Tel Aviv University
Raymond and Beverly Sackler
Faculty of Exact Sciences
School of Mathematical Sciences

Computational Expansion of Genetic Networks

by

Amos Tanay

The research work has been conducted
under the supervision of
Prof. Ron Shamir

Submitted as partial fulfillment of the requirements
towards the M.Sc. degree

November 2001

Acknowledgment

I wish to express my thanks to Prof. Ron Shamir for not giving up on teaching me how to transform vague ideas into science . I also wish to acknowledge Itsick Pe'er, Zohar Yakhini, Martin Kupiec and Jacques Tanay for helpful discussions, ideas and support along the way.

Abstract

Computational genetic networks are mathematical models for biological systems. Biological systems were shaped by evaluation as extremely complex, parallel and robust entities. The many biological functions that are essential to the existence of any leaving organisms are achieved via interaction of genes, proteins and external signals where the nature of interaction is as important as the potency of specific biological factors. The emergence of modern DNA sequencing techniques allows us to fully describe many organisms genomes, but creating long lists of genes cannot by itself shed light on their function, or on the interaction scheme that tie them together. The technology of DNA chips, with its many applications and variants, provides high throughput measurements of thousands of biological factors (mRNA, binding sites). The work in this thesis is focused around the problem of **network reconstruction**: Given a large data set of measurements, what is the model that best explains the data. We develop the concept of **network expansion** which uses a known sub model as a starting point for the network reconstruction process. The motivation for expansion is the combinatorial explosion of data needed for "de-novo" reconstruction and the attempt to provide computational tools that may be of help to the biologist specializing on a subsystem of a complex organism. We have formulated new network models that can integrate biological constraints and built methods and algorithms for evaluation of models vs. experimental data. The entire environment was implemented in a new software system called GENESYS. We used GENESYS to evaluate the performance of our methods on both simulated and real life data sets. The results show that the network expansion methodology as implemented in GENESYS can be used to improve our understanding of relevant and important biological subsystems.

Contents

Chapter 1

Introduction

The work described in this thesis deals with the computational identification of biological relations among genes and proteins in a living organism. This problem is of major scientific importance, even more so in the post-genomic era, where the dictionary of biological factors (genes, proteins) is at hand or within reach. Biological systems are the outcome of an evolutionary process, and they manifest phenomena which are massively parallel and stochastic. This shapes biological networks (composed of the entire set of biological variables and the relations among them) as an extremely complex entity, having numerous levels of redundancies in one hand and composed of multi-function factors on the other hand. The vision of creating a biological network reconstruction "pipeline" that will rapidly analyze and "reverse engineer" the biological machinery is thus as difficult as it is attractive.

The advent of novel biological analysis methods, most notably the ability to profile gene expression on a genomic scale using DNA chips, has changed the way biology is being explored. Huge databases of biological data are accumulated in an accelerated pace, stressing more and more the challenge of making sense out of the data. Still, as we will explain below, the amount of information needed for a naive computational attack on network reconstruction suffers from combinatorial explosion and will always be impractical to obtain.

One way to address these difficulties is by introducing new experimental techniques, collecting focused and high quality information on the relations among biological factors [43, 35]. A complementary approach, which is the subject of our work, is to use sophisticated computational methods that would enable biologists to ask directed questions on specific functional units and to get statistically sound answers. These methods should support the incorporation of large datasets and use them to generate and validate high quality biological hypotheses.

1.1 Our Approach

In this section we sketch the approach taken by this study. The terms used here will be defined more fully and formally later.

As we shall explain below in detail, the challenge of genetic network reconstruction is difficult due to many reasons. The main reason being the complexity of a massively parallel, redundant and "evolution made" system. Compared to the numerous degrees of freedom in the space of possible molecular networks, even datasets with hundreds of well planned distinct cellular states cannot directly identify the perfect model out of many other possible alternatives. Another major source of difficulties is the noise inherent in high throughput experiments and the need to either give away much of the data or tolerate significant levels of "false positives".

The methodology developed in this thesis offers computational devices that improve our ability to handle both complexity and noise. A **network model** consists of a set of elements (e.g., genes, proteins) and the laws of interactions among them. Given an ensemble of possible network models (the **models space**), tailored for the description of a selected target system, we define a **network core** as a sub-network consisting of a set of known elements and interactions among them. Cores can be constructed manually by an expert with a focused interest on a specific pathway or directly from a database containing collections of known pathways. Our approach avoids "de novo" reconstruction and instead tries to add factors and interactions to the core so that a given dataset is predicted better by an extended core model. We call this process **network expansion**. Limiting our search to expansion of a given network core leads to a major reduction in model space dimensionality and enables the algorithm to both spend more computational power on a target subsystem and to be less sensitive to over fitting. The core serves us also as a means for integrating diverse sources of information into the computational process.

We have implemented the expansion methodology with discrete network models and formalized the required concepts and mathematical models. Discrete network models are not new [40, 2] but we have generalized them to better suit our needs. Our models distinguish local logic constraints and global network topology limitations and uses multiple valued logic to represent different states and control of biological variables.

To improve our process specificity, we have developed specialized scoring functions to assess the fit of each possible core expansion to the input dataset. Our methods enable the approximate calculation of a p-value for a speculated regulation pattern and simulation studies show that compared with conventional scoring method (mutual information for example) the novel scoring function gives more specific results. In an attempt to further improve the utilization of high throughput dataset we have built a probabilistic representation for expression data sets and generalized our scoring functions to use it.

A new software platform, named GENESYS (GEnetic Network Expansion SYStem) was developed and used to test the framework with real biological information. We have focused on budding yeast and used publicly available transcription profile data to generate likely expansions of ergosterol related pathways. The results suggest a novel transcription factor and identify interesting regulation patterns, proving that computational analysis can reveal complex relations in genetic networks, even with today's data sets.

Using GENESYS simulation engine, we have also tested our algorithm with simulation of networks and datasets. Results show that, under our model hypothesis, very specific results can be obtained. We have also compared different scoring functions and analyzed the method robustness to noise with good results on both aspects.

GENESYS was built to support an interactive workflow and should be considered as a prototype for a biological network analysis system. The many viewers in GENESYS allow various ways of viewing and manipulating network concepts, and are connected with the computational back-end to enable an on-line expansion workflow. Our algorithms were designed and implemented to provide very fast response time and are capable of expanding cores of more than a dozen variables within minutes on a standard PC, so interactive manipulation of the network is really possible.

We believe the expansion methodology may prove a valuable tool for the working biologist who is interested in a specific function of a living organism and cannot deal simultaneously with thousands of genes interacting with each other. In the future, if a large database of transcription profiles and other datasets is available, an expansion process can serve as a search tool giving the expert of a particular biological domain the freedom to take advantage of the whole dataset in a focused way.

1.2 Outline of Thesis Results

The main results of this thesis are listed below, a perliminary version of most of these has been published in [47] :

a. We formulate a new network model that generalizes prior work by allowing multi-valued variables and logic and incorporating noise.

b. We define several new measures to assess the fitness of a network and show how to compute or approximate them.

c. We define a new computational challenge, the pathway expansion problem.

d. We study the complexity of several variants of the expansion problem and provide hardness results.

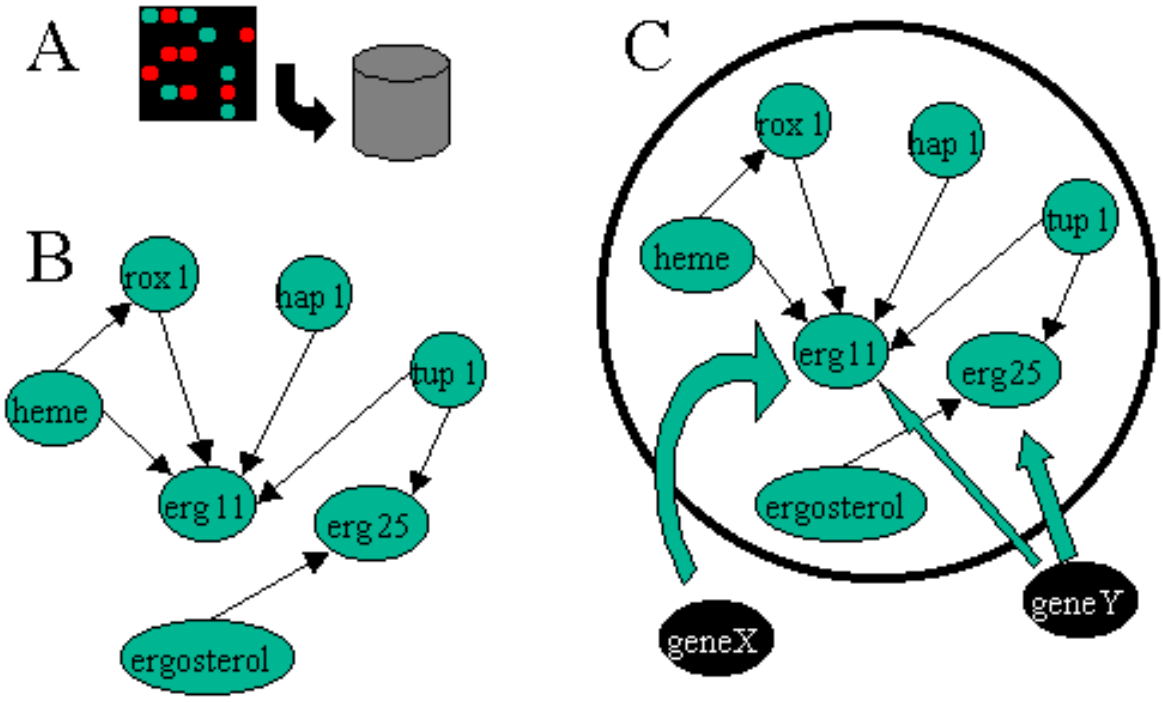

Figure 1.1: Overview of the core expansion methodology. A) Large data sets (e.g. expression profiles) are transformed into a uniform database. B) A pathway core is constructed based on the literature. The core shown here is the dependency structure of part of the yeast ergosterol pathway, consisting of 7 variables. The exact logic of the system is defined by the association of a discrete function to each model variable. C) The core model is expanded with additional variables and interactions. Expansions are scored by their level of fitness to the database.

e. We define a sub-challenge of the pathway expansion problem which is more tractable, the single node expansion problem.

f. We develop a software system for modeling, visualizing and manipulating networks, and for solving single node expansion problems.

g. We study the performance of the new system on both simulated and real biological data.

The thesis is organized as follows: Chapter 2 contains biological and computational background. Chapter 3 contains results a-e. Chapter 4 describes simulation results and Chapter 5 describes results on real biological data. Chapter 6 describes the software system.

Chapter 2

Background

In this chapter we provide basic background on genetic networks. We define and illustrate the elements of such networks and their different relations. We also describe prior work on computational modeling and analysis of networks. The biological literature is full with excellent expositions on the various aspects of regulation in biological systems (e.g. [5]). We provide in this chapter sporadic examples from it to motivate later modeling decisions.

2.1 Biological Motivation

Every organism in nature is completely characterized by its DNA sequence (with some exceptions that are irrelevant here). The DNA provides instructions for the synthesis of proteins which function in a wide variety of biochemical reactions as amazingly efficient, robust and flexible molecular machines. This was the basic paradigm of later 20th century biology, and the source of many puzzling questions still awaiting to be resolved.

Each cell in a multi cellular organism contains duplicates of the same DNA molecules, with the same instruction for the synthesis of the same proteins. Still, two of these cells can be remarkably different from each other in size, shape and function (compare a large neuron to a lymphocyte) and produce an entirely different collection of proteins. What are the details of the sophisticated program enabling a human egg to transform, in a well defined and tightly reproducible process, into the detailed and complex system of cells forming a human being?

A different aspect of basically the same question can be seen in much lower organisms. Yeast, the popular model system, can sense and respond to external stimulations (heat, starvation, antibiotics) by changing the concentration of important proteins and performing controlled modification in others. This is done without changing the DNA or learning to produce new proteins. How are such programs implemented? How did these mechanisms

evolve?

Having studied molecular biology for half a century, science is now in a position to start and understand how complex biology actually works. The basic paradigm of DNA-RNA-Protein-Function is still the key, but now it is only the basic pattern of a complicated **biological system** or **biological network**. A biological system is composed of a large number of biological factors (different molecules and metabolites, including DNA, RNA and proteins). The different factors interact with each other in a variety of biochemical reactions (protein synthesis, mRNA transcription, protein docking and many others). The exact nature of interaction determines how the system would react to different conditions and how it should develop in time. The details of interaction are still determined by the DNA (which acts as the only inheritance device) but we must look not only for genes (segments of the DNA used for generating proteins) but also for other **regulatory** signals. We must also examine proteins in a broad context, searching for the relations with other proteins and their potential regulatory role.

Before giving some examples on important regulation mechanisms, let us examine the more global aspect of the architecture of a biological network. It turns out that natural systems such as the two mentioned above (human cell differentiation and development, fungi adaptation to environment) are completely different from systems built by humans. The engineering ideals of simplicity and precision through well defined components are not the dominating rules in biology. The reason for that is evolution. Consider, by way of analogy, how evolution would try to turn a car into a airplane. In order to do so, a viable evolutionary path should exist in which each intermediate step has a chance to survive (i.e. have high **fitness**, e.g., in car/plain terms, it must be able to move and carry passenger efficiently and reliably). But a car is extremely vulnerable to perturbations. Changing a single component of the cylinders, fuel system, cooling system or wheels would create a useless piece of metal. This would completely prevent the emergence of wings or jets since mutations would have a huge chance of simply destroying the species.

Biological systems are built differently. As an outcome of evolution they are redundant and robust. Not only is the genetic code degenerate, but also many of the interactions and relations are backed by alternatives. Single points of failure would rarely exist and this enables evolution to explore new possibilities without destroying useful building blocks. This is greatly facilitated by the mathematical nature of the **genetic network**. Theoretical and simulative studies have shown [38] that a random boolean network can feature a surprisingly simple underlying structure when adequate parameters are chosen (see below). Specifically, although there might be a very large number of interacting factors, and although it may seem that the system can behave chaotically, order can be found in the network state space such that a relatively small family of network modes ("attractors", in the language of dynamical systems) exists. This feature enables a large collection of factors to behave in an ordered, non

chaotic way and provide evolution with a relatively smooth and non rugged **fitness landscape**. The fitness as a parameter of the DNA would behave reasonably since changing few parameters of the system would have a low probability for dramatic effects on the structure of network modes.

Biological systems are thus complex yet ordered, and the order is introduced not by a strict, engineering style hierarchal organization but via redundancy and stability in the presence of small changes. The challenge of exploring these entities thus involves an interdisciplinary effort from biologists, computer scientists and mathematicians. The numbers play an important role in understanding why standard techniques would fail. There are about 40,000 genes in human [39] producing well over 100,000 proteins that may go through various modifications to generate a total of over a million biological factors in this system. Assuming we would like to have an analog of a physics field theory or state equation for this system (the good old scientific ideal), this means we should write down equations for yet an order of magnitude more (10^7) reactions between factors. This task cannot be done manually and it is not like anything we have attempted before. It is the opposite of the 20th century physics attempt to unify a theory, since the complexity is not an artifact waiting to be canceled by an ingenious mathematical trick: It is the object itself.

We shall now describe some of the biological effects playing key roles in molecular networks. The first of these is called **transcription control**, and provides the basic understanding for the reason why cells with the same DNA may look so different. The protein synthesis pathway starts by transcribing a gene into an mRNA molecule. The biochemical reaction responsible for transcription involves the assembly of a complicated protein complex (usually TF-II but sometime TF-I or TF-III) to a site near the gene sequence start (the promoter of the gene). As always in biochemistry, chances of spontaneous TF-II assembly at a gene promoter are negligible and some catalysis is needed to actually produce mRNA. This catalysis is performed by a dedicated family of proteins called **Transcription Factors**. Some of these factors are **general** and bind to any DNA sequence, with low sequence specificity. These may be related to un-packaging DNA secondary structure (nucleosome packaging of DNA may prevent transcription). The more interesting family of transcription factors are **sequence specific**, i.e. can bind only special DNA motifs. This limits them to act only on a specific subset of genes that have this motif in some proximity to their transcription start site. Studies have showed that each gene may have more than two dozen DNA binding sites and associated transcription factors in the short 2kbp sequence upstream of its TATA box (gene start). The exact combination of transcription factor sites differentiate one gene from another and can be viewed as the input of a complex genetic switch.

At each cell, and in each condition, the gene DNA binding motifs (categorized as cis regulatory sites) may be bounded by transcription factors (TFs) or missing them. The exact combination of existing bounded TFs may enable or disable transcription invocation. The

interaction between the various transcription factors may be formulated as a complex logical expression, with some TFs acting positively and some negatively, or with others that may have no effect unless their hetro-dimer partner is also available.

The mechanisms of transcription control were used by evolution to build a detailed network of TF-gene interactions. The key point here is that TFs are protein themselves and so are also subject to transcription control. This means that the global system of transcription regulation looks like a network in which inputs of one gene are the output of others, similarly to the abstract boolean network models we will describe below (and of course this was the inspiration for these models). The cis acting motifs, being part of the DNA, are inherited and are thus manipulated by evolution which can not only modify proteins and their structure but also add or delete links in the network.

Biological regulation cannot be entirely dominated by transcription control. First of all, the cell must sense external signals, and only then can the genetic network be used to respond. The cell sensors are usually membrane proteins that change their conformation in response to changes in the environment or the arrival of some external signaling protein (for example hormone or an immune cell). The conformational changes may change the catalytic potency of the sensor and trigger modification in other, internal proteins that may be TFs in the simplest case. Such TFs will lack catalytic potency as long as their conformation is not changed, and so the external signal may be transmitted from the membrane signaling protein through the TF activation into the genetic network.

The example of the simplest signaling cascade noted above reflects a general pattern in molecular networks. The conformational modification, usually achieved via phosphorilation of selected amino acids in the protein is a type of **post translational** control device. A special family of proteins (Kinases) can specifically phosphorilate given targets and change their catalytic potency. We have noted that such devices are essential in signaling cascades, but their role extends much beyond. Two other types of post translational control device are **cooperative binding** and **ubiquitination**. The first of these may allow one active protein to act only in the presence or absence of a catalytic or inhibitory partner. The second is used to tag proteins for destruction by the proteosome.

Why did nature create both transcriptional and post-translational controls? The first answer is that nature does not look for simplicity in the engineering sense, but there are many additional explanations. Transcription control, as we saw above, is easily manipulatable by evolution and provides much power and robustness. Post-translational control is on the other hand much faster (two order of magnitudes) and enables fast signaling and "real time" controls. It is the combination of the two that makes complex biological functionality possible.

