Chain functions and scoring functions in genetic networks

I. Gat-Viks* and R. Shamir

School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel

Received on January 6, 2003; accepted on February 20, 2003

ABSTRACT
One of the grand challenges of system biology is to reconstruct the network of regulatory control among genes and proteins. High throughput data, particularly from expression experiments, may gradually make this possible in the future. Here we address two key ingredients in any such ‘reverse engineering’ effort: The choice of a biologically relevant, yet restricted, set of potential regulation functions, and the appropriate score to evaluate candidate regulatory relations.

We propose a set of regulation functions which we call chain functions, and argue for their ubiquity in biological networks. We analyze their complexity and show that their number is exponentially smaller than all boolean functions of the same dimension. We define two new scores: one evaluating the fitness of a candidate set of regulators of a particular gene, and the other evaluating a candidate function. Both scores use established statistical methods. Finally, we test our methods on experimental gene expression data from the yeast galactose pathway. We show the utility of using chain functions and the improved inference using our scores in comparison to several extant scores. We demonstrate that the combined use of the two scores gives an extra advantage. We expect both chain functions and the new scores to be helpful in future attempts to infer regulatory networks.

Contact: [iritg,rshamir]@post.tau.ac.il

INTRODUCTION
The regulation of mRNA transcription is critical to cellular function. Large-scale gene expression (GE) measurements, using, e.g. DNA microarrays (Derisi et al., 1997; Lockhart et al., 1996), may enable the reconstruction of the regulatory relations among genes. By the regulatory relation of a target gene, we mean the set of genes that together regulate it, and the particular logical function by which this regulation is determined. This paper focuses on inference of regulatory relations from GE profiles.

Most current expression analysis tools are based on clustering (e.g. Eisen et al. (1998), Ideker et al. (2001) and Sharan et al. (2002)). Such analyses successfully reveal genes that are co-regulated, but not their regulatory relations. More advanced approaches rely on mathematical models of the regulation process. Different models at various levels of detail have been suggested. These include boolean (Ideker et al., 2000; Akutsu et al., 1999; Liang et al., 1998), qualitative (Thieffry and Thomas, 1998), linear (Dhaeseleer et al., 1999), differential equations (Chen et al., 1999) and detailed biochemical models (Arkin et al., 1998).

A key obstacle in the inference of regulation relations is the large number of possible solutions, and consequently the unrealistically large amount of data needed to identify the right one. This inherent complexity of genetic network inference (Akutsu et al., 1998, 1999) led researchers to seek ways around this problem. Ideker et al. (2000) studied how to dynamically design experiments so as to maximize the amount of information extracted. Friedman et al. (2000) used Bayesian networks to reveal only parts of the genetic network which are strongly supported by the data. Hanisch et al. (2002) and Ideker et al. (2002) used prior knowledge about the metabolic network structure in order to identify relevant processes in GE data. Another approach to tackle the complexity issue is to reduce the set of allowed network models. Tanay and Shamir (2001) suggested a method of ‘network expansion’, in which one starts from a partially known network and augments it according to the GE data. Pe’er et al. (2002) make certain biologically-motivated assumptions on the local topology of the network, which reduce the space of possible global networks. Several other works used restrictive models of regulation relations (e.g. decision trees (Segal et al., 2001)).

In this paper, we study two nuclear problems in regulation relation inference, which are at the heart of inferring transcription networks: (1) determining the set of regulators of a gene (the gene is called regulatee and the set is called its regulators set), and (2) deducing the precise mathematical function by which the regulators set determines the gene’s transcription (the regulation function).

We assume throughout a boolean model, i.e. each of the candidate regulators and regulatees can be in one of two
states: expressed (present) or non-expressed (absent). The inference of regulatory relation of a single gene is a fundamental step in the long-term effort to infer regulation networks.

To study these problems we design two new methods which evaluate how well a candidate regulatory relation of a particular regulatee fits experimental data. Such fitness scores are essential in order to pick the right relation among many candidates. Our first score evaluates the specificity of the regulators set. The second score evaluates how well a particular regulation function (for a given regulators set) fits the data. Both scoring functions utilize established statistical methods, and are expressed as p-values, and thus are not very sensitive to over-fitting. Moreover, due to the Gaussian shape of these scoring functions, they always score only a few solutions at the high end. The two scores are affected differently by different problem parameters, so using both scores in combination gives an added advantage.

The second component of this work is the introduction and study of a novel family of regulation functions called chain functions. In a chain function, the state of the regulatee depends on the influence of its direct regulator, whose activity may in turn depend on the influence of another regulator, and so on in a chain of dependencies (we will provide formal definitions later). The class of chain functions has several important advantages: First, as we shall argue, these functions reflect common biological regulation behavior, and often occur in networks, so many real biological regulatory relations can be elucidated using them. Second, as we shall show, the number of chain functions with \( n \) control variables is \( \Theta(n! \cdot \log^2 n)^{n+1} \). This number is exponentially smaller than the total number of boolean functions. Hence, by limiting inference to chain functions, we reduce exponentially the size of the candidate solution search space.

