabolic reaction rates.

4.1.1 Model details

The synthetic organism consists of a randomly generated regulatory network and the toy model metabolic network from [6]. The regulatory network consists of five transcription factor genes, 95 target genes, three randomly chosen connections between transcription factor genes, 277 randomly chosen connections from transcription factor genes to target genes and seven randomly generated boolean rules for how target genes control the seven genetically controlled reactions in Covert's toy model. The regulatory network was generated to have properties similar to a real organism. Specifically, it was modeled after Harbison's description of the regulatory network of *S. cerevisiae* which is described in section 4.2.3. The synthetic regulatory network has the same ratio between regulated and unregulated genes as Harbison's network, its nodes also have the same mean degree and a similar degree distribution in the sense that the vast majority of genes has a degree lower than the mean. To achieve a degree distribution similar to that of Harbison's network, edges was added using preferential attachment as described in [40]. Each gene to gene connection is either activating or repressing. Boolean rules to control the seven genetically controlled metabolic reactions of Covert's toy model was chosen randomly

from three classes with equal probability. A reaction is either controlled directly by one gene, controlled by two genes via a basic logical operation or controlled by four genes via a binary tree of three basic logical operations, figure 3.4 exemplifies this.

4.1.2 Data generation using Gibbs sampling

The known (randomly generated) genetic network $G = (V,E)$ is used to create a factor graph based on the conditional probability distribution

$$P(X = x|W = w) = \frac{1}{Z(w)} \exp\left(-\frac{1}{2}\sum_{i,j\in V} w_{ij}x_i x_j\right)$$

for the interaction strengths $W = \{W_{ij} \in \{-1,1\} : (i,j) \in E\}$ and the gene expression levels $X = [X_1,\ldots,X_n] \in \{-1,1\}^n$. This describes a discrete multinomial distribution and, more specifically, an Ising model. Ising models were first developed to study ferromagnetism [41] but have also been found to be useful in the study of gene expressions. $Z(w)$ is a normalizing constant known as the *partition function*, and is needed to normalize the function and make it a proper distribution (that is, $\sum_x P(x|w) = 1$, where the sum is over all valid values of x) [42]. The factor graph that is created based on this distribution is used to generate 100 synthetic gene expression profiles, using Gibbs sampling, for our algorithms to learn from. It is also used to generate 1000 synthetic gene expression profiles for each transcription factor knockout. By counting the number of times each gene is turned on in the 1000 clamped expression profiles we get a good approximation $\mathbf{p}_{\text{true}}$ of the true target gene probabilities.

Gibbs sampling, a Markov Chain Monte Carlo algorithm which is both simple and powerful, is used to draw representative samples from the multinomial distribution.

To use Gibbs sampling we must have a way to compute the marginal probabilities of the distribution. That is, in our case, we need

$$P(X_i|X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n),$$

where n is the number of genes, $i \leq n$. These marginal distributions are exactly what is computed by the sum-product algorithm.

After initialization, in each step one of the parameters X_i, for some $i \leq n$, is replaced by sampling a new value from the marginal distribution. This is done for a fixed number of steps; the parameters can be replaced either in order or randomly. It can be shown that if all marginal distributions are always non-zero, this procedure will give samples from the distribution $P(X|W)$. In the case of the Ising model, where all marginal distributions are exponential, this is clearly true.

4.1.3 Results

All numerical values are averages from 100 different random organisms.

Table 4.1: Comparison of methods for predicting target gene probabilities. Vector $\mathbf{p}_{\text{true}}$ contains the close estimations of probabilities that are calculated using Gibbs sampling and the underlying genetic network, and $\hat{\mathbf{p}}$ contains the estimations by each method.

method	$\max(\|\|\mathbf{p}_{\text{true}} - \hat{\mathbf{p}}\|\|_1)$	$\text{mean}(\|\|\mathbf{p}_{\text{true}} - \hat{\mathbf{p}}\|\|_1)$	mean accuracy
PROM	22.3	16.1	0.71
Bethe	14.6	9.4	0.82
Bayesian network	9.0	5.1	0.86

Table 4.2: Comparison of methods for predicting reaction rates. Vector $\mathbf{v}_{\text{true}}$ contains the close estimations of reaction rates that are calculated using Gibbs sampling, the underlying genetic network and letting metabolic reactions be either completely on or completely off, and $\hat{\mathbf{v}}$ contains the estimations by each method.

method	method	$\max(\|\|\mathbf{v}_{\text{true}} - \hat{\mathbf{v}}\|\|_1)$	$\text{mean}(\|\|\mathbf{v}_{\text{true}} - \hat{\mathbf{v}}\|\|_1)$
PROM	PROM	69.2	45.7
Bethe	PROM	40.0	25.3
Bayesian network	PROM	39.6	22.0
Bethe	bounds	30.9	21.3
Bayesian network	bounds	26.5	17.4

Table 4.1 compares the three algorithms for prediction of target gene probabilities. We let each algorithm make a prediction for 30 different transcription factor configurations. PROM does only make predictions for a few of the target genes whereas our methods make predictions for all target genes. Using only the target genes for which PROM makes prediction, we find the l_1-distance between each method's prediction and $\mathbf{p}_{\text{true}}$. The table shows both the mean and the max error for the five knockouts. It also shows the mean accuracy defined as

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{4.1}$$

where $\mathbf{p}_{\text{true}}$ and the estimation $\hat{\mathbf{p}}$ are rounded to on/off and the two-letter abbreviations stand for true positives, true negatives, false positives and false negatives, respectively. All three measures show a clear advantage for our methods over PROM.