We will conclude our discussion of basic biological regulation mechanisms with an example from the relatively well characterized regulatory circuit of yeast (s. Cerevisea) cell

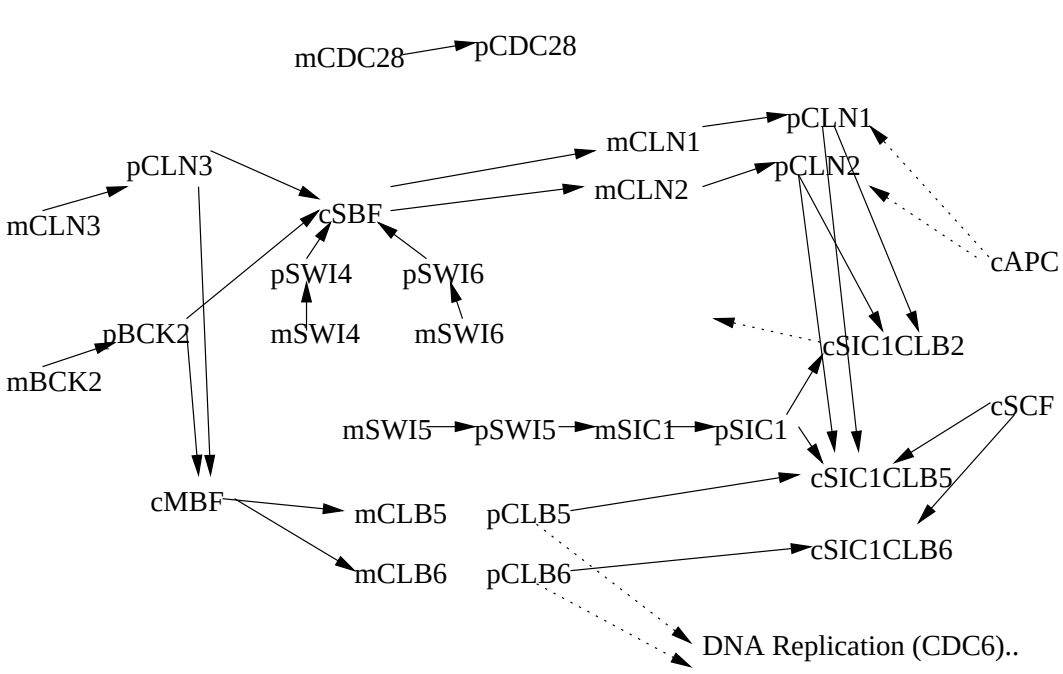

Figure 2.1: Fraction of the yeast cell cycle regulatory network. G1 phase regulation is modeled as a dependency graph with 3 types of nodes, mRNAs are denoted with an m prefix, proteins with a p prefix and complexes are denoted by c prefix. An arrow from x to y indicates that the level of x is directly effecting the level of y. To proceed through the cell cycle, transcription of G1 cyclins (cln1,cln2,clb5,clb6) must raise at the G1 phase, and this is achieved by activating two transcription factor complexes (MBF,SBF) post translationally. Function of the clb5,6 proteins is however inhibited by high levels of another protein, sic1, and only when another post translational reaction takes place (phosphorilation of sic1 by cln1 and subsequent ubiquitination and degradation of it via the SCF complex) the important function of those cyclin can take place. These examples from G1 phase regulation show the importance of the different regulation mechanisms to the understanding of the whole biological process.

cycle control. The system is controlled by three families of proteins and genes. The Cdk (Cyclin Dependent Kinase) cdc28 serves as a platform for catalytic activity and is being cooperatively bound by a group of cyclins proteins. Each of the complexes cyclin + cdk is responsible for catalyzing a critical step in the cell cycle. The abundance of the cyclins is controlled by transcription factors which invoke their transcription at well defined points of the cycle. Important protein complexes (APC, SCF) are responsible for tagging specific proteins (ubiquitination) so that their abundance would fall sharply when their function is completed. Many signals may lock the cell cycle at specific steps (Pheromone response, DNA damage). This is done by inhibiting critical post-translational reactions that are required for the completion of the cycle. (see figure 2.1).

Budding yeast cell cycle control is considered as one of the best understood biological control systems and is still only partially characterized. How can we approach the vast complexity of biological networks in larger scale and make their systematical modeling possible? Until recent years, this challenge was attempted via step by step experimental trial and error process, analyzing few genes/proteins in each experiment and trying to gain insights both on the general mechanisms of regulation and on specific regulatory pathways. The efforts of 40 years or so resulted in good understanding of the regulation mechanisms. Still, progress was largely limited by the amount of information laboratories could have generated.

Recent years have witnessed an information revolution in biology with the introduction of high throughput experimental methods. This revolution builds the foundation for a larger scale understanding of biological networks. The most notable high throughput technology in use today is DNA arrays [1, 21, 20]. Using high density cDNA or oligonucleotide microarrays, biologists can measure mRNA levels in a genomic scale. While this technology becomes more and more mature, the quality and reproducibility of results increases and additional standard chips are being developed (following the completion of various genome projects). The output of a single chip experiment is an **expression profile**, i.e., a vector of mRMA expression levels of all genes present on the chip. cDNA microarrays are used to measure ratios between expression of a baseline culture and the target cellular state. Oligo chips (as produced, for example, by Affymetrix) output absolute values for expression levels. In both techniques, only relatively large changes within or between experiments are considered significant. This is due to both biological and experimental noise.

As mentioned above, mRNA levels by themselves are extremely important biological factors and biologists have already used DNA chips to dramatically expand our knowledge about many biological systems [29, 28, 32, 46, 26]. The technological possibilities of advanced materials, robotics and image analysis go much further than measuring mRNAs, and new important technologies provide insight into both transcription and post-translational effects.

Using Immunoprecipitation, PCR amplification and DNA chips, researchers at Stanford and MIT [43, 35] developed a method for the genomic scale prediction of transcription factor binding affinities. Using this method, the promoter region of each gene can be tested to see if a specific transcription factor bind it. Although many effects contribute to the noise in such experiments (most importantly effects of chromosome packaging), such data provide a direct view into the genetic network and compensate for many of the problems in using only expression data.

New types of chips are being suggested to measure various properties of proteins directly. Brown et al. [23] have shown how membrane associated proteins can be searched for, Uetz et al. [50] used high throughput two hybrid experiments to identify a large set of putative protein interactions. These methods are complemented by emerging proteomics techniques [54, 30, 31] that can measure the abundance of a large number of proteins, similarly to the

way DNA chips can do for mRNA.

The biological information revolution is accompanied by efforts for standardization and automated data sharing via database systems and the Internet. Organism's databases [7, 16, 14] provide standard reference for the genome and annotation of each model system and are referenced by the entire community. Functional databases [51, 36] provide a specialized, cross organism view of biology. The Gene Ontology consortium [15] is developing a generic biological vocabulary that should make the translation of data and models across organisms possible in a semi automatic way. These databases and atlases are a powerful resource in using diverse data sources for genetic network analysis.

The incredible pace in which new methods are being invented and utilized, and the increasing rate in which high throughput data is being collected make the vision of characterizing biological networks realistic. The mathematics, statistics and algorithmic involved in such a huge task are however in very early stages.

2.2 Previous Computational Work

We shall next review the current literature on computational genetic networks. As a young field, one cannot identify a central body of work which dominates the domain. Rather, there are different theories and techniques which may merge in the future and create a more unified scientific discipline. As a result of this situation, one should pay particular care to assumptions and terminology used in the different studies.

The mathematical formulation of a genetic network has been introduced in early works on mathematical biology. An important example is Kauffman's work on the evolutionary implications of various biological structures [38]. Kauffman noted the effect of network structure on evolutionary stability and studied the evolving network and the possible advantages of special organization of its state space. In particular, synchronous boolean networks were suggested as a idealized model for a biological system. A **boolean network** N is defined on a set of variables U by a set of boolean functions $f_v : \{0,1\}^U \to \{0,1\}$ for $v \in U$. The simplistic biological interpretation of a boolean network associates each variable with a gene and interprets 0 values as deactivation of the gene and 1 values as activation of the gene. Transcription control logic can be modeled by encoding the relations between variables (transcription factors and genes) into the functions f_v. The **state** of a network is the vector of the levels of all variables. The model is a **synchronous** dynamical system, whose state at time t depends on the values of variables at time $t-1$. As a dynamical system, one can analyze the network behavior under different initial conditions or perform global state space analysis looking for attractors and basins of attraction. **Attractors** are closed trajectories (cycles) in the state space of the dynamical system; **basins of attraction** are equivalence classes of

states leading to the same attractor (since the state space is finite, every state must lead to an attractor).

Kauffman studied random boolean networks behavior and noted that while random networks are chaotic in their behavior, enforcing bounded in-degree, i.e. limiting the direct inputs of a variable to some small number, introduces order in the state space and results in a number of basins of attraction that is less than square root the number of nodes. Kauffman further interpreted each attractor as a distinct cell type and claimed that during the course of cell differentiation, careful sorting of global states into local attractors takes place. The ability to obtain organized state space using weak constraints, claimed Kauffman, was an important factor in the evolution of higher organisms. Researchers at Santa Fe institute used Kauffman's ideas to develop the concept of a mathematical genetic network. Boolean models were randomized and analyzed exhaustively to explore their attractor space structure and to draw conclusions about the possible effect of known biological features on them.

The vision of computational network reconstruction from experiments became more and more widespread with the first works on DNA chips circa 1996. Repeating DNA chips experiments in different environment and cellular conditions yields a matrix of expression level per gene per condition. Combining this novel technology and the emerging mathematical formalization of genetic networks, Somogyi et al. [40] suggested a **network reconstruction** algorithm. The basic goal is natural: Given a large data set of expression profiles taken from different states of a cell, reconstruct the underlying genetic network such that the measurements are explained in the best possible way. So, looking at a specific gene, and assuming our model is boolean, we should look for a set of functions f_v such that the behavior of each gene is completely defined by the states of its inputs. Somogyi et al. developed their theoretical reconstruction algorithm while making many assumptions on the data and the biological model. The data was assumed to describe a trajectory of the synchronous boolean network, i.e. the measurements at time $t-1$ completely determine those at time t. The topology of the network was simplified by allowing a limited number of D inputs for each gene. Under these assumptions, the reconstruction algorithm is very simple: scan all D-subsets for each gene and take a subset for which the input boolean expression states always match the output expression state.

In a subsequent work, D'hasslear et al. [22] tried to model regulation in a linear framework. This time, a linear equation was used to associate the expression of each gene with the expression of its input genes. The linear approach is limited in its ability to describe complex logic but can handle the real values measured in the DNA chips experiments without preprocessing them by discretization. The results of using a linear system for the analysis of a true biological system (mouse central nervous system) were naturally severely over-fitted (the number of parameters needed to be determined was quadratic in the number of genes, while the number of available states was sub-linear). The known biological mechanisms of

regulation are non linear by nature, and indeed linear models are not an attractive approach to study gene regulation.

Much of our understanding of biological regulation, and the inspiration to study computational problems related to it came from the works of Davidson and colleagues [53, 19]. Studying the sea urchin model organism, Davidson was able to construct very detailed examples of specific gene regulation. Well planned perturbations of the promoter region were performed in order to identify cis regulatory sites, compose them into **modules** and analyze their logical relations. The resulting logical apparatus is an impressive, sophisticated computational device. One should note that the examples studied by Davidson relate to the cell differentiation process and to development rather than to "real time" control mechanisms. Developmental biology is more naturally modeled by discrete genetic switches than other effects.

A key to the research of biology in general, and of biological networks in particular is the ability to perturb the genome via gene deletion, modification or over-expression. Akutso et al. [2, 3, 4] studied the mathematical problem of boolean network reconstruction via perturbations in a biologically inspired setting. In [2], Akutso et al. examined the worst case behavior of an experimental plan in which small sets of genes are perturbed and the effect on all genes expression levels is measured. Upper and lower bound were given on the number of experiments under various assumptions regarding the structure of the direct dependencies among variables in the network. The outcome of the combinatorial analysis implied that it will take an unrealistically large number of experiments to reconstruct a network in the worst case. It would take $\Omega(N^D)$ and $O(N^{2D+1})$ perturbation on a network with N nodes and maximal in-degree D to completely characterize a boolean network. In fact, we can improve the upper bound to $O(N^{D+2})$. Even when assuming an acyclic network, this lower bound cannot be removed as it is a direct consequence of the case of a single node logic reconstruction: The case where the level of a single node is an AND function of the levels of D other nodes cannot be identified by less than $\Omega(N^D)$ experiments in the worst case.

A more optimistic approach was described in a latter paper [3] where it was shown that in a randomized experimental plan, a logarithmic number of experiments is enough in order to reconstruct the network with high probability. However, this result can be obtained only when assuming a random distribution of samples, independently for each of the variables. This is a problematic assumption since in practice many perturbations lead to the same attractor and thus produce highly dependent samples.

The computational problem of bounded in-degree boolean network reconstruction is defined by the set of expression profiles (of size m) on a set of variables (of size n). The goal is to find an assignment of boolean functions to the variables that would best explain the observed expression levels. The trivial $O(mn^{D+1})$ algorithm suggested in [40] was improved in [4] using fingerprinting and matrix multiplication techniques to $O(n^D m^{\omega-2} + n^{D+\omega-3}m)$

time, where ω is the matrix multiplication algorithm exponent) . The improvement remains theoretical, since efficient matrix multiplication algorithms are not practical.

From a different direction, researchers have tried to address the related problem of optimizing an experimental strategy for efficient network reconstruction. Karp et al. [37] aimed at model verification. A specific logical circuit is assumed as a model that must be verified experimentally and some of its nodes are defined as **stimulators**. The goal is to find a small set of value assignments to the stimulators, such the measured outputs verify the correctness of the model. The problem, which is easily shown to be NP-hard, was approached via branch and bound algorithm and some results on a real biological pathway (pheromone response in yeast) were presented.

Ideker et al. [34] addressed the related problem of dynamic experiment planning. The authors assumed a boolean acyclic network and devised an algorithm for selecting the next perturbation experiment given a current set of attempted perturbations and their resulting expression profiles. The idea was that each experiment invalidates a sub model space by making some logic and dependency assignments contradicting the measured profile. A good perturbation should thus have high probability for disqualifying as many models as possible, or, in an alternative formalization, have high expected mutual information on the model space. The exact calculation of this probability is impractical and the authors suggested a heuristic approximation to assess it. The method was tested with simulations. As of today, there is no published practical use of this or any other method in genome wide experiment planning, but efforts in these direction are emerging [33].

In the studies described above, researchers have created theoretical frameworks for the computational manipulations of genetic networks. Important biological data sets that have been published in the last three years [46, 32] both enable and require practical analysis and modeling of specific biological systems. The analysis of microarray data is currently dominated by clustering algorithms [6, 24, 45] which we shall not review here. Researchers have also tried to combine clustering techniques with DNA motif search algorithms and apply them to expression data [10, 48]. Such a "pipeline" yields motifs which accurately predict known cis sites. However, it cannot handle the complex logic models typical to genetic network techniques and aims at the identification of objects (motifs, clusters) rather than relations.

Using a yeast cell-cycle dataset [12], Chen et al. [11] attempted genetic network reconstruction in a full scale. The time-series data was first analyzed using signal processing techniques and then discretized to activation and de-activation events. The genetic network model assumed was a digraph with edges marked as activators or repressors, requiring each node to have at least one activator and one repressor. The reconstruction algorithm searched for a minimal set of regulators that will completely "explain" the dataset (each of the activation and de-activation event would be explained by at least one edge). The problem was

shown to be NP-hard and a 2-approximation algorithm with was described based on linear programming relaxation. A simulated annealing heuristic was used to attempt reconstruction of an actual network but no significant biological results were obtained. This early work may be considered as one of the first attempts to try and computationally reconstruct a genetic network using a large dataset. Although its biological reasoning is non-standard it might have been more successful if the cell-cycle data set were less noisy and less dominated by post-translational events.

In today's largest expression datasets, hundreds of samples are characterizing expression of thousands of genes. This makes any network reconstruction attempt very vulnerable to over fitting. The high level of noise in the readings of DNA chips further complicates the task of creating a practical network algorithm. One of the possible approaches for overcoming these difficulties is to take a probabilistic approach and apply aggressive feature selection to filter out non specific results. The work by Friedman et al. [25] outlines a Bayesian genetic network learning framework to improve the validity of the results obtained by network analysis.

The Bayesian network paradigm assumes that the set of genes can be characterized by the joint distribution of their expression levels. A Bayesian network is an economical representation of such a distribution in which an acyclic directed graph encodes independencies of variables (Given its parents values, a gene's level is independent from the levels of all other genes). The learning of a Bayesian network from examples uses a Bayesian local score function that assesses the fit between examples and a given network topology. The Bayesian Dirichlet equivalent (BDe) score function is a notable example. Given the local node score, and assuming the global network score is decomposable (e.g., as sum of local scores), one can view the learning process as an optimization problem and apply local search algorithms to find a good model. To improve their techniques robustness, Friedman et al.[25] used non-parametric bootstrap on the data set to re-apply their learning procedure repeatedly and filter out dependencies with less than a given percent repeats across the bootstrap instances (i.e., only dependencies that were learned more then a certain percent of the time were considered specific).

The framework was further extended in [42] to enable incorporation of genetic perturbation information and to explore sub-networks of high significance. The extended methodology searches ensembles of learned networks (generated again using non parametric bootstrap) and locates separators (sets of nodes whose removal makes two nodes independent), cliques (subgraphs with high confidence on many of their edges) and more. When applied to the Rosseta compendium data [32], results were predictive to known biological pathways and suggested novel reasonable predictions of interactions.

Several authors constructed databases that can represent known network structures [36], [51]. Indeed, years of biological exploration have created a vast body of knowledge regarding many pathways and regulation mechanisms. A first attempt to systematically associate

existing knowledge with expression information was reported in [55], were metabolic pathways were correlated with expression information. The authors scored a set of alternative pathways (variations on galacatose metabolism in yeast) according to their match with expression data. The scoring function used was based on correlation and time dependence. The use of a metabolic pathway (series of enzymes responsible for the catalysis of a series of biochemical reactions) may limit the practical results of such methodology, since such pathways do not always reflect direct regulatory effects. The expression profiles of downstream enzymes need not be directly correlated with that of their predecessors (the whole set of enzymes may be clustered together, but the internal relations may be much more complicated and involve different regulatory functions applied to critical steps of the pathway).

A different use for large interaction databases and assays is to try and review the global structure of a genetic network. Combining information from many sources (two hybrid assays, locus proximity, orthologous genes) different authors [41, 50, 44, 52] suggested such an analysis. The graphical model of the genome was used to try and relate different functions and cellular components by arguing that whenever many edges connect genes with one function to genes with another function one may claim there is a specific association between the two. The focus in these works was more on the analysis of a given network structure than on a computational attempt to reconstruct the network.

Chapter 3

Theory

3.1 Theoretical Framework

In this section we provide definitions formalizing our model, and explain our modeling decisions. We shall first define the models space which will later be used as our search space in which attractive models should be located. We then formalize the concept of an experiment and associate the two entities (models, experiments) using a scoring function (modeling fitness).

Definition 1 *A* **biological network** *(or* **model***) is defined by a set U of* **variables***, an alphabet C of* **values** *or* **states** *that the variables may attain, and* **functions** $f^v : C^{\|U\|} \to C$ *for each $v \in U$. We use the term* **arguments** *of f^v for the non trivial arguments of the function. (u is a trivial argument of f_v if changing the value of u alone never alters the value of the function.) We denote by $arg(f^v)$ the set of arguments of f^v. The* **dependency graph** *of N is a directed graph $G(N) = (U, A)$ where $(u, v) \in A$ iff u is an argument of f^v). The set of arguments of v in G is $arg^G(v) = \{u|(u, v) \in E\}$. We shall use the term digraph for dependency graph when there is no ambiguity.*

The set U of biological variables may be composed of genes, proteins, mRNAs or metabolites. It can also include artificial "signal" variables indicating the occurrence of an event or stimulation. The interpretation is that the value of variable v at time t depends on the values of its input variables at time $t - 1$, and the functional dependence is described by f^v. An example of a network is given in Figure 3.1 and a dependency graph is shown in Figure 1.1.

One should note three major restrictions we have already enforced on our models. First, our value alphabet is discrete, which means that our network logic is completely discrete. Second, our functional dependencies (f^v) are deterministic, which means that stochastic behavior cannot be directly described in the model. Finally, we are assuming Markovian

property of the network and even more importantly, we discretize the time which means that our logic is synchronous. We shall discuss the reasoning behind those restrictions shortly, but first introduce some additional structure over families of network models to create the model space.

Definition 2 *A* **model space** *is defined by the four-tuple* (U, C, F_{bio}, G_{bio}) *where* U *and* C *are as above,* $F_{bio} \subseteq \mathcal{F} := \{f : C^{\|U\|} \to C\}$ *is the class of* **candidate functions** *and* G_{bio} *is a class of dependency graphs on* U*. The space consists of all networks with functions from* F_{bio} *and dependency graphs from* G_{bio}*.*

The set of allowed functions F_{bio} should reflect biological knowledge and realistic constraints. For example, $MONO_d$ is the set of monotone functions with at most d inputs. In contrast with F_{bio} that constrains the properties of each particular function, G_{bio} allows one to prescribe constraints on the overall architecture of the network. For example, $INDEG_r$ is the set of graphs with in-degrees at most r, $MAXREG_r$ is the set of digraphs having at most r nodes with outgoing edges (those nodes would be interpreted as the *regulators* of the network), and $DIAM_k$ is the set of all graphs with diameter at most k.

Examples :

1) In our study of transcription regulation in yeast using gene expression data (Chapter 5) the following model space is used: U was the set of all mRNAs corresponding to ORFs in budding yeast. C was the set $(-1, 0, 1)$ representing down, normal and up regulation of transcription. G_{bio} was set to $INDEG_r$, and F_{bio} was left unconstrained.

2) When trying to model cell cycle control in yeast, our variable universe U contained not only mRNA variables but also a variable denoting the protein encoded by each gene. In addition we have used special variables to denote the important transcription factors complexes and cell cycle related events (chromosome alignment, mitosis, etc.). We have also extended the alphabet C with special states indicating phosphorilation of specific proteins (see Figure 2.1).

3) One can extend the framework outlined above by variable classification information. A finite set of types is introduced and a mapping from U to the type set is defined. Types are now used to control the dependency structure and logics. We can use different F_{bio} for each type or include type based constraints in G_{bio} construction. For example, when studying transcription regulation, one can identify the class TF of putative transcription factors and introduce G_{bio} for which only variables from the TF class can have outgoing edges.

In our manipulation of network models, we will often use dependency structures instead of the detailed discrete model. The main reason for this is that often we may still have insufficient information to infer precise functional relations. Inferring dependencies only is less prone to over-fitting, yet it provides key information on the network. We can use the

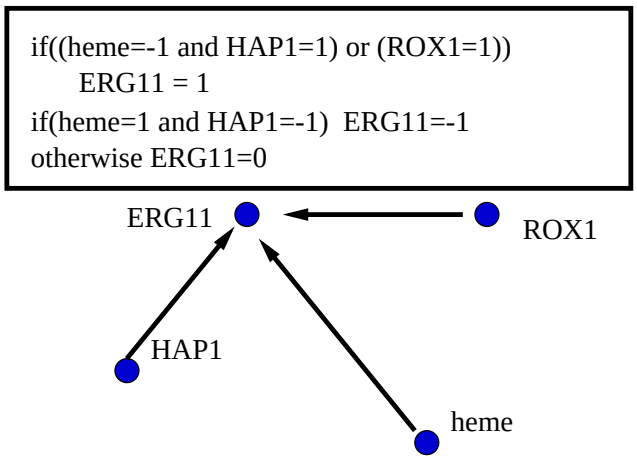

Figure 3.1: A fragment of a network describing the regulation of the yeast gene ERG11. ERG11, ROX1 and HAP1 are genes, while heme is a metabolite. A Boolean function describing the regulation of ERG11 is shown in the top box. Alternatively, it could be encoded as a table, associating an output value to each combination of values of the input variables. The bottom diagram shows the corresponding fragment of the dependency graph.

richer structure of the functional model space when developing computational devices to assess the quality of a dependency structure (when analyzing a dependency structure we actually view it as an equivalence class of detailed models with an identical digraph).

As remarked above, our model space is discrete, deterministic and synchronous. We have selected this modeling approach in order to minimize the number of degrees of freedom and to avoid over-fitting. We believe that important features of biological systems (mainly complex genetic/protein switches) can be elucidated using such simplified models. Continuous or stochastic modeling would require rate constants tuning, which drastically increase the amount of information needed to validate a model, and thus is currently impractical for all but very small networks. The discrete nature of our models enables the use of discrete algorithms and combinatorial analysis techniques. This is another important advantage of using discrete entities.

The definitions of network models and model spaces provide the platform for searching high quality models of a given biological system. This search is driven by the availability of significant amounts of experimental information characterizing the system behavior over different conditions. We next define our viewpoint on micro-array data, thus providing a formal definition for the most important biological data source today.

Definition 3 *An* **experiment** *is a triplet* $(INP, OUT, PERT)$ *where* INP *and* OUT *are the* **input and output vectors**, *assigning values from* C *to each variable in* U, *and* $PERT \subseteq U$ *is the set of* **perturbed variables**.

$PERT$ indicates those variable whose logic was perturbed in the experiment, e.g. by knock-out or over-expression of a gene. If in a particular experiment $v \in PERT$, then we cannot draw conclusions on the regulation of v based on that experiment, since its value was fixed, irrespective of the values of its inputs. We can, however, use that experiment to infer possible regulatory effects v may have on other variables. Typically, a knock-out experiment will produce one triplet. Time-series data, providing expression levels at a series of n time points, yield $n-1$ experiment triplets, where the vectors at time points i and $i+1$ form INP and OUT of the i-th experiment. Note that this transformation assumes that data dependence is Markovian. We will use INP_S^e (OUT_S^e) to denote the input (output) values of the variable set S in the experiment e.

Some compensation for the modeling limitations caused by discretization of the network is provided by using a probabilistic approach to model the data. This enables better data utilization by factoring in the noise inherent in high throughput experiments. Below, and occasionally later, we use overlines on vectors for clarity. Formally:

Definition 4 *A* **(noisy) experiment** *is a triplet* $(\overline{PINP}, \overline{POUT}, PERT)$*, where* $PERT \subseteq U$*, and* $\overline{PINP}$ *and* $\overline{POUT}$ *assign to each variable in* U *a distribution over the values in* C*. In other words,* $PINP_v(c)$ *(*$POUT_v(c)$*) is the probability that* v *attains the value* $c \in C$ *in the input (output).*

Definition 5 *A* **steady state experiment** *is an experiment in which* $INP = OUT$*.*

The distinction of steady state experiments is an important one. Real life data sets are often either time series of samples along some synchronized biological process [46, 20], or a single sample from a cell culture under some condition [32]. Steady state experiments might contain an averaging of an underlying temporal process and so modeling them correctly entails a less detailed representation of the biological system. Mathematically, for steady state data, one must exclude models with variables regulating themselves, in order to avoid the trivial self-regulation solution.

We can now tie models and experiments together using a scoring function, and provide the key elements in the expansion process: a core and its expansion.

Definition 6 *A* **fitness function** *is a function assigning a real value to each network in the model space, based on experiments data. A* **model** *(or network)* **core** *is a network defined on a subset* $U' \subseteq U$*. A* **core digraph** *is a dependency graph defined on* $U' \subseteq U$*. An* **expansion** *of a core digraph* G' *is a digraph* G *containing* G' *as a subgraph.*

In this terminology, the ultimate network reconstruction problem is to find a network of maximum fitness in the model space. We focus here on a much more modest task: finding an

expansion of the core by a few nodes and/or edges so that fitness is maximized. See Figure 1.1 for an illustration of an expansion.

3.2 Evaluation of Fitness Functions

Our goal is to computationally infer biological pathways by finding an expansion network or digraph that fit the experimental data best. The core graph would represent existing biological knowledge and should enable us to achieve improved specificity (compared to the quality of de-novo reconstruction). This must be preceded by developing a good fitness function. Such function should perform well both in "sensitivity" (scoring good expansions high) and "specificity" (scoring bad expansions low), and must also be efficiently computable. *Local* fitness functions evaluate the fit of the experimental data to the function f^v of a single variable v, while *global* fitness evaluates the overall network. The following definition serves as our starting point in developing local fitness functions.

Definition 7 *Given a function $\phi \in F_{bio}$ and a set of experiments $E = (INP^e, OUT^e, PERT^e)_{e \leq n}$, the* **consistency** *of ϕ for variable v, or the consistency of the pair (ϕ, v), is :*

$$Consist(v, \phi, E) = |\{e \in E, v \notin PERT_e \mid \phi(INP^e) = OUT^e_v\}| \tag{3.1}$$

Denote the arguments of ϕ by $x_1, \ldots, x_d$; the **consistency** *of (ϕ, v) given a noisy experiment set E is:*

$$Consist(v, \phi, E) = Pr(\phi(x_1, \ldots, x_d) = v) = \sum_{e \in E} \sum_{u=(u_1,\ldots,u_k) \in C^k} (\prod_i PINP^e_{x_i}(u_i)) \times POUT^e_v(\phi(u))$$

See Figure 3.2 for an illustration of consistent and inconsistent experiment set. The explicit formula assumes statistical independence of the distributions $PINP$ and $POUT$.

In a similar fashion we define the consistency of a set S of arguments for node v as the maximum consistency obtained by any $f^v \in F_{bio}$ whose arguments all belong to S:

Definition 8 *Let $S \subseteq U$ and $v \in U$. The consistency of (S, v) based on a set E of experiments is:*

$$Consist(v, S, E) = max\{Consist(v, f^v, E) \mid f^v \in F_{bio} \, and \, arg(f^v) \subseteq S\}$$

We say that the consistency is **perfect** when its value is exactly n. In the unconstrained case ($F_{bio} = \mathcal{F}$) this means that for each combination of arguments values, the output value is unique. Indeed, when there are no constraints on functions in the model space we can compute the consistency of a candidate argument set for a node efficiently, as follows:

Proposition 9 *For any $S = \{s_1, \ldots, s_d\} \in U$, if $F_{bio} = \mathcal{F}$ and E is a set of noiseless experiments we have:*

$$Consist(v, S, E) = \sum_{c_1,\ldots,c_d \in C} max_{c \in C} |\{e \in E \mid INP^e_{s_i} = c_i, i = 1, \ldots, d \wedge OUT^e_v = c\}|$$

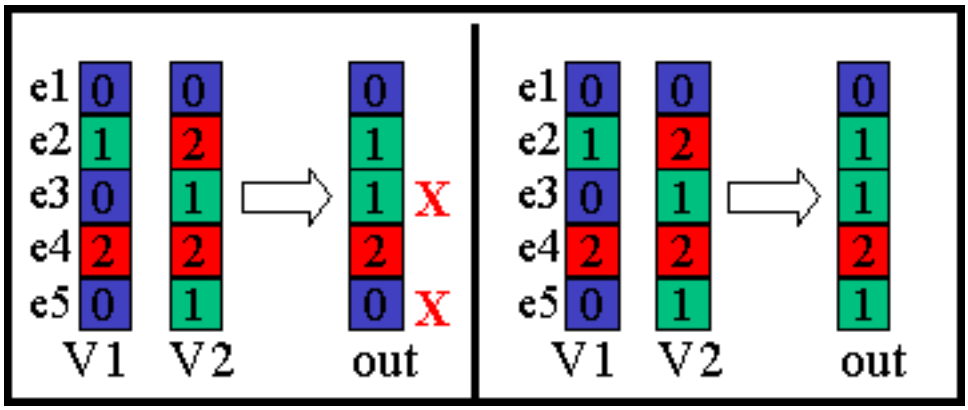

Figure 3.2: The concept of consistency. $S = \{v_1, v_2\}$ is the set of arguments of the variable *out*. In the left figure, experiments e_5 and e_3 have the same input values but a different output value. This contradicts the assumption that S directly regulate *out*. The right figure illustrates the perfectly consistent case.

Proof: Since we have no constraints on the function once its set of arguments is determined, we can optimize the consistency by making the best choice for each input assignment independently. ■

A similar reasoning applies to noisy experiments, by maximizing the likelihood of the function value for each input assignment independently. Figure 2 outlines the algorithm for noisy experiments. The algorithm uses a table of dimensions $|C|^d \times |C|$ and iterates on the experiments to simultaneously sum the probabilities of all i/o transitions. Let $S = \{v_1, \ldots, v_d\}$, denote the number of experiments by n and let $m = |C|$.

Consist(v, S, E):
Initialize a $m^d \times m$ real valued table *vote* and
a scalar *consist* with zeroes.
For all $e \in E$ if $v \notin PERT^e$ do
    For all vectors $u \in C^d$ do
        $p_u = \prod_i INP^e_{v_i}(u_i)$
        For $i = 1, \ldots, m$ do:
            $vote[u, i] = vote[u, i] + p_u * POUT^e_v(c_i)$
For all $u \in C^d$ do:
    $consist = consist + max_i\{vote(u, i)\}$.

Figure 3.3: Consistency computation.

Proposition 10 *The consistency of an argument set for a variable in the model space* $(U, C, \mathcal{F}, INDEG_d)$ *can be computed in* $O(nm^{d+1})$ *steps for noisy experiments and* $O(nd + m^{d+1})$ *steps for perfect experiments.*

Proof: For both cases we must iterate over all experiments. In the perfect experiments case we spend $O(d)$ time per experiment reading the arguments and updating the table and conclude with exhaustive iteration over a m^{d+1} table to collect the maximum in each row. In the noisy case we spend $O(m^{d+1})$ time for each experiment (filling the table with each value combination), and again $O(m^{d+1})$ time in the post process. ■

One should note that other methods for evaluating the fitness of a dependency structure can be used (for example, mutual information or Bayesian BDe/MDL). Consistency has the advantage of being easy to compute and theoretically simpler. It also provide a convenient basis for the development of the more sophisticated scoring functions below.

The main drawback of the consistency function is that it does not give information regarding the specificity of a speculated regulation pattern, and is thus very sensitive to over fitting. For example, if every combination of values of the arguments appears at most once, a function with perfect fitness can be formed, but it will have very pour confidence. To address this problem, we shall describe how to calculate a probability, or as it is often called in the life sciences, "p-value" of the observed consistency (to be denoted as the regulation specificity function $rSpec$). As our null hypothesis, we assume independence of the measured values of the variable v and the variables in $arg(v)$. Having calculated the consistency of $(arg(v), v)$ we wish to estimate the probability of observing such level of consistency or higher in the data under the null hypothesis. Consider first the case of n perfect experiments and assume v was not perturbed in any experiment (if it was, simply remove these experiments from the data set when considering fitness of v). Now define a probability space based on the data. We use two random variables, X and Y. X attains values in C with probabilities :

$$p_i = Pr(X = c_i) = \frac{1}{n}|\{e \in E \mid OUT_v^e = c_i\}| \tag{3.2}$$

If $arg(v) = \{v_1, \ldots, v_d\}$, Y is taking values in C^d with probabilities :

$$Pr(Y = (c_1, .., c_d)) = \frac{1}{n}|\{e \in E \mid INP_{v_i}^e = c_i, i = 1, \ldots, d\}| \tag{3.3}$$

Definition 11 *Let* $S \subset U$ *and* $v \in U$. *If* (S, v) *has consistency* k *based on a set* E *of noiseless experiments, then the* **regulation specificity** *of* (S, v) *given* E, *denoted* $rSpec(S, v, E)$, *is the probability of obtaining a consistency of* k *or higher in the probability space* $(Y \times X)^n$ *(*n *samples from* $Y \times X$, *each interpreted as an experiment with a value from* Y *as the input and a value from* X *as the output).*

The size of the probability space defined above is exponential in the number of experiments, so a naive algorithm for computing $rSpec$ is not practical. We will present an approximation that is practical for the case where $n-k$ is $O(1)$ (almost perfect consistency) and is linear in the number of experiments. We use a random variable from the space X defined above and set the input values deterministically to $(INP_S^e)_{e\leq n}$. We now calculate the probability $\pi(k)$ for obtaining a consistency k or better in a data set with the n inputs from INP_S and outputs sampled from X. If there are l input configurations with multiplicities $n_1,\ldots,n_l$ then $\pi(k)$ is the probability of getting n_i' identical values out of n_i samples from X for $i=1,\ldots,l$ where $\sum n_i' \geq k$.

Denote by $\psi(r,s)$ the probability of getting exactly s identical values when sampling r times from X. Then $\psi(r,s)$ can be computed exhaustively in $O(rm^r)$ time. To compute $\psi(r,s)$ efficiently we use the following lemma:

Lemma 12 *$\psi(r,s)$ can be computed in $O(m+rm^{2(r-s)})$ time.*

Proof: distinguish two cases:

(1) if $s > \frac{r}{2}$ then $\psi(r,s) = \sum_j \binom{r}{s} p_j^s (1-p_j)^{r-s}$. This is true since we are sure that we do not count any assignment twice (in more than half of the cases the value is identical). Therefore, in that case ψ is computable in $O(m)$ time.

(2) if $s \leq \frac{r}{2}$ then $r \geq 2s$ or $2r-2s \geq r$, so $rm^r = O(rm^{2(r-s)})$. ■

So in particular, when $r-s=O(1)$ computing $\psi(r,s)$ is polynomial in r and m for all s.