Table 4.2 compares the predicted reaction rates for different combinations of methods for predicting target gene probabilities and reaction rates. Again, a clear advantage for our methods over PROM. Here the values $\mathbf{v}_{\text{true}}$, similarly to $\mathbf{p}_{\text{true}}$, are used as correct values to compare estimations with. The values are calculated by Gibbs sampling 1000 times as before, and for each sample FBA is run with reactions turned absolutely on or off in accordance with the gene activity given by the sample and the boolean reaction rules. Then $\mathbf{v}_{\text{true}}$ is the average reaction rates over the 1000 samples. In the second column in the table, "bounds" means that instead of using the slack variable approach from PROM,

the bounds are simply set to $pV_{\max}$ for each reaction, where p is the probability of the reaction being active and $V_{\max}$ is the maximum possible reaction rate calculated with FVA.

The results of this experiments show an advantage to both factor graph methods compared to PROM. The Bayesian network factor graph has the ability to capture more dependencies in the underlying network, and as expected it has a slight advantage over the Bethe approximation. As discussed in section 3.1 the Bayesian network approach becomes intractable for networks with nodes of high degree, a problem that the Bethe approximation does not have. We conclude that, while the Bayesian network approximation performs best in this experiment, the Bethe approximation also shows strong improvements over PROM, and that it therefore is a good alternative when the Bayesian network approximation is intractable. For example, Harbison's regulatory network for *S. cerevisiae* has several nodes with degree above 200 which would result in factor node matrices with more than 2^{200} elements. This experiment does also show the ability to achieve better reaction rate approximations by setting hard bounds on reaction rates. Unfortunately, it is the relative simplicity of the metabolic network used here (Covert's toy model) that makes these results possible. Setting hard bounds on metabolic reactions for more complex organisms often results in wrongfully predicting lethal outcome. We conclude that there is room for improvement over PROM's method for limiting reaction rates, and that the problem should be studied further.

4.2 Descriptions of datasets

Although the synthetic organism discussed in section 4.1 is a schematic model of real organisms, a lot changes when trying to predict results for real organisms. Most prominently, real systems are bigger than our synthetic system and existing knowledge about real systems contains errors and noise. Our methods were tested on regulatory and metabolic systems of three real species. *Escherichia coli*, a prokaryotic bacteria, and *Saccharomyces cerevisiae* (baker's yeast), an eukaryotic fungi, are two extremely well studied model organisms [8]. Within the field of computational biology *E. coli* is used as a model for prokaryotic organisms, organisms without a cell nucleus, whereas *S. cerevisiae* is used as a model for the more complex eukaryotic organisms, organisms with cell nucleus. Chandrasekaran et al. evaluate their method on *Mycobacterium tuberculosis*, a critically important human pathogen. For comparison our method is also evaluated on the same dataset. Our model of a regulatory system requires two types of data. First, a list of interacting genes, which is used to create a graph with genes as nodes and edges between interacting genes. Second, a set of gene expression profiles consisting of measurements of gene activity for each gene and for different environments or strains is used to find the strength of gene interactions. To be able to make predictions of metabolic reaction rates and growth rate, rules for how reactions are controlled by genes and a stoichiometric matrix describing the metabolic reactions are also required.

4.2.1 *E. coli*

The gene expression profiles for *E. coli* were downloaded from the M3D database [2]. Specifically, the file containing values from 4297 probes, averaged over all repetitions, from the dataset `E_coli_v4_Build_6` was used.

4.2.2 *M. tuberculosis*

The same data was used as in the PROM paper, that is, 437 microarrays of *M. tuberculosis* strain H37Rv measuring the effects of 75 drugs [1].

4.2.3 *S. cerevisiae*

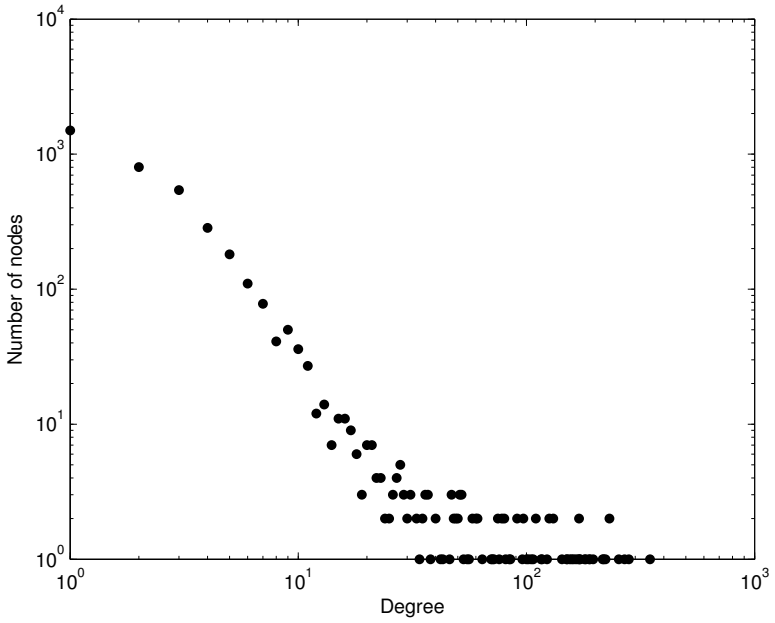


Figure 4.1: Degree distribution for Harbison's regulatory network.