Lemma 13 *$\pi(k)$ is computable in $O((l+1)^t(lm+nm^2t))$ time*

Proof: To compute $\pi(k)$ we enumerate all integer partitions of t, $t_1,\ldots,t_l$ s.t. $\sum t_i \leq t$, and compute :

$$\pi(k) = \sum_{(t_i)} \prod_i \psi(n_i, n_i - t_i) \tag{3.4}$$

The number of partitions is bounded by $(l+1)^t$. By the previous lemma, each term in the product $\psi(n_i, n_i-t_i)$ can be computed in $O(m+n_i m^{2t_i})$ time. Hence, $\pi(k)$ can be computed in $O((l+1)^t(lm+nm^{2t}))$. ■

Corollary 14 *If (S,v) has consistency k and $k=n-O(1)$ then $rSpec$ can be approximately evaluated in polynomial time.*

The main point here is that for fixed alphabet, we can approximate $rSpec$ in linear time in the number of experiments, whenever the consistency is almost perfect.

The generalization of regulation specificity to noisy experiments is done as follows. Given a noisy experiment $e = (PINP, POUT)$ we assume statistical independence of the distributions $PINP$ and $POUT$ and construct a probability space that represents possible deterministic instantiations of e. The probability of obtaining a given perfect experiment value $h = (INP, OUT)$ is :

$$Pr(h) = \prod_{v \in U} PINP_v(INP_v))POUT_v(OUT_v)) \tag{3.5}$$

The overall probability of a set of noisy experiments $h = (h_1, \ldots, h_n)$, assuming statistical independence of the experiments is $Pr(h) = \prod_i Pr(h_i)$. Note that the latter independence assumption would not hold for time series experiments.

Definition 15 *For a set of noisy experiments E and the above setting:*

$$rSpec(v, S, E) = E(rSpec(v, S, E_r)) \tag{3.6}$$

A naive computational approach for the finding the expectations above is not practical here, since the size of the probability space is exponential in the number of experiments. We have performed approximate evaluations by exhausting only part of the probability space for E_r. This was done by sampling from a set of artificial perfect experiments in which almost all of the variables have their most frequent value in $PINP_v$.

Before moving on to define global fitness, we shall review a standard way of scoring dependency structure using mutual information. The mutual information of two random variables X and Y [17] is defined as $M(X, Y) = H(X) + H(Y) - H(X, Y)$ where H is the entropy function defined for discrete distribution as $H(X) = \sum_i -Pr(X = i)log(Pr(X = i))$ (summing over the distribution alphabet). The entropy of X, Y is the entropy of their joint distribution. Given an experiment set E, we can calculate the input (output) distribution of a variable set $S = \{v_1, \ldots, v_d\}$ as $Pr((c_1, \ldots, c_d))$ = fraction of experiments in E for which the inputs (outputs) of S were exactly $\{c_1, \ldots, c_d\}$. Denote this distribution by X_S^E. A variable set thus induces a distribution over a probability space of m^d elements and we can calculate the entropy $H(X_S^E)$. We can now define :

Definition 16 *The* **mutual information** *of a regulating set S, a variable v and an experiment set E is defined as :*

$$MInfo(S, v, E) = M(X_S^E, X_{\{v\}}^E) \tag{3.7}$$

Note that mutual information assumes by definition a constraint-less model space ($F_{bio} = \mathcal{F}$), as its definition uses only the dependency structure. This may limit the incorporation of biologically motivated constraints.

The computational tools we have developed so far define **local** evaluators, i.e. estimators for the modeling quality of a single model variable given a speculated regulation formula or dependency structure. We can extend this to evaluation of a core expansion in more than one way. The most natural and immediate of these would be simple averaging.

Definition 17 *Given a core dependency graph G' and an expansion $G'' = (U'', E'')$, define the* **fitness** *of the expansion G'' by:*

$$fit(G'') := \frac{1}{\|U'\|} \sum_{v \in U'} rSpec(v, S_v, E) \tag{3.8}$$

where $S_v = \{x \in G'' | xv \in E''\}$.

3.3 The Pathway Expansion Problem

Having created the framework for manipulating expansions and their fitness, we analyze in this section the computational implications of some of the possible forms of the optimization problem associated with the expansion procedure.

Definition 18 *The* **pathway expansion problem** *is defined with respect to a model space* (U, C, F_{bio}, G_{bio}) *and using a prescribed fitness function* fit*. Given a set of experiments* E *and a core digraph* $G' = (U', E')$*, find a core expansion* $G'' \supseteq G'$ *maximizing* $fit(G'')$*. If several solutions exists, find one minimizing* $\|G''\|$*.*

For an expansion $G'' = (U'', E'')$ and $v \in U'$, set $S_v = \{x \in G'' | xv \in E''\}$ and define

$$fit(G'') = \sum_{v \in U'} consist(v, S_v, E) \tag{3.9}$$

3.3.1 Complexity

Proposition 19 *The pathway expansion problem, with the fitness function (3.9), is NP hard, even assuming constant time computation of fitness, and even for cores of size one.*

Proof: We shall show that the decision version of the problem, "is there an expansion with perfect consistency and size $\leq l$?" is NP-complete. Clearly that problem is in NP. We will construct a reduction from SET COVER which is known to be NP-complete [27]. The inputs to the latter problem are set $S = \{a_1, \ldots a_r\}$, a collection of subsets $I = \{S_1, \ldots, S_q\}$ of S and an integer l. The question is whether there exist a subset $I' \subset I$, $|I| \leq l$ such that $\cup_{i \in I'} S_i = S$. We construct an instance of the expansion problem as follows. U will be the set of subsets plus an additional variable, i.e., $U = \{1, \ldots, q, "c"\}$. The experiments set will consist of $n + 1$ steady state experiments indexed by $S \cup "0"$ and defined by the matrix below (columns are variables, rows are experiments, χ is the standard subset characteristic function):

	c	1	..	q
0	0	0	..	0
1	1	$\chi_{S_1}(a_1)$	$\ldots$	$\chi_{S_q}(a_1)$
$\vdots$	$\vdots$			
r	1	$\chi_{S_1}(a_r)$	$\ldots$	$\chi_{S_q}(a_r)$

(3.10)

The core is set simply to the single variable c and we set $l = k + 1$.

We will show that an expansion of c with perfect c consistency is equivalent to a set cover. First note that any set of expansion variables corresponds to a collection of subsets $I' \subset I$. Now if c is perfectly consistent with I' then there do not exist e_1, e_2 such that

$\overline{INP^{e_1}_{I'}} = \overline{INP^{e_2}_{I'}}$ and $OUT^{e_1}_c \neq OUT^{e_2}_c$. Taking e_2 as the "0" experiment implies that there is no e s.t. $\overline{INP^{e}_{I'}} = \overline{INP^{0}_{I'}}$. Since $\overline{INP^{0}_{I'}}$ is a vector of zeros, we conclude that for each $e \in E - \{0\}$ (equivalent to an S element) we must have a variable in I' (equivalent to a subset in the cover) with non zero value (equivalent to having a subset covering the element). In other words, I' is a set cover.

Now assume I' is a set cover. Taking the set $I' \cup \{c\}$ as an expansion yields perfect consistency since the only experiment with 0 values over all the arguments in I' is the "0" experiment (otherwise the node represented by the experiment is not covered).

In conclusion, there exist a set cover I' with $\|I'\| \leq k$ iff there exist an expansion $U'' = I' \cup \{c\}$ s.t. $\|U''\| \leq k+1$. ■

In the case of bounded in-degree (e.g. if $G_{bio} = INDEG_r$), the pathway expansion problem is polynomial for bounded core sizes. To see this note that if each variable in the core can have at most d inputs then the largest effective expansion cannot have more than $d * |U'|$ variables. If fitness calculation is polynomial (as consistency is), we can exhaust all $\binom{|U|}{d|U'|}$ subsets of U and take the smallest among those with optimal fitness. Note the same argument holds for any fitness function which is polynomially computable but that we still have exponential dependency in the size of the core.

Another interesting model space is $G_{bio} = MAXREG_r$, i.e. all graphs with at most r nodes of positive out degree. Biological studies show that the number of control genes is often small fraction of the entire genome. It is thus desirable to add this constraint or goal to the reconstruction environment.

Definition 20 *The* **minimum regulation reconstruction problem** *is defined with respect to a model space (U, C, F_{bio}, G_{bio}) using a prescribed fitness function fit and a parameter r. Given a set of experiments E we wish to determine if the set of optimal digraphs on U (i.e. the set for which $fit(G)$ is optimal) contains a graph for which at most r nodes have non zero out degree.*

Proposition 21 *The minimum regulation reconstruction problem, with the fitness function (3.9), is NP complete.*

Proof: We will construct a reduction from TOTAL DOMINATION SET [13]. In this problem, the input is a directed graph $G = (V, E)$ and an integer k. The question is whether there exists a subset S of at most k vertices such that every vertex in G has an incoming edge from a vertex in S. Note that in this variant each vertex in G must have an incoming edge from a vertex in S and a vertex cannot dominate itself. The idea now is to tailor an experiment set that will force any consistent regulation digraph to include for each variable direct dependency on at least one of its neighbors.

Given a digraph $G = (V, E)$, we let our variable universe U equal V and work over the binary states. Denote $N_i = \{v \in V | (v, i) \in E\}$ and $M_i = \{u \in V, u \neq i | \exists v \in V s.t. (v, i) \in E, (v, u) \in E\}$. Now generate the set of experiments E as follows. For each node i we introduce one experiment e_i with the following perturbations: All nodes in N_i are perturbed to 1 and all nods in M_i are perturbed to 0. The node i itself is left unperturbed. The OUT, INP function are now defined as:

$$INP(j) = OUT(j) = \begin{array}{ll} 1 & j \in N_{i,1} \cup \{i\} \\ 0 & otherwise \end{array}$$

We also add an unperturbed experiment *reset* for which $INP(i) = OUT(i) = 0$ for all i. Finally set $r = k$. The reduction is clearly polynomial. We claim that a set of nodes is a total domination set in the graph if and only if it can function as the set of regulators in a network that will be consistent under the experiment set E.

We first show that for each dependency graph which is perfectly consistent with E, the set of regulators is a total domination set. This is true since for each node i, the experiments *reset* and e_i can be consistent only if one of the nodes in N_i is an input of i (all other variables are set to 0 in this experiment).

For the other direction we shall show that each total domination set S can be used as a consistent set of regulators. We construct the dependency graph connecting the nodes of S to all their neighbors in G. We will show that no inconsistencies can happen.

Perfect consistency of a variable i is obtained whenever the input value combinations for all experiments in which i is unperturbed and $OUT(i) = 0$ are disjoint from those where $OUT(i) = 1$. Now note that the only experiment in which i is unperturbed and has $OUT(i) = 1$ is the experiment e_i. In this experiment, those nodes of $N_i \cap S$ (denote them by D_i) take input values of 1. All other experiments in which the set D_i has input values of 1 must include either perturbation of i to 1 (in e_k for which $(i, k) \in E$) or perturbation of i to 0 (in e_k for which some $(v, k) \in E$ and $(v, i) \in E$). This implies the input assignment of 1 to D_i is consistent. Since this is the only case in which $OUT(i) = 1$ and is unperturbed, we are done. ■

3.3.2 A Practical Approach

Core expansion is a very difficult problem when attacked in its fullness. Fortunately, very limited expansions (1-3 variables) are very informative already and so computation is not the current bottleneck in our methodology (experimental data is the bottleneck). We define the **single node expansion** problem as the pathway expansion problem where U'' is obtained from U' by adding a single node. The expansion process is thus an exhaustive pass over U calculating the global score of the core with the addition of a single variable each time. We

can now examine the top scoring variables as potential candidates for involvement in the core. We can also use the more detailed information of local (not global) gain in fitness for specific core nodes in top scoring expansion candidates, and improve our understanding on the potential function of the expansion candidate.

Chapter 4

Simulation Results

We have performed simulations to test our modeling assumptions and to verify our system performance in a laboratory environment. The studies we report on here are by no means complete. They only form the basis for deeper exploration of the relations between various factors of the expansion problem. The results elucidate some interesting, non trivial effects, enable us to compare the performance of different fitness functions and provide an insight on the effects of noise and core structure on the performance of our algorithms.

4.1 Simulation Setup

There are numerous ways to simulate biological networks, even when assuming discrete, synchronous models. Our selected simulation scheme, as described below, was designed to follow known biological systems whenever it is possible, keeping in mind that we are using the simulation mainly for a comparative study and that we are not trying to directly deduce real life performance from the simulated runs.

Our simulated models had ternary values. The network topology was a layered graph imitating the structure of master regulators/local regulators/functional genes known from biology. We randomly partitioned the variable universe into layers and then selected random edges such that variables in the upper layers will have larger expected out degree. All variables had the same in-degree distribution. Our randomization procedure allows for feedback loops as there can be edges from lower to upper layers. The topology mimics the tendency for large out degrees in the "master regulators" layers. The topology randomization algorithms is described in Figure 4.1 and an example is given in Figure 4.2.

The logic of our randomized network model was generated using a combination of boolean circuits of limited size. The idea was to limit the complexity of the logic we consider, again imitating known biological features. To create a ternary function on ternary inputs we have

RandomizeNetworkTopo(U, l, p, w, d_e, σ):
d_e, σ: The expectation and standard deviation for the normal distribution of in-degrees.
U : variable universe, l : number of layers
$p = (p_0, \ldots, p_{l-1})$: layer size distribution ($p_i \geq 0$, $\sum_i p_i = 1$).
$w = (w_0, \ldots, w_{l-1})$: layer outgoing edges distribution ($w_i \geq 0$, $\sum_i w_i = 1$).
Step 1 : Partition U into layers
For each $v \in U$ determine the layer of v by sampling from the distribution p.
Step 2 : Determine in-degrees
For each $v \in U$, the indegree of v, $indeg(v)$ is computed by sampling from $N(d_e, \sigma)$.
Step 3 : Determine edges
For each $v \in U$, repeat $indeg(v)$ times :
Pick a layer k according to the distribution w.
Repeat sampling a vertex u in layer k with uniform distribution
until (u, v) is not in E.
Add the edge (u, v) to E.

Figure 4.1: Network topology randomization process.

first randomized a layer of qualifier nodes transforming the ternary inputs to boolean values. We have not limited the number of qualifiers for a single variable so that a given input could have both negative and positive qualifiers. Using the qualifiers layer as inputs, we constructed 3 random boolean circuits with AND/OR nodes only, one for each output value. We also randomized an order on the ternary alphabet. The function output was then computed by iterating on the alphabet, calculating the outcome of each boolean circuit and choosing as output the value of the first circuit giving true. A default value was chosen if all three circuit values were false. An example of a function is given in Figure 4.3.

Note that we avoided completely random logic and used expressions which are biologically reasonable (an AND gate may correspond to a protein complex needed to invoke transcription or block it; an OR gate is interpreted as the combination of two alternative regulation pathways). Still our functions were not monotone since a given variable could have been used positively and negatively (see the example).

Having created random network model, we generated random data sets using perturbations and trajectory recording. A set of perturbations were selected at random. We applied only perturbations of one or two variables and interpreted them as knock-outs fixing the perturbed variable value to -1. For each perturbation time we recorded the system trajectory starting from the native state (0 for all variables) plus the perturbations (-1 for each perturbed variable). The trajectory is calculated by repeated computation of logic function

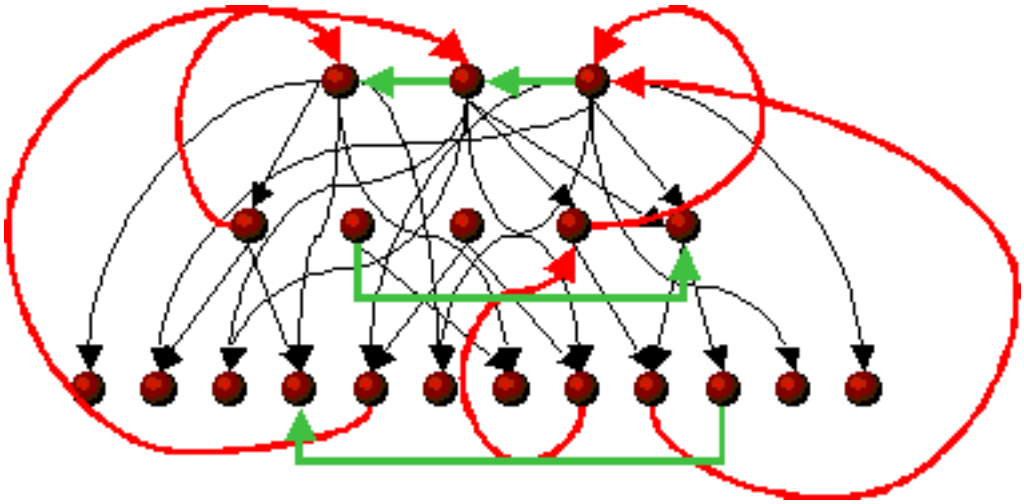

Figure 4.2: Example of the layered topology used in our simulations. Note that all variables have the same indegree distribution but upper layers have higher outdegrees. Note also we enable loops in our random topologies.

output at time t using as input the values at time $t-1$. Each time point was used as one experiment.

We applied measurement noise to our experiments using two parameters $p_{falseneg}$ and $p_{falsepos}$. $p_{falseneg}$ was used as the probability to changes of non zero variable values to 0. $p_{falseps}$ was used as the probability to change 0 values to a non zero value, the actual value is selected with uniform distribution on $C - \{0\}$.

To generate pathway cores, we selected a focus node at random and perform breadth first search from it in the undirected graph corresponding to the model topology. We used a parameter $p_{coremisses}$ to possibly ignore variables encountered in the search and so avoid the assumption of perfect knowledge on the core. The BFS was terminated after s_{core} variables were selected.

4.2 Results

Results of 840 different runs are summarized in the graphs below. The instances analyzed varied in their scoring function (consistency, regulation specificity or mutual information), their random core size, their core completeness factor and level of measurement noise. All experiments were performed with variable universe of size 500 and experiments dataset consisting of 250 different perturbations and 10 time points trajectory for each of them. A complete list of parameters is given in Table 4.1.