In *S. cerevisiae* a dataset by Harbison et al. [12] was used for high confidence information about interacting gene pairs. It contains 10884 pairs, connecting 3868 unique genes. 60 of the genes are only regulating and 3692 of the genes are only regulated. The nodes in the resulting graph has a mean degree of 5.6 and its degree distribution is presented in figure 4.1. It can be seen that a vast majority of nodes has a lower degree than the mean, but some nodes have a degree that is far above the mean.

A small set of gene expression profiles was provided by Nookaew [34]. The set contains gene expressions for 5667 genes and four different environments; carbon limited aerobic, carbon limited anaerobic, nitrogen limited aerobic and nitrogen limited anaerobic. For each environment there are three independent measurements at 15 °C and three

independent measurements at 30 °C, resulting in a total of 24 gene expression profiles. The dataset was discretized by classifying each value as activated if it was higher than the mean for that gene over all profiles and not activated otherwise, as per Nookaew's recommendation.

A big set of gene expression profiles was downloaded from Many Microbe Microarrays Database (M3D) [2]. The set that was used contains gene expressions for 6929 genes with systematic names. Each gene has corresponding measurements of gene expression level for 159 experiments for various strains and environments. This dataset was discretized in the same way as Nookaew's.

Herrgård provided us with a dataset that was compiled for [24]. It contains gene expression levels for 750 genes for 8 different transcription factor knockout strains. The gene expressions are combined with a high confidence discretization into repressed, unchanged, activated or unknown, as well as a manual prediction based on literature studies as discussed in 2.5.2.

Publicly available datasets for *S. cerevisiae* use standard gene names (for example ROX1), systematic gene names (for example YPR065W) or a mixture thereof. We converted all datasets to contain systematic names only, using the Saccharomyces Genome Database [3].

4.3 Results

In this section we present the results of our experiments on non-synthetic datasets. An experiment using the graphical lasso was done using the *E. coli* data, while our model was used to predict gene knockout effects using *M. tuberculosis* data, and to predict target gene expressions on *S. cerevisiae* data given the transcription factor states.

4.3.1 *E. coli*

Construction of regulatory networks is time consuming, and requires extensive literature studies. It would therefore be very useful if the regulatory network could instead be inferred from the available gene expression profile data.

We therefore ran the graphical lasso algorithm, described in 2.3.5, on 4297 gene expression profiles from the M3D database (as described in 4.2.1), and compared the resulting network with the network made available by the RegulonDB team [32]. The parameter ρ was set to 0.05. Note that the graphical lasso used the real valued raw data (as required by the graphical lasso algorithm); no binarization was done.

The results were discouraging, and we therefore abandoned this approach. As an example, for the gene gadC and considering only connections in the network inferred by graphical lasso with connections greater in magnitude than 0.05, 10 connections were found. Only 5 of these were the same as in the RegulonDB network. Of the erronous ones, 1 was wrongly found to be inhibitory, and the other 4 connections were not in the RegulonDB network at all. Results of other genes had similarly low accuracy.

Instead, a regulatory network must be supplied to be able to use our model. This is also the case for PROM and rFBA.

4.3.2 *M. tuberculosis*

As in the PROM paper [8], knockouts were simulated for all 30 transcription factors of the *M. tuberculosis* regulatory network compiled by Balazsi et al. in [33]. This was done to determine the essentiality of the different transcription factors. A transcription factor was considered as essential if it was needed for optimal growth, which is defined as the growth rate of the wild type.

The gene expression were discretized in the same way as by PROM, described at the end of 2.5.3.

The results are summarized in table 4.3, which shows the predicted growth rates for different knockouts. If the value in a column is not equal to (or very close to) the column's highest value, the knocked out gene is predicted as essential.

Knockouts which PROM predicts as being essential, but for which [8] did not find any information about in the literature, were considered to be candidates to be essential genes. A recent analysis of PROM [13] found that all transcription factors hypothesized to be candidate essential genes in [8] are, in fact, nonessential according to literature.

We were unable to reproduce the results found in the PROM paper. It was suggested by Sriram Chandrasekaran [43] that this might be due to the particular version of the optimization problem solver, GLPK version 4.47. However, using GLPK version 4.43 yielded the same results.

Therefore, we also solved the optimization problem using a commercial optimization software, MOSEK [39]. The predicted growth rates for PROM when using this solver were different from both the results in the PROM paper and the results when using GLPK, but followed the same pattern with a few exceptions (dnaA, Rv3575c, argR).

Gene dnaA is incorrectly predicted as being nonessential when using MOSEK and correctly predicted as essential using GLPK, Rv3575c is correctly predicted as nonessential when using MOSEK and incorrectly when using GLPK, and argR is correctly predicted as nonessential when using MOSEK but incorrectly predicted when using GLPK. In other words, when using MOSEK, PROM gets one more prediction correct.

The results indicate that the optimization problems these algorithms need to solve are not always numerically stable, since the results are different when using different solvers. The zero values indicate that no solution was found. This happens only once for our model with GLPK, and never when using our model and MOSEK.

The predictions made by our model (using the Bethe approximation) are generally worse than those made by PROM. However, [13] showed that PROM gets worse results using more recent regulatory networks, and this is a very limited number of data points.

4.3.3 *S. cerevisiae*

Although vast amounts of experimentally measured gene expression profiles are available for *S. cerevisiae*, the lack of data appended with structural metadata proved to be a

Table 4.3: Growth rate predictions for *M. tuberculosis* knockouts. Subscript g indicates that the GLPK solver was used, m that the MOSEK solver was used. Columns "paper" and "claim" have the values found in the supplemental material of [8], while the values in "literature" are taken from [13]. In the last two columns, "E" means essential, "C" means candidate essential, and "N" means nonessential.

TF	Paper	PROM$_g$	PROM$_m$	Bethe$_g$	Bethe$_m$	Claim	Literature
dnaA	0.03	0.031	0.055	0.026	0.058	E	E
Rv0485	0.042	0.042	0.047	0.027	0.030	E	E
crp	0.03	0.000	0.000	0.016	0.024	E	E
sigD	0.05	0.000	0.000	0.050	0.057	E	E
kdpE	0.052	0.052	0.057	0.052	0.057	E	E
ideR	0.038	0.000	0.000	0.016	0.011	E	E
Rv1395	0.028	0.000	0.000	0.027	0.030	C	N
argR	0.047	0.047	0.057	0.026	0.031	C	N
sigC	0.024	0.000	0.000	0.026	0.029	C	N
sigH	0.05	0.000	0.000	0.027	0.032	C	N
lrpA	0.032	0.032	0.036	0.016	0.018	C	N
Rv3575c	0.026	0.027	0.056	0.026	0.010	C	N
oxyS	0.052	0.052	0.057	0.050	0.058	N	N
nadR	0.052	0.052	0.057	0.052	0.057	N	N
hspR	0.052	0.052	0.057	0.052	0.057	N	N
regX3	0.052	0.052	0.057	0.052	0.057	N	N
Rv0586	0.052	0.052	0.057	0.052	0.057	N	N
narL	0.052	0.052	0.057	0.052	0.057	N	N
sigE	0.052	0.052	0.057	0.052	0.057	N	N
furA	0.052	0.052	0.057	0.052	0.057	N	N
Rv1931c	0.052	0.052	0.057	0.000	0.057	N	N
furB	0.052	0.052	0.057	0.052	0.057	N	N
lexA	0.052	0.052	0.057	0.033	0.036	N	N
pknK	0.052	0.000	0.000	0.041	0.045	N	N
dosR	0.052	0.052	0.057	0.052	0.057	N	N
birA	0.052	0.052	0.057	0.052	0.057	N	N
sigF	0.052	0.052	0.058	0.052	0.057	N	N
kstR	0.052	0.052	0.057	0.037	0.040	N	N
cyp143	0.052	0.052	0.057	0.052	0.057	N	N
embR	0.052	0.052	0.057	0.052	0.057	N	N/E

problem. To be able to evaluate our model fully, we would need several gene expression profiles annotated with both machine readable data about the environment in which each profile was measured and the resulting phenotype (growth rate or other reaction rates) for that profile or group of profiles. We were unable to find such datasets. In this section we therefore aim to evaluate our model's ability to make predictions for the regulatory network, that is the interaction between transcription factor genes and target genes. We show that our model makes better predictions than PROM, but that the improvement is small. Since PROM lacks the ability to make phenotype predictions for *S. cerevisiae* [13] and our model only shows a small improvement in predictions for the regulatory network, we strongly believe that an evaluation of our entire model would show that further improvements are needed in modeling, knowledge of the regulatory network or both to be able to make phenotype predictions for eukaryotic organisms.

Two experiments were conducted. In the first we evaluate our model's ability to predict the activity of target genes given the activity of transcription factor genes, and compare it with PROM.

The input to both algorithms is one set of gene expression profiles to learn from and Harbison's regulatory network. The datasets are described in section 4.2. Ordinary leave-one-out cross-validation [29] is used. For each gene expression profile in the dataset, one prediction is made and evaluated, where the algorithms learn from all other gene expression profiles in the dataset and predicts the activity of the target genes in the left-out profile, given the activity of the transcription factors in that profile. To be able to evaluate the influence of the size of the datasets on the algorithms, this experiment was run both for Nookaew's 24 gene expression profiles and for the 159 profiles in the M3D dataset. All gene expressions are discretized as described in section 3.2, both for learning and when evaluating the success of predictions.