To assess the quality of the expansion process, we define the *specificity* of the expansion

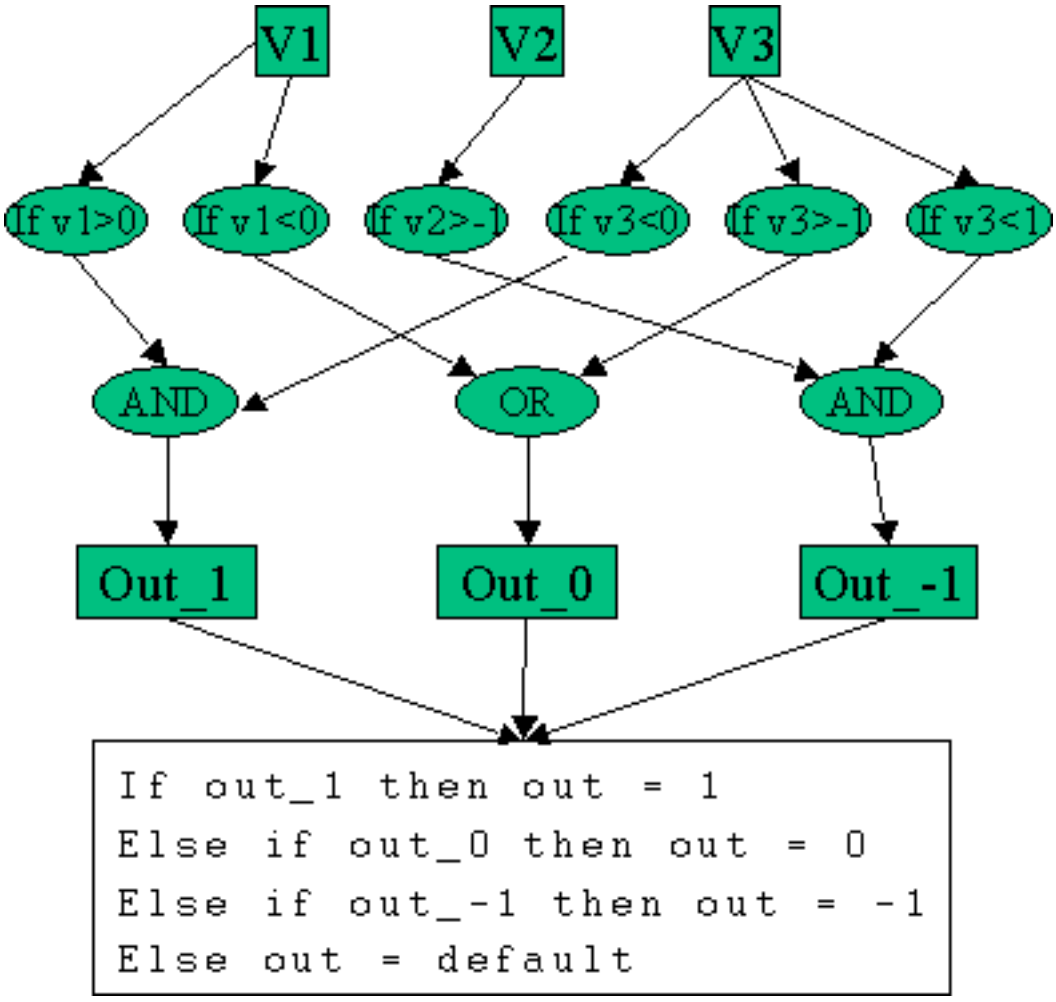

Figure 4.3: An example of limited complexity random discrete logic function. A qualifier layer is used as input for three different boolean circuits that are combined together to determine the function output.

process as the probability that a predicted expansion variable is at distance 1 from the core. For each set of parameters we repeatedly generated the network and experiments followed by single node expansion to score each of the variables in U vs. the core (the score, as defined in earlier chapters, reflects the gain in the modeling fitness of the core obtained by adding the variable to the core regulators). The k top scoring genes are tested to see if their graphic distance is 1 and the specificity of the expansion process is defined as the fraction of correct predictions (each selection of k gives a different specificity, larger k yield lower specificity). The results were averaged across 20 different networks and datasets for each parameter set.

The expansion running times for core size 10 were 8-11 minutes for regulation specificity and 5 minutes for mutual information/consistency scores. Times were measured on a Pentium

Parameter	Values	Remarks
Variable Universe size ($\|U\|$)	500	
Number of layers l	5	
Indegree distribution parameters	$d_e = 3$, $\sigma = 1$	Same for all layers
Layer size ratios (p)	(1:5:10:20:100)	
Layer outgoing edge distribution (w)	(1:1:1:1:1)	Note the relation to p
Core Size (s_{core})	10 or 5	
Fraction of core misses ($p_{coremisses}$)	0.1 or 0.5	
Single gene perturbations	150	
Two genes perturbations	100	Pairs of genes selected for single pert.
Trajectory steps	10	
False positive noise	0.01,0.02,0.05 or 0.1	
False negative noise	0.01,0.02,0.05 or 0.1	

Table 4.1: Parameters used in the reported simulations.

III 500MHz Linux machine (laptop version, 992.87 bogomips).

The overall picture outlined by the results is that expansion using reasonable number of experiments yield very high quality results (higher then 90% specificity with conservative thresholds). One should, however, avoid drawing direct conclusions from these numbers to real life biological systems, and use them only in comparison to other methods applied to such artificial models.

Using the results we can compare the performance of various scoring schemes (Figure 4.4,4.5,4.6,4.7). In all cases, it is evident that *rSpec* performs consistently better than the other two methods. This is an important validation of the theoretical development of *rSpec* and serves as the basis for further development of high specificity scoring functions.

The effect of increased noise levels was studied in a series of parameters settings false positive and false negative noise factors (Figure 4.8). As expected, high noise levels resulted in poorer specificity, but the process functions reasonably even when noise levels are rather high. The effect of false negative (nullifying non default readings) was almost unnoticed, while the effect of false positive (making default readings non zero) was more severe when noise levels were 0.05. One should note that the noise was applied on a global basis even when the whole perturbation resulted in very local value changes, so the false positive levels we have experimented with represent high noise levels (for an average variable, only 5% of the perturbations yield value changes so false positive of over 5% means there is more noise than signal in the non-default readings).

The effect of core size and completeness follows the expected pattern (compare Figure

4.4 and Figure 4.5). Larger cores are generally more reliably expanded, as the scoring is based on more variables. When only the first one or two best variable are considered for expansion, scoring is better for cores with more missing variables since there is a good chance of identifying a missing variable that is tightly connected to the core. When more variables are considered, dense cores are better, as there is better chance to find a new good variable that is adjacent to a more compact set of variables.

The simulation environment available from GENESYS provides a convenient way for continued exploration of hypotheses and novel scoring/expansion methods. We intend to use it in our future study of the network expansion and reconstruction problems.

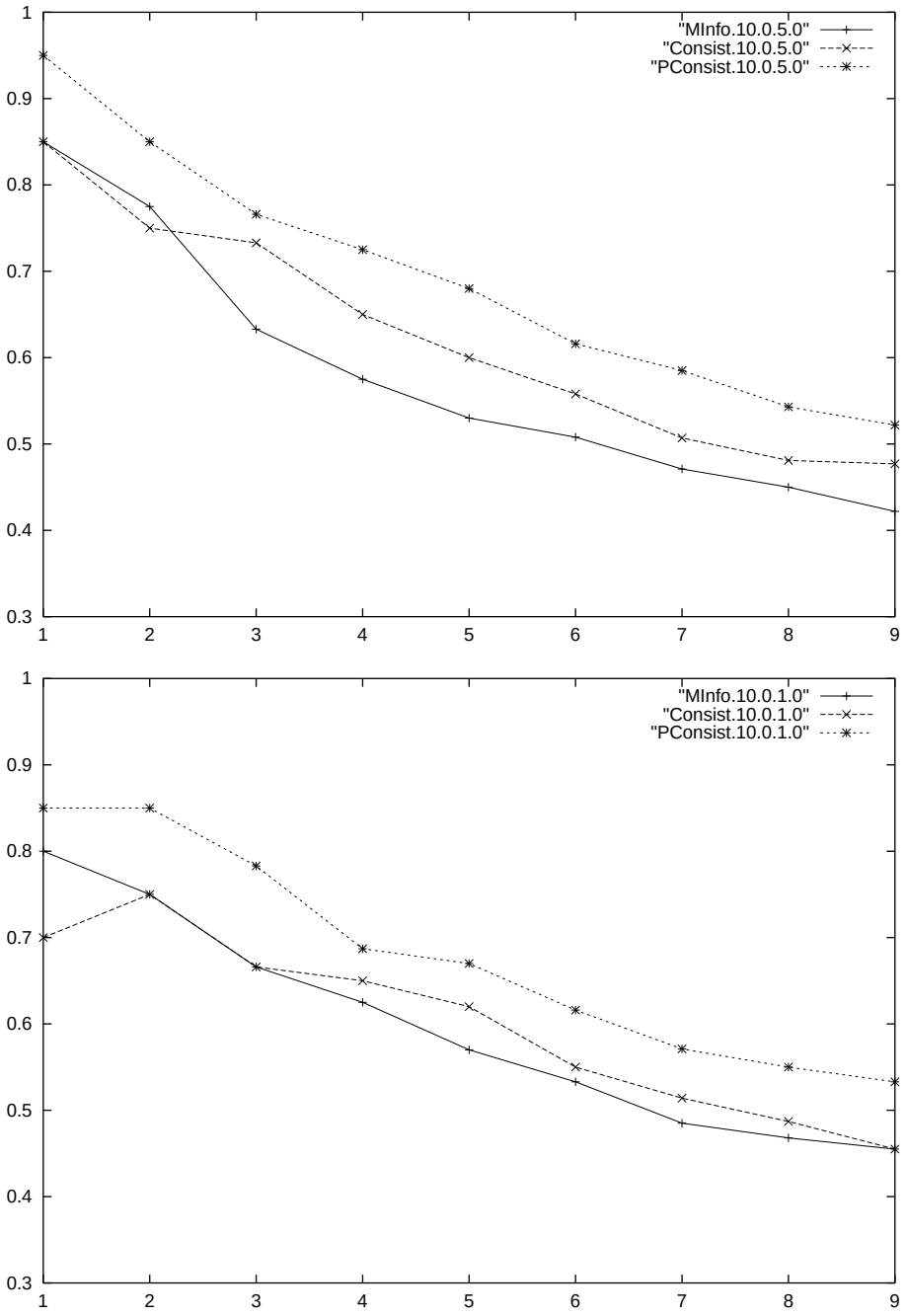

Figure 4.4: Specificity in noiseless experiments with core size 10, $p_{coremisses} = 0.5$ (upper figure) and 0.1 (lower figure). x axis: the k parameter (number of top scoring genes used), y axis: Specificity. The '+', 'x' and '*' correspond to mutual information, consistency and regulation specificity scoring methods, respectively.

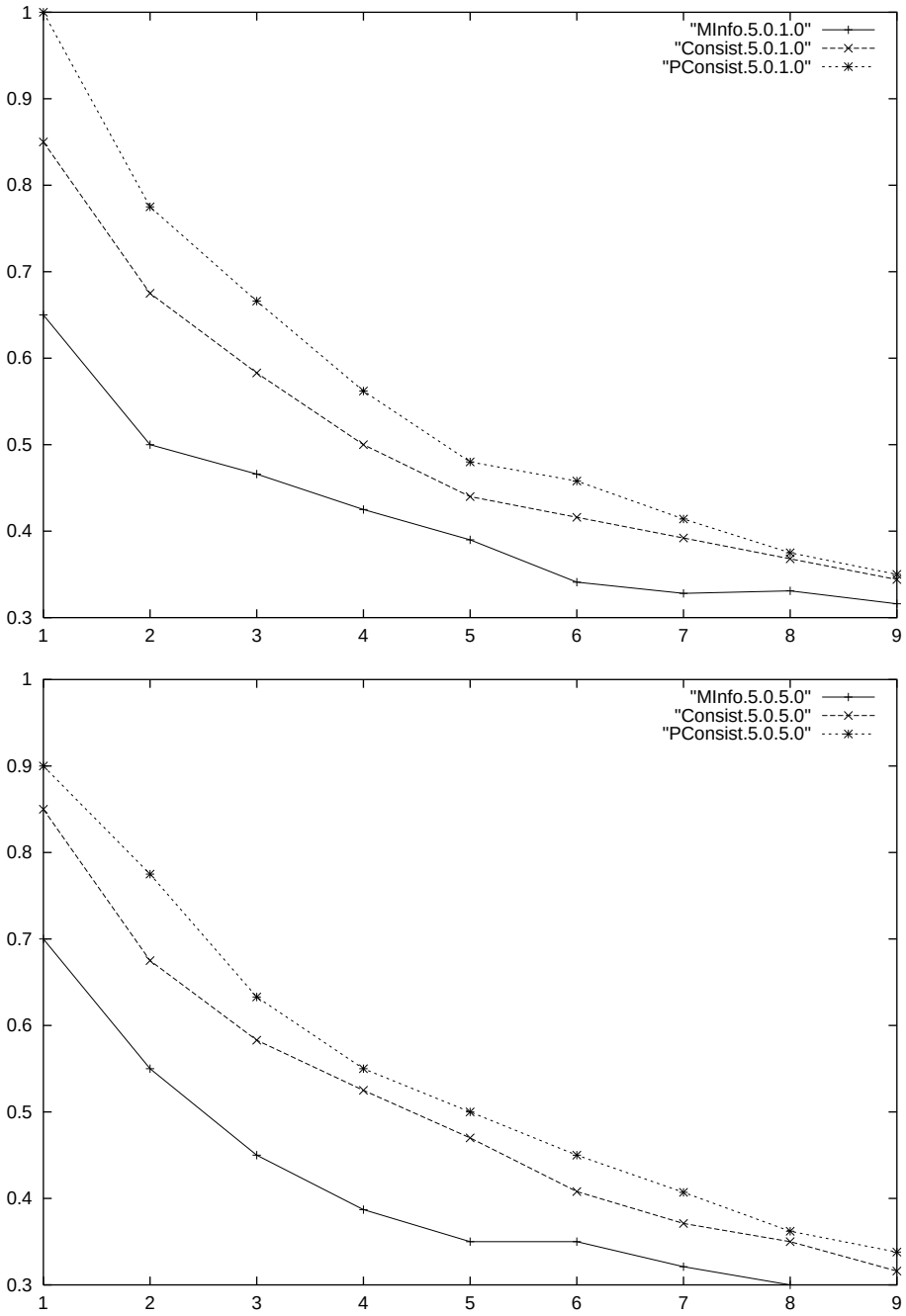

Figure 4.5: Specificity in noiseless experiments with core size 5, $p_{coremisses} = 0.5$ (upper figure) and 0.1 (lower figure). x axis: the k parameter (number of top scoring genes used), y axis: Specificity. The '+', 'x' and '*' correspond to mutual information, consistency and regulation specificity scoring methods, respectively.

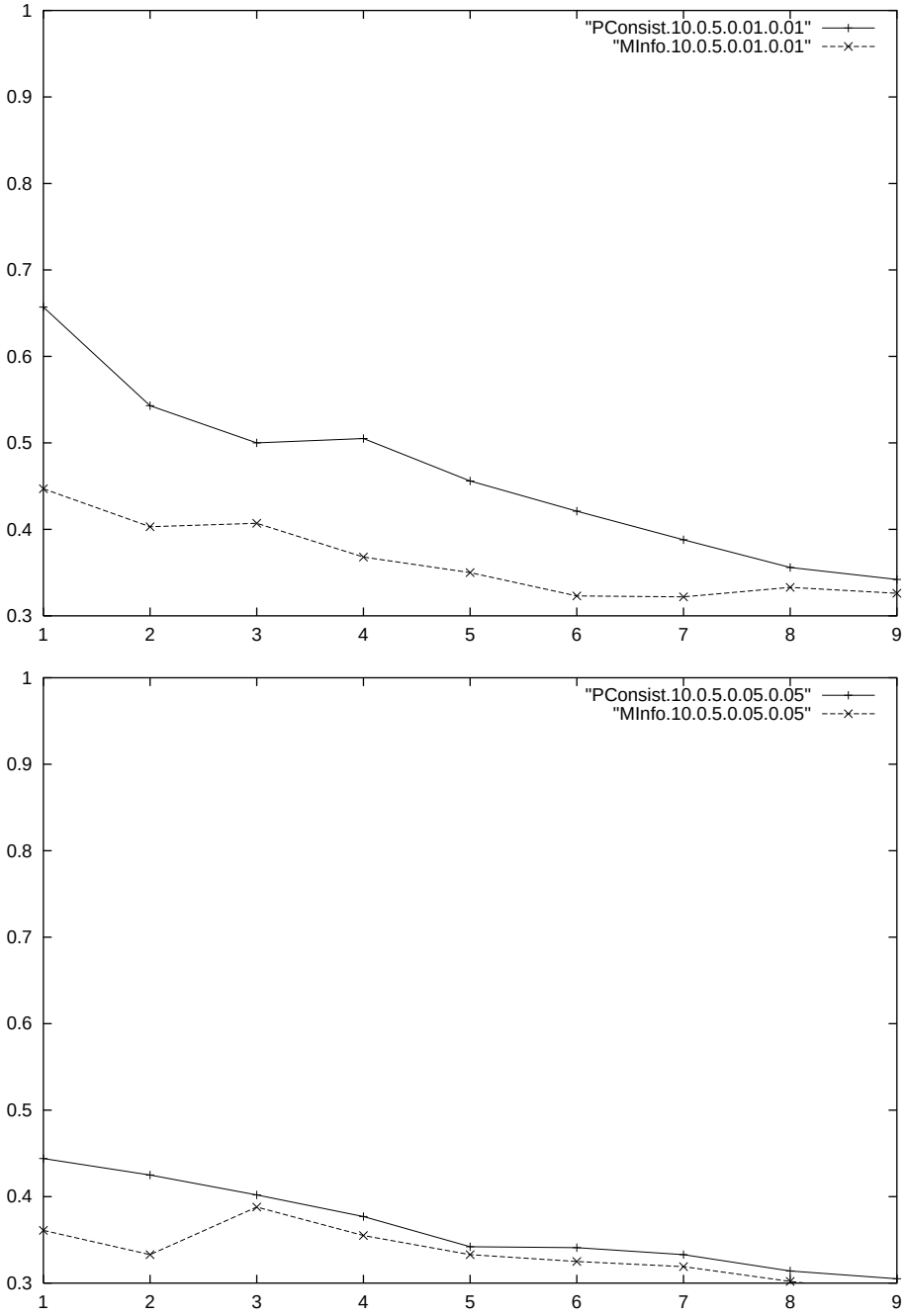

Figure 4.6: Specificity in symmetric noise experiments. Upper figure: $p_{falsepos} = 0.01, p_{falseneg} = 0.01$. Lower figure: $p_{falsepos} = 0.02, p_{falseneg} = 0.02$. Core size = 10 and $p_{coremisses} = 0.5$. x axis: the k parameter (number of top scoring genes used), y axis: Specificity. The 'x', and '+' correspond to mutual information and regulation specificity scoring methods, respectively.