The PROM algorithm does not take arbitrary transcription factor configurations as input, it only makes predictions for single transcription factor knockouts. Chandrasekaran et al. propose a method for allowing double knockouts as well, but the algorithm has not been extended to support arbitrarily many knockouts. This experiment aims to examine the impact of not making the assumption of independence that the PROM algorithm does. Therefore the PROM algorithm was adapted to keep its inherent assumption of independence, but add support for arbitrary transcription factor inputs. Hence, this experiment can not be seen as a critique of the original PROM algorithm, it is solely an attempt to show the impact of not making this assumption of independence. In the case of single gene knockouts, PROM estimates the probability of a target gene to be active by calculating the frequency in the learning dataset of times when the transcription factor is off but the target is still on. See equation 2.20. Using this equation we get a set of probabilities for each target, one for each transcription factor that is connected to that target. If the average probability of a target is more than 0.5 the gene is predicted to be on and otherwise it is predicted to be off. Thus, we have constructed an adaptation of PROM which gives predictions given arbitrary transcription factor configuration and that has an assumption of independence as the original PROM method.

As explained in chapter 3, our model can use a Bethe approximation factor graph or a Bayesian network factor graph. The Bayesian network factor graph gives better estimates but the large degree of some nodes in Harbison's network makes this method computationally intractable. Therefore the Bethe approximation is used. The target gene probabilities that are produced by our method for each transcription factor configuration are thresholded around 0.5 to give predictions that are either on or off, active or repressed.

Table 4.4: Comparison of an adaptation of the PROM algorithm with the Bethe factor graph for predicting target gene activation given transcription factor gene activation. Data from 24 leave one out trials for each dataset and Harbison's genetic network. One of the trials resulted in equal accuracy, hence victories for the Nookaew dataset sum up to 23.

dataset	method	mean accuracy	min accuracy	victories
Nookaew	PROM	0.62	0.53	4
Nookaew	Bethe	0.64	0.57	19
M3D	PROM	0.62	0.53	3
M3D	Bethe	0.68	0.56	21

The results of this experiment are shown in table 4.4. It compares the methods for both datasets. For each dataset, 24 leave-one-out trial are made. At each trial the accuracy, defined as

$$\text{accuracy} = \frac{|\text{targets predicted correctly}|}{|\text{targets}|},$$

is recorded, this formulation is equivalent to the one in equation 4.1. After all trials for that dataset, the mean and minimum accuracy is computed. These values are presented in the table along with the number of "victories" for each method, that is the number of trials in which that method had a higher accuracy than the other. It can be seen in the table that the ability of the Bethe approximation to take dependencies between transcription factors and between targets into consideration improves the accuracy when predicting target gene activity. The improvement of our method for the bigger dataset compared to the smaller is too small to use for making any assumption about the method.

Our second experiment on *S. cerevisiae* uses Herrgård's gene expression profiles. Again, see section 4.2. This dataset contains eight gene expression profiles, each profile was measured in a yeast strain where one of eight different transcription factor genes was knocked out. These profiles contain expression levels only for target genes, and can therefore not be used by PROM or our method to learn from. Therefore, the previously mentioned M3D dataset was used to learn regulatory interaction from. Further, Herrgård's gene expression profiles are already discretized with a more advanced method than the one that we are using. Instead of a binary discretization, the gene expression levels are discretized into four classes; active, repressed, unaffected and unknown. Thus, only a subset of the gene knockout combinations can be correctly predicted by PROM

and our method. In contrast to the previously described experiment, in this experiment target activity is predicted given a single transcription factor knockout and thus the original PROM algorithm can be used.

The aim of this experiment is to make a fair comparison between PROM and our method. Herrgård's dataset contains manual predictions for a subset of the target knockout combinations from [24]. Therefore, both methods can also be compared to predictions that have been made manually using vast knowledge and literature studies of the model organism.

The experiment was conducted by letting both PROM and our method use the Harbison regulatory network and learn interaction strengths from the M3D gene expression profiles as before. Then for each of the eight knockouts considered by Herrgård, both methods made predictions for how target genes were affected by that knockout. This results in four 750×8 matrices; one with the experimentally measured target activities (here seen as the correct target activities) with each element discretized into active, repressed, unaffected or unknown, one containing PROM's predictions with each element containing either a probability or missing value in the cases no prediction was made, one containing our methods predictions, also with each element containing either a probability or missing value, and, last, one containing Herrgård's manual predictions with each element in one of the four classes used in the experimental matrix. The matrix representing PROM's predictions misses a lot of values since PROM only makes predictions for targets that are connected directly to the transcription factor. Our method's matrix, on the other hand, misses values only in a few places where that specific target is not included in Harbison's network.

Two measures were taken to cope with the missing values in these matrices, when comparing the predictions. First, the 750×8 cases were divided into three subsets; one subset where the experimental result is either active or repressed and the manual prediction is either active or repressed, one subset where the experimental result is either active or repressed and where there is no manual prediction but PROM makes a prediction, and one subset where the experimental result is either active or repressed and our method is the only method that makes a prediction. These subsets are called "manual", "close" and "distant" respectively, and constitutes one column each in table 4.5. The names "close" and "distant" are chosen since the second subset consists of targets that are connected directly to a knocked out transcription factor, whereas the third subset consists of targets that are only indirectly connected to a knocked out transcription factor. The second measure that is taken to cope with missing values, is that we use *precision* and *recall* to evaluate the methods' success. In this way we can evaluate both how many predictions that are made, and how good they are. In information retrieval, precision is the proportion of retrieved predictions that are relevant, and recall is the proportion of relevant material that is correctly retrieved [44]. In our setting we have that

$$\text{precision} = \frac{|\text{cases predicted correctly}|}{|\text{cases predicted}|}$$

and

$$\text{recall} = \frac{|\text{cases predicted correctly}|}{|\text{all cases}|}.$$

Since the class of targets experimentally classified as active is not the same size as the class of targets classified as repressed, it is not consistent to let the threshold be 0.5 when classifying predictions as either active or repressed. We therefore varied the threshold between 0 and 1 and used the one that gave the highest precision (and hence also the highest recall).

Table 4.5: Comparison of Herrgård's manual predictions and predictions from two data-driven methods; PROM and our method using the Bethe approximation. The table shows precision and recall on three subsets of combinations of transcription factor knockouts and target genes. For each knockout, the predictions of each method is compared with the experimentally measured activity.

	manual	close	distant
no. of cases	26	11	271
	precision recall	precision recall	precision recall
Herrgård	0.96 0.96	-	-
PROM	0.80 0.15	0.82 0.82	-
Bethe	0.86 0.69	0.82 0.82	0.65 0.49

The results of the second experiment are shown in table 4.5. The line named Herrgård contains values for the predictions made manually by Herrgård. The first column compares the three sets of predictions for the subset of cases where manual predictions are made. As expected, the manual predictions are a lot better than the predictions made with data-driven methods. An advantage for our method over PROM can be seen in both precision and recall. The second column compares the cases where target genes are connected directly to a knocked out transcription factor, but where there is no manual prediction. This subset is too small to show a difference between PROM and our method. The third column shows the precision and recall of Bethe in the cases where neither Herrgård nor PROM is able to make a prediction. These are targets that have a distance of two or more to a knockout, so the decline is expected. The precision and recall are both clearly lower than for the other subsets. This is expected, since these genes lies at a higher distance in the graph from the points where information is supplied, than the genes in the other subsets. Still, it should be noted that a precision of 0.65 is significantly better than random guesses, and that Bethe thus has an ability to make predictions in cases where the other methods have not.

5

Conclusion and future work

We have shown that our model performs very well on data generated from an Ising model, which is believed to be a reasonable model for gene expression profiles. Compared to PROM, our model can deal with more complex interactions in a sound way, by considering the joint distribution of the dependencies, rather than considering each interaction independently. It is possible that this is a disadvantage when insufficient amounts of data, or data with an insufficient amount of variation (over different strains, knockouts, and environment), are available for organisms with complex regulatory networks, which could explain our results on *M. tuberculosis* in section 4.3.2.

We would have liked to run more experiments on *E. coli*, but we had problems finding the proper parameter settings to use for FBA, and were unable to reproduce the results in [8] and [45]. The results in [13] show that PROM gets much better results for *E. coli* than for *S. cerevisiae*. Presumably, this would also be true for our model.

We have shown that our method performs somewhat better than PROM, when predicting target gene activity in *S. cerevisiae* using gene expression profiles. Lack of datasets with machine readable metadata made it impossible to evaluate the performance of our entire model, including environment and metabolism. Given better datasets it would be a good evaluation to compare the results of our model on *S. cerevisiae* with the predictions made manually in [24]. When predicting target gene activity given transcription factor gene activity, we have shown that predictions improve when the independence assumptions of PROM are not made. We have also shown that our method is able to make significant predictions in cases where other methods are not.

The quality of the gene expression profile data is of primary importance to our algorithm. Discretization by using a single value (either the 0.33 quantile, as PROM does, or the mean) for deciding if a gene is on or off is a simplistic way to do it. Several more sophisticated methods exist, such as MADE [46] and iMAT [47]. It would be interesting to use our model with binary gene expression profiles obtained from one of these

algorithms. Our model could also be extended to use trinary gene expression profiles, where genes are either repressed, unchanged or over-expressed. The belief propagation framework is well suited to such an extension.

A great advantage of our method is its ability to include the entire regulatory system, both environment, genes and reaction regulation, in one coherent probabilistic model. It takes a principled approach and uses the well-studied framework of factor graphs to perform belief propagation on the regulatory network. Regulation of metabolism is a well-studied problem in systems biology, but to our knowledge the factor graph concept has not been used to study it before.

Instead of trying to solve the FBA problem in section 1.2, both PROM and our model use slack variables to try to ensure that the problem is feasible. The slack variables are difficult to motivate biologically, and even then, the solvers we used in this thesis did not always find a solution when using PROM, as shown in section 4.3.2.

Therefore, rather than using slack variables and modified bounds, future work could model gene activations as random variables, with appropriately chosen mean (average probability of activation, computed using our model) and variance.

Since the bounds for the reaction rates are based on these random variables, they too are uncertain, they should be considered as normal distributed random variables with a certain mean and some variance.

This leads to *chance constraints* where reaction rates are bound to lie within a constraint set with high probability when the constraint parameters are random. This is an example of what is known as a stochastic robust optimization problem [48]. It can be shown to be a convex problem, which can be solved efficiently.

Bibliography

[1] H. Boshoff, T. Myers, B. Copp, M. McNeil, M. Wilson, C. Barry III, The transcriptional responses of mycobacterium tuberculosis to inhibitors of metabolism: Novel insights into drug mechanisms of action, Journal of Biological Chemistry 279 (38) (2004) 40174–40184.