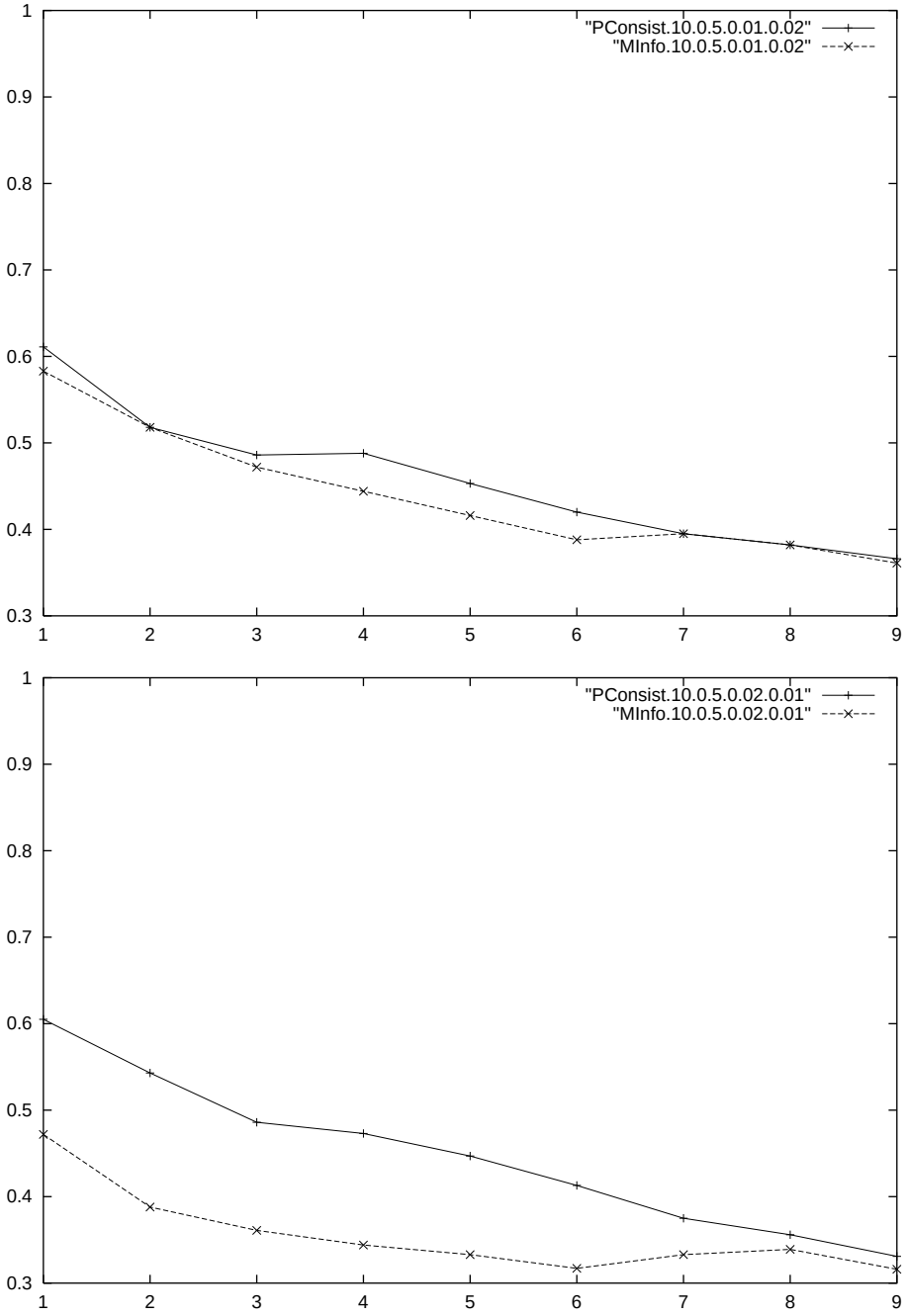

Figure 4.7: Specificity in non symmetric noise experiments. Upper figure: $p_{falsepos} = 0.01, p_{falseneg} = 0.05$. Lower figure: $p_{falsepos} = 0.05, p_{falseneg} = 0.01$. Core size = 10 and $p_{coremisses} = 0.5$. x axis: the n parameter (number of top scoring genes used), y axis: Specificity. The 'x', and '+' correspond to mutual information and regulation specificity scoring methods, respectively.

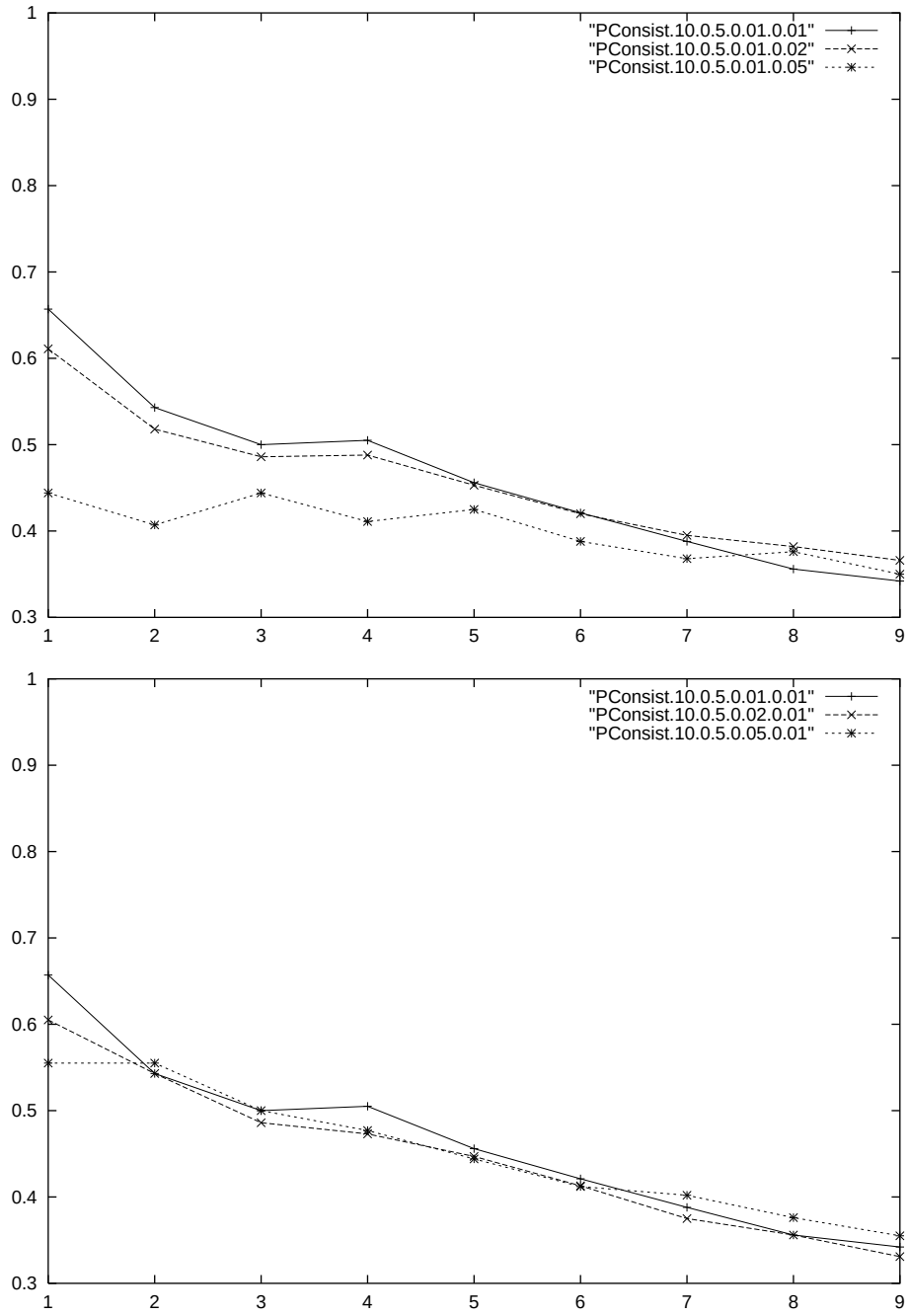

Figure 4.8: Algorithm behavior under variable noise levels. All data from experiments with core size 10, $p_{coremisses} = 0.5$ and $rSpec$ scoring function. Upper figure represents three different false positive rates with fixed false negative level of $p_{falseneg} = 0.01$. Lower figure represents three different false negative rates under fixed false positive level of $p_{falsepos} = 0.01$. x axis: the k parameter (number of top scoring genes used). y axis: Specificity. The '+','x' and '*' correspond to noise levels of 0.01, 0.02 and 0.05, respectively. The $rSpec$ score was used in all experiments

Chapter 5

Results On Biological Datasets

To test our ideas in a real world setting, we applied GENESYS to yeast transcription datasets using the ergosterol pathway as a core. We focused on the simplest possible core expansion: The **single node expansion** process (see section 4) examines each of the variables in U and calculates the sum of fitness gains to all core variables from adding that variable to the core. Note that unlike clustering or similarity tests, we are not looking for genes that are similar across the entire data set, but rather seek genes that might regulate or indirectly affect the pathway in those experiments which are left unexplained by the core model. We present below the results of two different screening processes, with different limitations and goals.

The fitness function was computed as follows. Denote the core by U'. For each non-core variable v, its global fitness is

$$\sum_{u \in U'} \max_{S \subset U \cup \{v\}, |S| \leq d} -rSpec(u, S, E).$$

5.1 Preprocessing Expression Data

Our study focused on yeast, which has the largest publicly available gene expression datasets. The variable set consisted of 6200 yeast ORFs. Transcription profiles were taken from two large scale yeast cDNA arrays experiments: Hughes et al. [32] performed some 260 selected knock-out experiments; Gasch et al. [28] performed 100 experiments testing yeast behavior in stressful conditions. We have chosen to view all experiments as steady state experiments: For knockout experiments this is a natural choice. For the stress time series data, we chose to view each measured transcription profile as a different steady state experiment, since the time intervals between measurements were non uniform and typically much larger than the transcription activation delay.

Each experiment was normalized by computing $\log(X_i)/\log(R_i)$ where X_i and R_i are the intensity levels for gene i in the specific condition of the experiment and in a common

reference tissue, respectively. By comparison with the same ratio measured in the wild type, a decision was made whether the gene was up-regulated, down regulated or unchanged.

Transforming the data to our noisy experiments representation was done as follows. The data in [32] contained detailed, per-gene noise model, assigning p-values for the statistical significance of up or down regulation based on a large number of repeated wild type experiments. We performed a conservative transformation of these values to distributions over the variables $\{-1, 0, 1\}$ as follows (compare table 5.1) : For genes designated in [32] as up (resp., down) regulated with p-value below 0.005, we assigned $POUT(1) = 1$ (resp., $POUT(-1) = 1$). Genes designated as up (down) regulated with p-value in the range 0.005 to 0.01 were assigned $POUT(1) = .7$ ($POUT(-1) = .7$) and $POUT(0) = .3$. All other designations were assigned $POUT(0) = 1$. For the data of [28], genes with increase factor 2.5 (resp., 2) were

Rosetta designations		GENESYS designations		
regulation	P-value range	POUT(-1)	POUT(0)	POUT(1)
down	[0,0.005)	1	0	0
down	[0.005,0.01)	0.7	0.3	0
up	[0,0.005)	0	0	1
up	[0.005,0.01)	0	0.3	0.7
otherwise		0	1	0

Table 5.1: Ranges table for the transformation of Hughes et al. p-values to distributions.

assigned $POUT(1) = 1$ (resp., $POUT(1) = .7$ and $POUT(0) = .3$). Down regulation with the same factors were assigned symmetrically, and all changes less than twofold were assigned $POUT(0) = 1$. These assignments are very conservative, trying to avoid false positive regulation assignments and reduce noise. As we demonstrate below, with the quantities of data available today, the noise introduced by the entire process can be tolerated.

Pathway cores were generated based on available literature, notably SGD [7] and YPD [16]. Other biological references will be cited when discussing the results.

Putative interaction datasets were taken from [44]. We have made the data available for the interactive work on pathways and genes. Using the data as part of the computational process was very limited since it turned out to be very sparse in the pathways we have worked on and due to its somewhat problematic specificity.

5.2 Ergosterol Metabolism

Ergosterol is an essential lipid in yeast which is similar to cholesterol in mammals. Ergosterol's primary role is in the cell membranes but it is also involved in aerobic metabolism, sterol

uptake and sterol transport. Ergosterol metabolism is understood rather well. As many of the knockout experiments of [32] targeted that pathway, and it is believed to undergo significant transcription regulation, we chose it to test our analysis techniques. Ergosterol metabolism is composed of two pathways in series. The first, the mevalonate pathway, transforms acetyl-CoA to farnesyl and provides essential components for few important metabolic pathways (e.g. heme and quinones). The second part transforms farnesyl to ergosterol. Much of the regulation of ergosterol is believed to be transcriptionally mediated, but the actual details are known only in part [18, 8, 49].

Figure 5.1 shows the basic known ergosterol metabolic pathway from farnesyl to ergosterol, including a series of 11 enzymes and three transcription factors. It is important to stress here the difference between metabolic pathways and regulatory networks: The fact that two enzymes follow each other in a biochemical process does not mean their *transcription* regulation is directly connected. We have modeled the ergosterol dependency structure core as the set of variables, with dependencies marked only between known transcription factors and their targets. In other words, no dependency was prescribed between enzymes. We have used this core and the expression data described above to test GENESYS.

Table 5.2 presents some of the pathway states (combination of pathway variables states) observed in our data set. A number of experiments showed a global reaction of the pathway: In those experiments most of the pathway enzymes underwent significant change. This is presumably the result of some self regulatory mechanism (and indeed ergosterol itself is reported to function as transcription regulator for its pathway enzymes). However, many other experiments (about 40) showed a change in one or more of the pathway genes, which is not explained by the above mechanism. Those experiments may be explained by a more elaborate model. This motivates our attempt to expand the model and explain more of the data.

5.3 Transcription Factors Screening

Out of the ˜6200 yeast ORFs, we identified 130 putative transcription factors (TFs). For this we used SGD annotations, as well as typical structural motifs (e.g., zinc fingers). We then applied the single node expansion algorithm, limiting the candidates for node expansions to these putative TFs. In the first test, we ranked the fitness gain of each of the putative TFs against a "naked" core consisting of the eleven ERG enzymes with no dependencies among them. HAP1 was ranked second out of 130 (Table 5.3), in agreement with the known role of HAP1 in ERG11 regulation. TUP1 is a general repressor and was thus ranked lower, ROX1 was less expressed in the data and was ranked much lower.

Having gained some confidence in the process quality, we focused on improving our un-

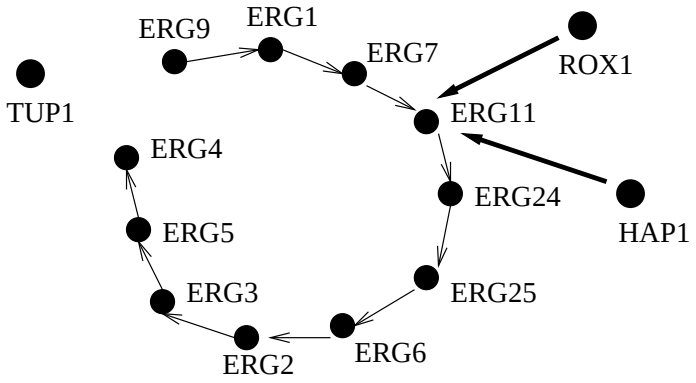

Figure 5.1: The ergosterol pathway from farnesyl to ergosterol. Only enzymes (names starting with ERG) and known transcription factors (ROX1, HAP1, TUP1) are shown. Thin arrows indicate subsequent enzymes (not a model dependency). Thick arrows indicate model dependencies

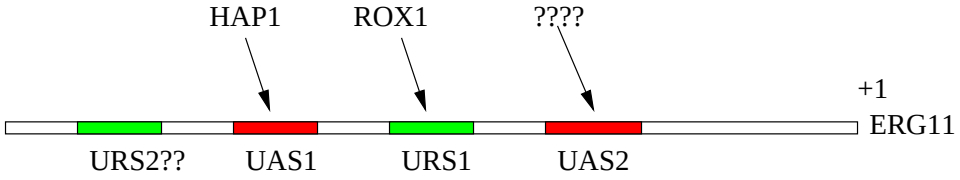

Figure 5.2: ERG11 promoter region according to [49]. UAS/URS: upstream activation/repression site. The transcription factors HAP1 and ROX1 induce and repress, respectively, ERG11 transcription via the binding sites UAS1 and URS1. UAS2 was identified as a likely binding site of an unknown activator. One of our goals in this study was to demonstrate that we can suggest the identity of the missing activator.

derstanding of ERG11 regulation. Turi and Loper (1992) analyzed the promoter region of ERG11 with results that are summarized in Figure 5.2. This time we applied the single node expansion to a core consisting of the eleven ERG enzymes as well as HAP1 and ROX1 as regulators of ERG11. The algorithm measured the improvement in fitness contributed by each of the 130 TFs, and an uncharacterized gene (YLR266C) was ranked first. That gene improves the fitness of ERG11 (and others). Remarkably, it also has a good homology to HAP1 (33% identity, 50% similarity along 100 amino acids and even better in a shorter range). Moreover, analyzing ERG11 logic as a function of HAP1, ROX1, TUP1 and the novel TF shows that the effect of the new putative TF on ERG11 is inductive (as expected from a UAS2 binding gene). We thus have evidence from three different methods: sequence homology, promoter analysis indicating a second inducer should exist, and our screening procedure using some 360 different expression profiles in distinct cell states. All three support the hypothesis that YLR266C is indeed an ERG11 regulator that might bind to UAS2.

name	9	1	7	11	24	25	6	2	3	5	4
erg2	+	+		+	+	+	+	-	+	+	+
erg11	+	+	+	-	+	+	+	+	+	+	+
erg3	+	+		+		+	+	+	-	+	+
erg4	+	+				+			+	+	-
hmg2	+	+		+			+		+	+	+
erg28		+		+		+			+	+	+
10 exp											+
3 exp											-
4 exp									+		
3 exp									-		

Table 5.2: States of the enzymes in the ergosterol pathway observed in [32]. Columns represent "ERG" genes (see Figure 5.1) with the enzyme number on the column label. Rows represent knock out experiments, with the suppressed gene (or the number of experiments manifested the same pattern) as the row label. + : up regulation, - : down regulation. The upper part of the table represents a global reaction of the pathway, presumably mediated through levels of ergosterol or ergosterol intermediates. The lower states are totally unexplained by current ergosterol regulation models and are highly unlikely to be entirely noise. 13 additional measured states in which some ERG genes are up or down regulated are not shown.

5.4 Screening All Genes

The admission of putative transcription factors only as added variables was important in the reduction of model space, and it allowed us to obtain very specific results. It is, however, interesting to try and screen all ~6200 yeast ORFs against the ergosterol core. This type of analysis may discover more general patterns of regulation that cannot be directly tagged as "A is a factor of B". Still, as shown below, some interesting biology may be learned from it. The results of such a screen are given in Table 5.4. The two top ranking genes, POS5, YBR043C, are both of unknown function. POS5 has homology to iron metabolism enzymes. Both present significant fitness gain for ERG4 regulation. ERG4 is the last of the ergosterol pathway enzymes, is not essential and little is known about its regulation. Figure 5.3 gives a more detailed look on the relations among the three genes. Note that using standard clustering or similarity, the behavior of ERG4 in experiments with no POS5, YBR043C involvement would have masked the pattern identified here.

The fourth gene in the screening list is INO1 that is involved in inositol biogenesis. Inositol has a regulatory function in the phospholipid pathway (adjacent to ergosterol). Note that the

Gene	Annotation	Gain
PIP2	Peroxisome proliferation	0.086
HAP1	erg11 activator	0.070
YDR213W	Unknown	0.062
GLN3	nitrogen catabolite	0.060
RAP1	transcription	0.054

Table 5.3: Putative transcription factors that ranked best in an expansion of the "naked" ergosterol core

dependency is localized differently (improving different variables) in that case. The relation of GAS1 to ergosterol might be rooted in its function in the cell wall. The dependency between our core and MKK2 is very reasonable considering its function in the signaling pathway to the cell wall protein PCK1. The 11th gene in the list is ERG10, which is the first gene in the mevalonate pathway leading to our core.

The dependencies revealed by the general 1-expansion screening can serve as the basis for deeper biological exploration. The process pinpoints statistically significant patterns which are hard to identify otherwise. In contrast with the TF 1-expansion screening, the results are less direct and do not identify specific dependencies.

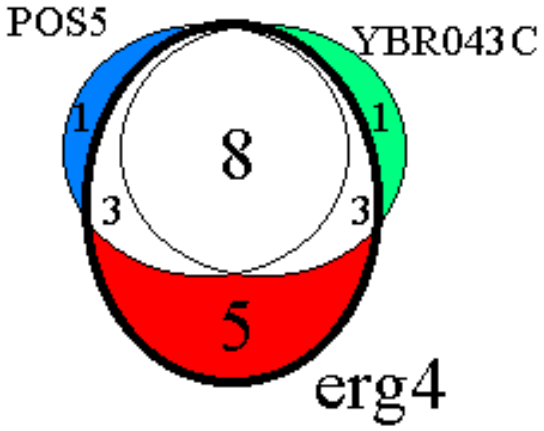

Figure 5.3: ERG4 dependent genes. The Venn-like diagram represents all experiments in which ERG4, POS5 and YBR043C were induced. Induction in this case is any up-regulation with regulation specificity less than 0.01. The number inside each of the sets indicates its size. The diagram shows that induction of POS5 and YBR043C strongly correlates with ERG4 induction (11/12 experiments in both cases). ERG4 is showing a second, separate regulation pattern (5 experiments) which is unrelated to POS5,YBR043C.

.

Gene	ORF	Annotation	Gain	Gain location
1.POS5	YPL188W	Unknown	0.026	ERG4
2.YBR043C	YBR043C	Unknown	0.023	ERG4
3.YDL054C	YDL054C	Unknown	0.021	ERG4,ERG5,ERG25
4.INO1	YJL153C	Inositol biosynthesis	0.018	ERG6,ERG25,ERG5
5.YDR531W	YDR531W	Unknown	0.017	ERG4,ERG5,ERG25,ERG6
6.RLI1	YDR091C	Unknown	0.017	ERG6,ERG25
7.GAS1	YMR307W	cell surface glycoprotein	0.016	ERG4,ERG6,ERG5,ERG25
8.ZRT2	YGL225W	zinc transporter	0.016	ERG6,ERG25,ERG5
9.YDR302W	YDR302W	Unknown	0.016	ERG4,ERG6,ERG25,ERG5
10.MKK2	YPL140C	PCK1 signaling	0.016	ERG4
11.ERG10	YPL028W	Ergosterol metabolism	0.016	ERG6,ERG25,ERG5,ERG7
12.YHR199C	YHR199C	Unknown	0.015	ERG6,ERG25,ERG5,ERG7
13.ARG4	YHR018C	Arginine biosynthesis	0.015	ERG6,ERG5,ERG25,ERG4
14.YDR426C	YDR426C	Unknown	0.015	ERG4,ERG25
15.YLR290C	YLR290C	Unknown	0.015	ERG4,ERG25,ERG5

Table 5.4: Top ranking genes (among all yeast ORFs) in 1-expansion of the ergosterol core pathway. Gene annotations are from SGD. 'Gain': the increase to fitness by using the additional variable. 'Gain location': the core genes whose regulation specificity was significantly improved by the variable, in order of significance.

Chapter 6

The GENESYS environment

GENESYS (GEnetic Network Expansion SYStem) is a new software platform implementing the concepts and methods described above. The environment includes engines for representing networks and computing fitness, a flexible expansion algorithm, viewers for visualization of biological data sets, an application to enable interactive usage of the viewers and engine and an internal database scheme for the storage of datasets and pathways.

The system was implemented in C++ and Perl/Tk under Linux (about 25000 code lines). It is built for efficient manipulation of data set with thousands of genes, thousands expression profiles on them and cores of up to 30 nodes. See Chapter 4 for statistics on its run time and performance.

The GENESYS environment is built as a modular prototype composed of back-end and front-end implemented in C++ and Perl/Tk. The front-end is a GUI application featuring a wide selection of viewers and providing means to invoke computational processes using the back-end. The back-end provides a set of libraries and computational engines using them. Both front-end and back-end use the same database, currently implemented over flat files.

To simplify the software architecture, and since GENESYS was implemented as a prototype, the relation between the front-end and back-end are based on process invocation and the file system. No inter-process communication is used in this stage. The use of flat files as database was also meant as a simplification but the code is built so that using a real database would effect only wrapper classes.

We will briefly outline in what follows the basic features currently supported by GENESYS front-end and the services provided by its back-end.

6.1 GENESYS viewers

The main GENESYS interface is the dependency structure editor, shown in Figure 6.1. The editor enables creation, viewing and saving of network cores, manipulation of nodes and arcs in the graph and it also provides connectivity to the logic viewer described below.

The GENESYS variable list window is a simple list widget providing access to the variable universe U with all the information attached to it. It is shown in Figure 6.2.

The GENESYS matrix viewer visualizes time series information by presenting the states of a selected set of variables and their development in time.

The knockout viewer (Figure 6.3) provides means for the analysis of knockout compendiums - large collections of expression profiles in which different genes were systematically knocked-out or over-expressed (e.g., [32]). The user selects a small set of **focus variables** and the viewer displays a graph in which nodes correspond to **genes equivalence classes** grouped by the effect of the focus variable knockout on their mRNA levels. Using this viewer, one can explore the complex relations between knocked-out genes as revealed by the families of genes behaving similarly on the different knock-out experiments.

The logic viewer (Figure 6.4) presents a **binary decision diagram** (BDD) given a focus variable and its current incoming neighbors. A BDD [9] represents the effect of inputs values on output value of the focus variable and provides insight into the structure of the discrete logic induced by a given dependency structure. To create a BDD, a layered tree is constructed where each level represents the addition of a new input variable. The leafs of the tree represent groups of experiments in which the input added up to the leaf level takes the same value. A parent child relation is added whenever the input assignment of level l is a subset of the assignment of the bigger input set at level $l + 1$. In the final tree level, the output value is introduced to reveal the relation between input assignment and output value. Note that the order of the inputs change the topology of the tree. Using the BDD, users can explore the logic underlying a putative dependency structure and deduce functional roles each input may have (activator/repressor or more complicated functions).

The interaction window (Figure 6.5) shows putative interactions from the GENESYS database. The window lets the user navigate in the interaction graph by showing each time the neighborhood of a focus variable which may be selected interactively. Different edge colors are used for different types of interaction.

The optimization window tracks the progress of an expansion process by displaying the quality of the current best solution and updating the dependency structure window with its structure. The user can control the progress of the optimization process, start, stop, resume and kill it.

The one-expansion screen result window lists the be expansion candidates of a given core

after the completion of a screen. The user can sort the list according to various parameters and evaluate the affinity of each of the expansion variables to different parts of the core.

The script window is the place of the main menu bar in the system, providing control on all system features. It also enables writing scripts that specify the logic of a given system. Currently not in use, the system supports compiling scripts into a dependency structure and tabular representation of the logical relation.

6.2 GENESYS back-end services

The GENESYS back-end is organized in the following libraries :

The base library provides general infrastructure, portability macros and several design patterns implementations.

The util library provides some utilities and data-structures which are not available from the Standard Template Library. It also contains some basic text parsers.

The bpm library provides interface to the GENESYS databases and objects to represent variables, systems, experiments, datasets and logic. It also provides some dependency structure evaluators and factories for dynamic object generation of such evaluators. The implementation of the consistency and *rSpec* algorithm is provided as part of the evaluator inheritance structure.

The onexp library provides a computational engine to perform one node expansion screening and multi node expansion searches.

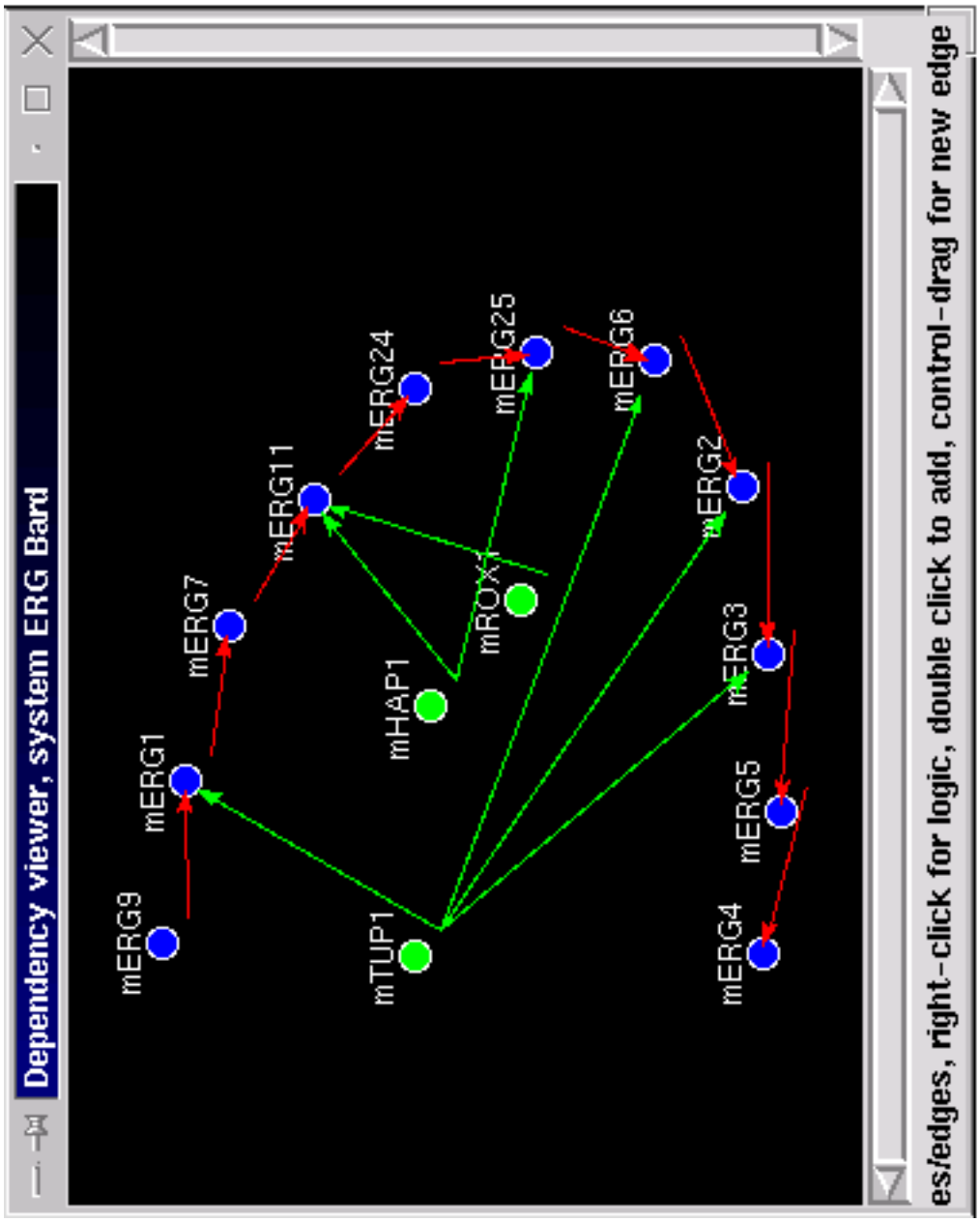

Figure 6.1: GENESYS dependency structure editor. Using this editor, users can manipulate cores and visualize expansion results.

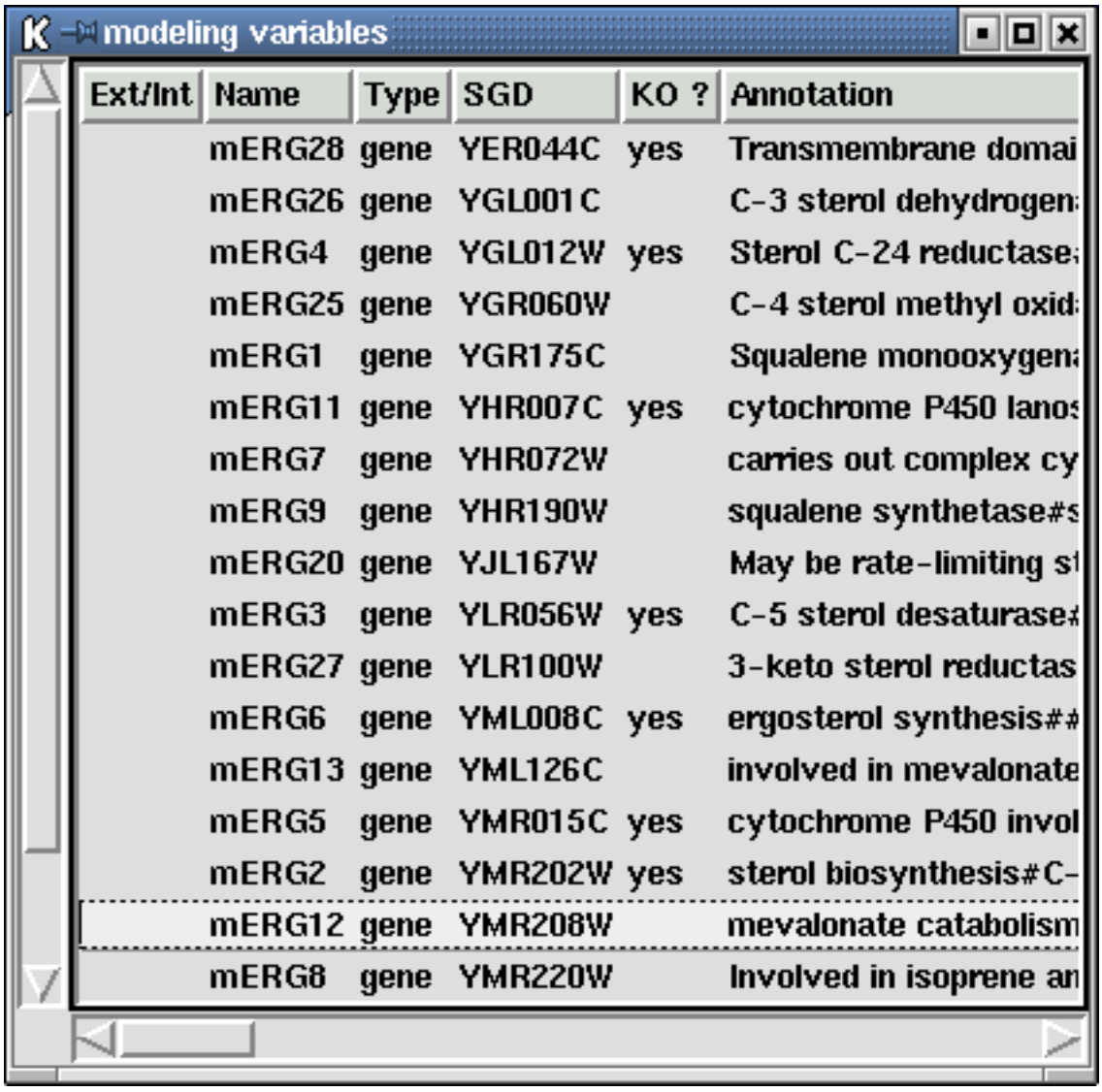

Figure 6.2: GENESYS variable list window. Representing the current variable universe and enabling the user to query and modify it.

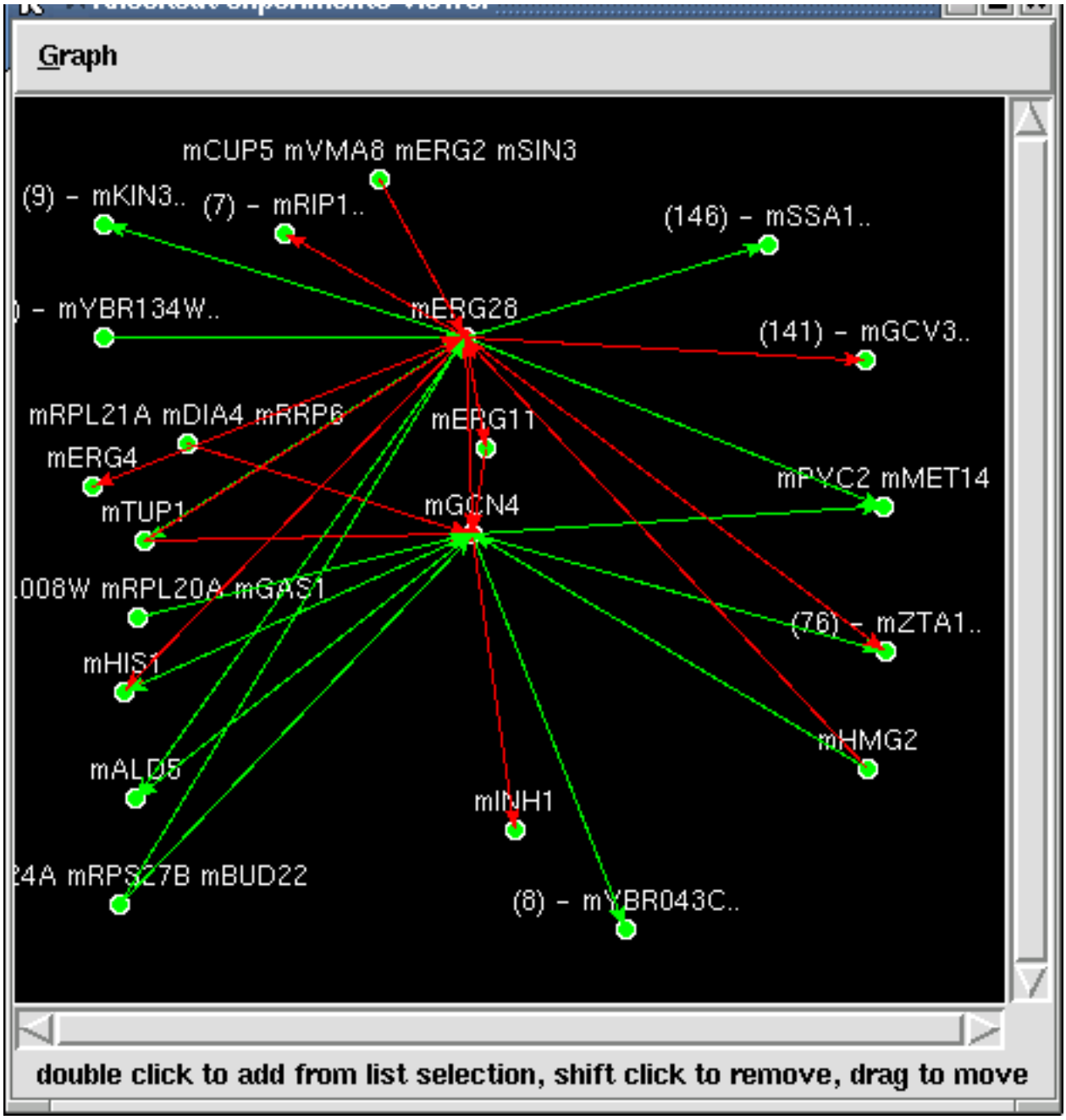

Figure 6.3: GENESYS knockout viewer. Visualizing a large collection of knockout experiments by grouping genes according to their behavior in selected experiments. Each node represents a group of genes which are equivalent in the selected set of focus genes (ERG28 and GCN4 in this example). Large groups are denoted by their size and an example variable and can be opened by clicking the mouse. User can manipulate the set of focus set to explore the dataset interactively.