[2] J. J. Faith, M. E. Driscoll, V. A. Fusaro, E. J. Cosgrove, B. Hayete, F. S. Juhn, S. J. Schneider, T. S. Gardner, Many microbe microarrays database: uniformly normalized affymetrix compendia with structured experimental metadata, Nucleic Acids Research 36 (suppl 1) (2008) D866–D870.
URL `http://nar.oxfordjournals.org/content/36/suppl_1/D866.abstract`

[3] J. M. Cherry, E. L. Hong, C. Amundsen, R. Balakrishnan, G. Binkley, E. T. Chan, K. R. Christie, M. C. Costanzo, S. S. Dwight, S. R. Engel, D. G. Fisk, J. E. Hirschman, B. C. Hitz, K. Karra, C. J. Krieger, S. R. Miyasato, R. S. Nash, J. Park, M. S. Skrzypek, M. Simison, S. Weng, E. D. Wong, Saccharomyces genome database: the genomics resource of budding yeast, Nucleic Acids Research 40 (D1) (2012) D700–D705.
URL `http://nar.oxfordjournals.org/content/40/D1/D700.abstract`

[4] R. Edgar, M. Domrachev, A. Lash, Gene expression omnibus: NCBI gene expression and hybridization array data repository, Nucleic acids research 30 (1) (2002) 207–210.

[5] J. Glasner, P. Liss, G. Plunkett III, A. Darling, T. Prasad, M. Rusch, A. Byrnes, M. Gilson, B. Biehl, F. Blattner, et al., ASAP, a systematic annotation package for community analysis of genomes, Nucleic Acids Research 31 (1) (2003) 147–151.

[6] M. W. Covert, C. H. Schilling, B. Ø. Palsson, Regulation of Gene Expression in Flux Balance Models of Metabolism, Journal of Theoretical Biology 213 (2001) 73–88.

[7] B. Palsson, Systems Biology: Properties of Reconstructed Networks, Cambridge University Press, 2006.

[8] S. Chandrasekaran, N. D. Price, Probabilistic integrative modeling of genome-scale metabolic and regulatory networks in escherichia coli and mycobacterium tuberculosis, Proceedings of the National Academy of Sciences.
URL `http://www.pnas.org/content/early/2010/09/22/1005139107.abstract`

[9] J. Reed, B. Palsson, Thirteen years of building constraint-based in silico models of escherichia coli, Journal of Bacteriology 185 (9) (2003) 2692–2699.

[10] J. D. Orth, I. Thiele, B. Ø. Palsson, What is flux balance analysis?, Nat Biotech 28 (3) (2010) 245–248.
URL `http://dx.doi.org/10.1038/nbt.1614`

[11] S. Becker, B. Palsson, Context-specific metabolic networks are consistent with experiments, PLoS computational biology 4 (5) (2008) e1000082.

[12] C. T. Harbison, D. B. Gordon, T. I. Lee, N. J. Rinaldi, K. D. Macisaac, T. W. Danford, N. M. Hannett, J.-B. Tagne, D. B. Reynolds, J. Yoo, E. G. Jennings, J. Zeitlinger, D. K. Pokholok, M. Kellis, P. A. Rolfe, K. T. Takusagawa, E. S. Lander, D. K. Gifford, E. Fraenkel, R. A. Young, Transcriptional regulatory code of a eukaryotic genome, Nature 431 (7004) (2004) 99–104.
URL `http://dx.doi.org/10.1038/nature02800`

[13] B. Caballero, Analysis of the PROM algorithm as a tool to generate genome-scale metabolic-regulatory networks.
URL `http://hdl.handle.net/2142/34309`

[14] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausble Inference, The Morgan Kaufmann Series in Representation and Learning, Morgan Kaufmann Publ., 1988.

[15] F. Kschischang, B. Frey, H. Loeliger, Factor graphs and the sum-product algorithm, Information Theory, IEEE Transactions on 47 (2) (2001) 498–519.

[16] B. Heavner, K. Smallbone, B. Barker, P. Mendes, L. Walker, Yeast 5 - an expanded reconstruction of the saccharomyces cerevisiae metabolic network, BMC Systems Biology 6 (1) (2012) 55.
URL `http://www.biomedcentral.com/1752-0509/6/55`

[17] S. N. e. a. Yanushkevich, The EXOR gate under uncertainty: A case study, Facta universitatis - series: Electronics and Energetics 24 (3).

[18] J. Friedman, T. Hastie, R. Tibshirani, Sparse inverse covariance estimation with the graphical lasso, Biostatistics 9 (3) (2008) 432–441.

[19] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society. Series B (Methodological) (1996) 267–288.

[20] N. E. Lewis, H. Nagarajan, B. Ø. Palsson, Constraining the metabolic genotype–phenotype relationship using a phylogeny of in silico methods, Nat Rev Micro 10 (4) (2012) 291–305.
URL http://dx.doi.org/10.1038/nrmicro2737

[21] L. Glass, D. Kaplan, Time series analysis of complex dynamics in physiology and medicine., Medical progress through technology 19 (3) (1993) 115–128, review,Research Support, Non-U.S. Gov't,.
URL http://europepmc.org/abstract/MED/8127277

[22] V. Jethava, C. Bhattacharyya, D. Dubhashi, Computational approaches for reconstruction of time-varying biological networks from omics data, unpublished (May 2012).