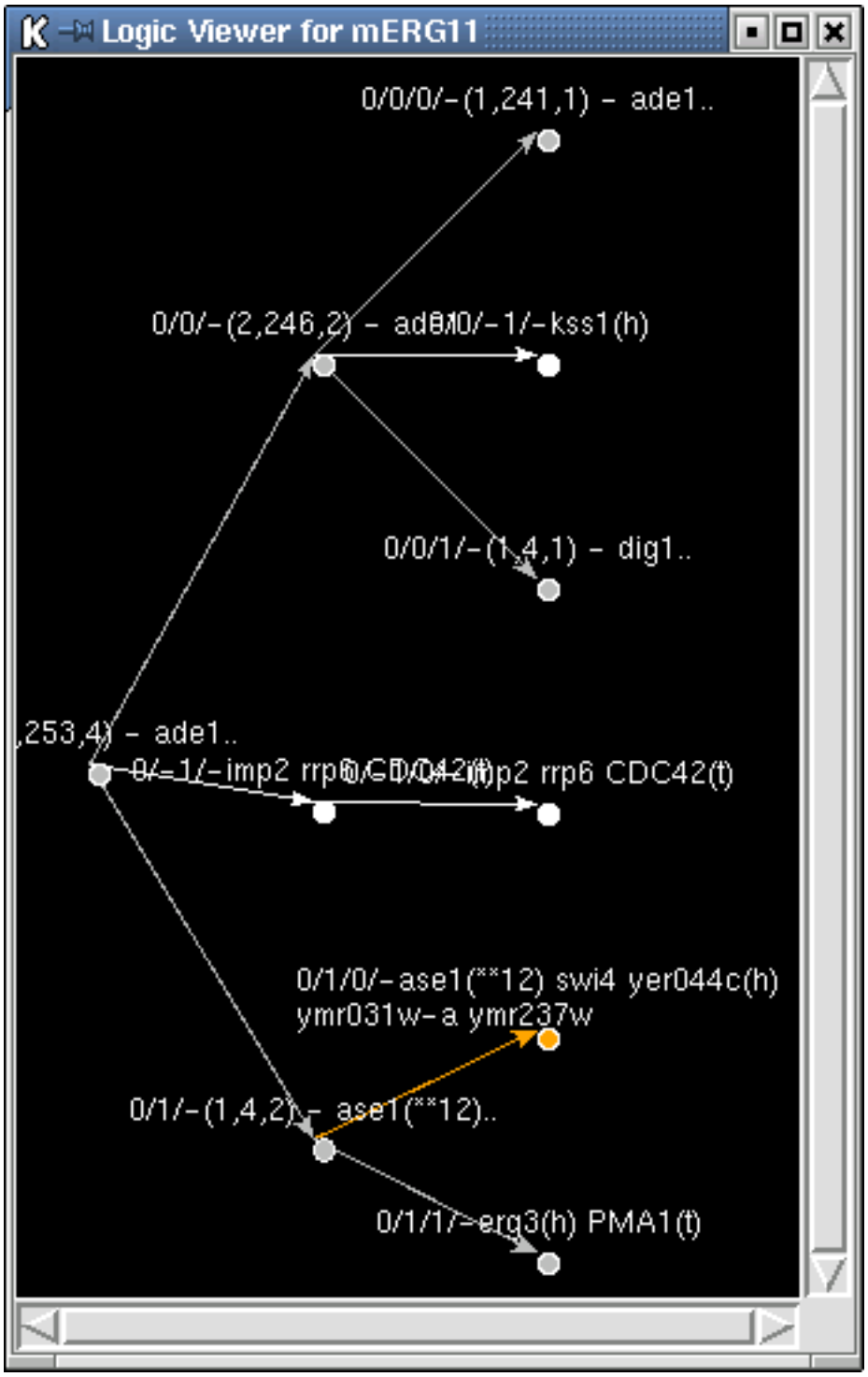

Figure 6.4: GENESYS logic viewer. A Binary Decision Diagram showing the dependency of a selected variable in the values of its direct inputs in an underlying dependency graph

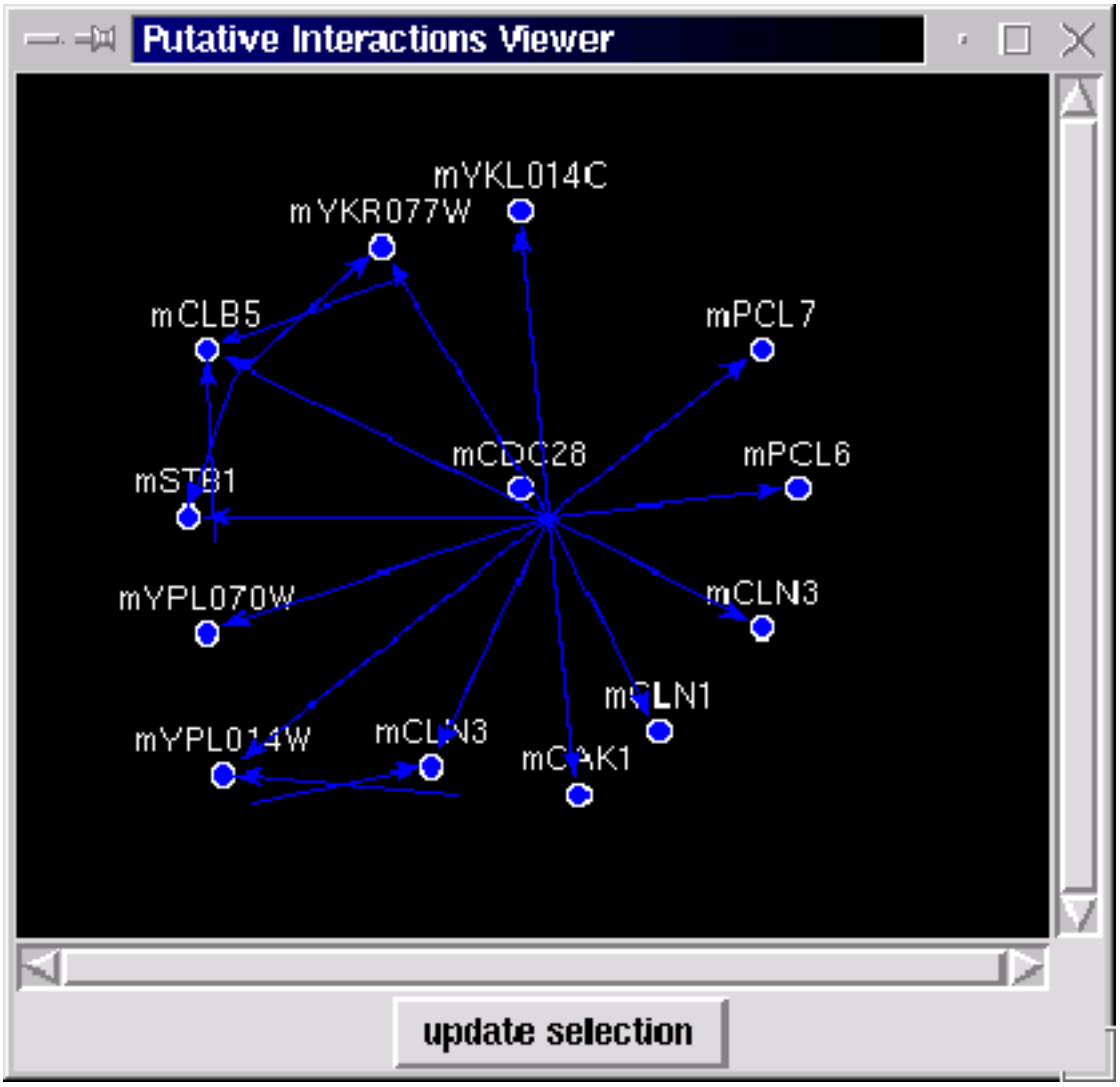

Figure 6.5: GENESYS interaction viewer. Visualizing a set of gene/protein interactions from many sources. User can navigate through the graph by clicking on nodes.

Bibliography

[1] The chipping forecast. Special supplement to Nature Genetics Vol 21, 1999.

[2] T. Akutsu, S. Kuhara, O. Maruyama, and S. Miyano. Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions. In *Proc. Ninth Annual ACM-SIAM Sympoium on Discrete Algorithms (SODA 98)*, pages 695–702, 1998.

[3] T. Akutsu, S. Miyano, and S. Kuhara. Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Proceedings of the 1999 Pacific Symposioum in Biocomputing (PSB 99)*, pages 17–28, 1999.

[4] T. Akutsu, S. Miyano, and S. Kuhara. Algorithms for identifying Boolean networks and related biological networks based on matrix multiplication and fingerprint functions. In *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology (RECOMB 00)*, pages 8–14, 2000.

[5] B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J.D. Watson. *Molecular Biology of the Cell.* Garlend Publishing Inc., New York and London, 1994.

[6] U. Alon, N. Barkai, D. A. Notterman, G. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *PNAS*, 96:6745–6750, June 1999.

[7] C.A. Ball et al. Saccharomyces genome database provides tools to survey gene expression and functional analysis data. *Nucleic Acids Research*, 29:80–1, 2001.

[8] G.F. Bammert and J.M Fostel. Genome-wide expression patterns in Saccharomyces cerevisiae: comparison of drug treatments and genetic alterations affecting biosynthesis of ergosterol. *Antimicrobial Agents and Chemotherapy*, 44:1255–1265, 2000.

[9] R. E. Bryant. Graph based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–91, 1986.

[10] P. Bucher. Regulatory elements and expression profiles. *Curr. Opin. Struct. Biol.*, 9:400–7, 1999.

[11] T. Chen, V. Filkov, and S. S. Skiena. Identifying gene regulatory networks from experimental data. In *Proceedings of the Third Annual International Conference on Computational Molecular Biology (RECOMB 99)*, pages 94–103, 1999.

[12] R.J. Cho et al. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol. Cell.*, 2:65–73, 1998.

[13] E.J. Cockayne, R.M. Dawes, and S.T. Hedetniemi. Total domination in graphs. *Networks*, 10:211–9, 1980.

[14] The FlyBase Consortium. The FlyBase database of the Drosophila genome projects and community literature. *Nucleic Acids Res.*, 27:85–8, 1999.

[15] The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[16] M.C. Costanzo et al. YPD, PombePD and WormPD: model organism volumes of the BioKnowledge library, an integrated resource for protein information. *Nucleic Acids Research*, 29:75–9, 2001.

[17] T. M. Cover and J. M. Thomas. *Elements of Information Theory.* John Wiley & Sons, London, 1991.

[18] G. Daum, N.D. Lees, M. Bard, and R. Dickson. Biochemistry, cell biology and molecular biology of lipids of Saccharomyces cerevisiae. *Yeast*, 14:1471–1510, 1998.

[19] E.H. Davidson. *Genomic Regulatory Systems. Development and Evolution.* Academic Press, San Diago, 2001.

[20] J. DeRisi, V. Iyer, and P. Brown. Exploring the metabolic genetic control of gene expression on a genomic scale. *Science*, 278:680–686, 1997.

[21] J. DeRisi, L. Penland, P.O. Brown, et al. Use of a cDNA microarray to analyse gene expression patterns in human cancer. *Nature Genetics*, 14:457–460, 1996.

[22] P. Dhaeseleer, X. Wen, S. Fuhrman, and R. Somogoyi. Linear modeling of mRNA expression levels during CNS development and injury. In *Proceedings of the 1999 Pacific Symposioum in Biocomputing (PSB 99)*, pages 41–52, 1999.

[23] M. Diehn, M.B. Eisen, D. Botstein, and P.O. Brown. Large-scale identification of secreted and membrane-associated gene products using DNA microarrays. *Nature Genetics*, 25:58–62, 2000.

[24] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *PNAS*, 95:14863–14868, 1998.

[25] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. In *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology (RECOMB 00)*, pages 127–135, 2000.

[26] E.E. Furlong, E.C. Andersen, B. Null, K.P. White, and M.P. Scott. Patterns of gene expression during drosophila mesoderm development. *Science*, 293:1629–33, 2001.

[27] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., San Francisco, 1979.

[28] A. P. Gasch et al. Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell.*, 11:4241–57, 2000.

[29] T. R. Golub, D. K. Slonim, et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, October 1999.

[30] T.J. Griffin and R. Aebersold. Advances in proteome analysis by mass spectrometry. *Journal of Biological Chemistry*, 19:375–8, 2001.

[31] B.B. Haab, M.J. Dunham, and P.O. Brown. Protein microarrays for highly parallel detection and quantitation of specific proteins and antibodies in complex solutions. *Genome Biology*, 2:RESEARCH0004, 2001.

[32] T.R. Hughes et al. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–26, 2000.

[33] T. Ideker, J.A. Thorsson, V. Ranish, R. Christmas, J. Buhler, J.K. Eng, R. Bumgarner, D.R. Goodlett, R. Aebersold, and Hood L. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 291:929–34, 2001.

[34] T.E. Ideker, V. Thorsson, and R.M. Karp. Discovery of regulatory interaction through perturbation: inference and experimental design. In *Proceedings of the 2000 Pacific Symposioum in Biocomputing (PSB 00)*, pages 305–316, 2000.

[35] V.R. Iyer, C.E. Horak, C.S. Scafe, D. Botstein, M. Snyder, and P.O. Brown. Genomic binding sites of the yeast cell-cycle transcription factors sbf and mbf. *Nature*, 409:533–8, 2001.

[36] M. Kanehisa and S. Goto. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.*, 28:27–30, 2000.

[37] R. M. Karp, R. Stoughton, and K. Y. Yeung. Algorithms for choosing differential gene expression experiments. In *Proceedings of the Third Annual International Conference on Computational Molecular Biology (RECOMB 99)*, pages 208–217, 1999.

[38] S.A. Kauffman. *The Origins Of Order, Self Organization and Selection in Evoultion.* Oxford University Press, 1993.

[39] E.S. Lander et al. Initial sequencing and analysis of the human genome. International human genome sequencing consortium. *Nature*, 409:860–921, 2001.

[40] S. Liang, S. Fuhrman, and R. Somogoyi. REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. In *Proceedings of the 1998 Pacific Symposioum in Biocomputing (PSB 98)*, pages 18–29, 1998.

[41] E. M. Marcotte, M. Pellegrini, H. L. Ng, D. W. Rice, T. O. Yeates, and D. Eisenberg. Detecting protein function and protein-protein interactions from genome. *Science.*, 285(5428):751–753, 1999.

[42] D. Pe'er, A. Regev, G. Elidan, and N. Friedman. Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 17 Suppl 1:S215–24, 2001.

[43] B. Ren et al. Genome-wide location and function of DNA binding proteins. *Science*, 290:2306–9, 2000.

[44] B. Schwikowski, P. Uetz, and S. Fields. A network of protein-protein interactions in yeast. *Nature Biotechnolgy*, 8:1257–61, 2000.

[45] R. Sharan and R Shamir. CLICK: a clustering algorithm with applications to gene expression analysis. In *Proceedings of the Eighth Annual Conference on Intelligent Systems for Molecular Biology (ISMB 00)*, pages 307–316, 2000.

[46] P.T. Spellman et al. Comprehensive identification of cell cycle regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol. Biol. Cell.*, 9:3273–3297, 1998.

[47] A. Tanay and R. Shamir. Computational expansion of genetic networks. *Bioinformatics*, 17(Suppl 1):S270–8, 2001.

[48] S. Tavazoie, J.D. Hughes, M.J. Campbell, R.J. Cho, and G.M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22:281–285, 1999.

[49] T.G. Turi and J.C. Loper. Multiple regulatory elements control expression of the gene encoding the Saccharomuces cerevisiae Cytochromoe P450, lanosterol 14a-demethylase (ERG11). *JBC*, 267:2046–56, 1992.

[50] P. Uetz et al. A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. *Nature*, 403:623–7, 2000.

[51] E. Wingender, X. Chen, E. Fricke, R. Geffers, R. Hehl, I. Liebich, M. Krull, V. Matys, H. Michael, R. Ohnhauser, M. Pruss, F. Schacherer, S. Thiele, and S. Urbach. The TRANSFAC system on gene expression regulation. *Nucleic Acids Res.*, 29:281–3, 2001.

[52] J. Wojcik and V. Schachter. Protein-protein interaction map inference using interacting domain profile pairs. *Bioinformatics*, 17 Suppl 1:S296–305, 2001.

[53] C.H. Yuh, H. Bolouri, and E.H. Davidson. Genomic cis-regulatory logic: Experimental and computational analysis of a sea urchin gene. *Science*, 279:1896–1902, 1998.

[54] H. Zhou, J.D. Watts, and R. Aebersold. A systematic approach to the analysis of protein phosphorylation. *Nature Biotechnology*, 19:375–8, 2001.

[55] A. Zien, R. Kuffner, R. Zimmer, and T. Lengauer. Analysis of gene expression data with pathway scores. In *Proceedings of the Eighth Annual Conference on Intelligent Systems for Molecular Biology (ISMB 00)*, pages 407–417, 2000.