[23] V. Jethava, C. Bhattacharyya, D. Dubhashi, G. Vemuri, NETGEM: Network embedded temporal generative model for gene expression data, BMC Bioinformatics 12 (1) (2011) 327.
URL http://www.biomedcentral.com/1471-2105/12/327

[24] M. J. Herrgård, B.-S. Lee, V. Portnoy, B. Ø. Palsson, Integrated analysis of regulatory and metabolic networks reveals novel regulatory mechanisms in saccharomyces cerevisiae, Genome Research 16 (5) (2006) 627–635.
URL http://genome.cshlp.org/content/16/5/627.abstract

[25] N. C. Duarte, M. J. Herrgård, B. Ø. Palsson, Reconstruction and validation of saccharomyces cerevisiae iND750, a fully compartmentalized genome-scale metabolic model, Genome Research 14 (7) (2004) 1298–1309.
URL http://genome.cshlp.org/content/14/7/1298.abstract

[26] R. Mahadevan, C. Schilling, et al., The effects of alternate optimal solutions in constraint-based genome-scale metabolic models., Metabolic engineering 5 (4) (2003) 264.

[27] B. Palsson, Systems Biology: Simulation of Dynamic Network States, Cambridge University Press, 2011.

[28] I. Nookaew, M. Jewett, A. Meechai, C. Thammarongtham, K. Laoteng, S. Cheevadhanarak, J. Nielsen, S. Bhumiratana, The genome-scale metabolic model iIN800 of saccharomyces cerevisiae and its validation: a scaffold to query lipid metabolism, BMC Systems Biology 2 (1) (2008) 71.
URL http://www.biomedcentral.com/1752-0509/2/71

[29] C. M. Bishop, Pattern recognition and machine learning, 1st Edition, Springer, 2006.

[30] J. S. Yedida, An idiosyncratic journey beyond mean field theory, Advanced Mean Field Methods, Theory and Practice (2001) 21–36.

[31] N. Friedman, M. Linial, I. Nachman, Using bayesian networks to analyze expression data, Journal of Computational Biology 7 (2000) 601–620.

[32] H. e. a. Salgado, RegulonDB v8.0: Omics data sets, evolutionary conservation, regulatory phrases, cross-validated gold standards and more, Nucleic Acids Research.

[33] G. Balázsi, A. Heath, L. Shi, M. Gennaro, The temporal response of the mycobacterium tuberculosis gene regulatory network during growth arrest, Molecular systems biology 4 (1).

[34] I. Nookaew, personal communication, PhD, Department of Chemical and Biological Engineering at Chalmers University of Technology (2012).

[35] J. M. Mooij, libDAI: A free and open source C++ library for discrete approximate inference in graphical models, Journal of Machine Learning Research 11 (2010) 2169–2173.
URL `http://www.jmlr.org/papers/volume11/mooij10a/mooij10a.pdf`

[36] S. A. Becker, A. M. Feist, M. L. Mo, G. Hannum, B. O. Palsson, M. J. Herrgard, Quantitative prediction of cellular metabolism with constraint-based models: the COBRA toolbox, Nat. Protocols 2 (3) (2007) 727–738.
URL `http://dx.doi.org/10.1038/nprot.2007.99`

[37] B. J. Bornstein, S. M. Keating, A. Jouraku, M. Hucka, LibSBML: an API library for SBML, Bioinformatics 24 (6) (2008) 880–881.
URL `http://bioinformatics.oxfordjournals.org/content/24/6/880.abstract`

[38] GLPK (GNU linear programming kit), version 4.43 (2006).
URL `http://www.gnu.org/software/glpk`

[39] MOSEK A.S., The MOSEK optimization tools manual. version 5.0 (revision 60) (2007).

[40] A. L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
URL `http://www.sciencemag.org/content/286/5439/509.abstract`

[41] E. Ising, Beitrag zur Theorie des Ferromagnetismus, Zeitschrift fur Physik 31 (1925) 253–258.

[42] N. Santhanam, J. Dingel, O. Milenkovic, On modeling gene regulatory networks using markov random fields, in: Networking and Information Theory, 2009. ITW 2009. IEEE Information Theory Workshop on, IEEE, 2009, pp. 156–160.

[43] S. Chandrasekaran, personal communication, PhD student, Center for Biophysics & Computational Biology, University of Illinois at Urbana-Champaign (2012).

[44] C. Van Rijsbergen, Information retrieval, Butterworths, 1979.

[45] M. Covert, E. Knight, J. Reed, M. Herrgard, B. Palsson, Integrating high-throughput and computational data elucidates bacterial networks, Nature 429 (6987) (2004) 92–96.

[46] P. Jensen, J. Papin, Functional integration of a metabolic network model and expression data without arbitrary thresholding, Bioinformatics 27 (4) (2011) 541–547.

[47] T. Shlomi, M. Cabili, M. Herrgård, B. Palsson, E. Ruppin, Network-based prediction of human tissue-specific metabolism, Nature biotechnology 26 (9) (2008) 1003–1010.

[48] S. Boyd, L. Vandenberghe, Convex optimization, Cambridge university press, 2004.