We apply our approach to transcription profiles of the yeast galactose pathway (Ideker et al., 2001). First, we demonstrate the advantages of using chain functions instead of searching through all boolean functions. Second, we use the yeast galactose pathway of Ideker et al. (2001) to compare our scores to several other fitness scores which were previously proposed for network inference, and show that on these data, our score outperforms them. Third, we show that by using in combination our two scores for regulator set and regulation function, we can obtain very high ranking of the correct solution.

The paper is organized as follows. We start by providing a formal framework for the model. We then define the chain functions, motivate them biologically and present their analysis. Next, the fitness scores are presented and analyzed. Finally, results on real transcription profiles are reported.

THE NETWORK MODEL

In this section we describe the formal model for our analysis and tools. The formalism follows Tanay and Shamir (2001) and Liang et al. (1998).

The set of all variables is denoted by \( U \). These may include genes, mRNAs, proteins and ligands such as disaccharides and amino acids. The set of states that each variable in \( U \) may attain is denoted by \( V \). A candidate regulation function for a variable \( g \) which is regulated by \( n \) variables \( R_n \subseteq U \), has the form \( f^g : V^n \rightarrow V \). In other words, the state of \( g \) is a function of the states of the variables in \( R_n \). We use the term regulatee for the regulated variable \( g \), and the term regulator (of \( g \)) for each variable in \( R_n \). The regulator set may actually include biological regulator, co-regulators, co-factors, etc.

The GE data consist of \( l \) conditions, \( E = \{e_1, \ldots, e_l\} \). Condition \( j \) is defined by a vector of levels (typically expression ratios) for each variable in \( U \), and by a set of variables that were externally perturbed (knocked-out or over-expressed) in condition \( j \). These externally perturbed variables must be indicated, as their levels are not determined by their regulation functions. We assume that the data are of steady state, so additional synchrony assumption is not needed, and the states of the regulators determine the state of the regulatee in the same condition. A simple modification of the model applies to time-series synchronous data, where the state of the regulatee is taken at one time point later than that of the regulators (cf. Tanay and Shamir (2001)).

We will narrow the range of network models by adding constraints as follows: We assume that the states are discrete, and that the functional relations are deterministic. Each variable can have only two levels: either on (1) or off (0), i.e. \( V := \{0, 1\} \). This can be achieved, for example, by setting a threshold on the input data values. We shall use state \( x, j \) to denote the binary value of variable \( x \) in condition \( j \), and suppress \( j \) whenever possible for readability. Each regulatee is regulated through a boolean function of at most \( n \) arguments. The boolean model is a drastic simplification of real biology, yet it captures important features of biological systems. Similar simplifying choices are frequently made in order to reduce the number of degrees of freedom, and to avoid over-fitting (cf. Akutsu et al. (1999) and Kauffman (1974)).

CHAIN FUNCTIONS

We now propose a class of regulation functions, called chain functions. We argue that this class covers many common regulation scenarios in biology. We analyze the chain functions and show that the set of chain functions is exponentially smaller than the set of all boolean regulation functions.
Definitions. We first define some related terms. Recall that the state of a variable in a condition is 1 if that variable is present and 0 if it is absent. The chain function $f^{g_0}$ on the variables $g_n, \ldots, g_4$ will determine the value of the regulatole $g_0$. The order of the variables is important, as it reflects the order of influence among them, as will be explained below. For that reason, we shall sometimes refer to $R_n$ as the ordered set $g_n, \ldots, g_1$. We call $g_i$ the predecessor of $g_{i-1}$ and the successor of $g_{i+1}$. $f^{g_0}$ depends on $n$ auxiliary control bits $c_n, \ldots, c_1$ that attain values $A$ or $R$. The semantic is that $c_i = A$ (R) if $g_i$ activates (represses) $g_{i-1}$. These two options are exhaustive. Note that the activation or repression by $g_i$ is of $g_{i-1}$ and not of the regulatole $g_0$. We also call $c_i = A$ and $c_i = R$ positive and negative control, respectively.

The control bit $c_i$ defines whether a regulator $g_i$ is a repressor or an activator of its successor $g_{i-1}$. However, this effect takes place only if the regulator $g_i$ is currently active. Consider, for example, a regulator $g_2$ with control bit $A$. $g_2$ will activate $g_1$, but only if $g_2$ is actually active. Inactivity may be due to its absence, or $g_2$ might be present and inactive, if it is repressed by its predecessor $g_3$. To define this situation, we use two concepts: the activity of a variable $a(g_i)$ and its influence on its successor $infl(g_i)$. Activity can be either 0 or 1; influence can be either positive (P) or negative (N). Their definitions are recursive. The influence on $g_n$ is always positive. Formally, $infl(g_{n+1}) = P$. The activity of $g_i$ is 1 iff the influence on it is positive and its state is 1:

$$a(g_i) = 1 \text{ iff } (infl(g_{i+1}) = P \text{ and state}(g_i) = 1)$$

(1)

The influence of $g_i$ on $g_{i-1}$ is defined by:

$$infl(g_i) = P \text{ iff } \left\{ \begin{array}{l} c_i = A \text{ and } a(g_i) = 1, \text{ or} \\ c_i = R \text{ and } a(g_i) = 0 \end{array} \right.$$  

(2)

Equivalently, $infl(g_i) = N \text{ iff } [c_i = A \text{ XOR } a(g_i) = 1]$. Finally, the state of the regulatole $g_0$ is simply the influence of $g_1$: $f^{g_0}(g_n, \ldots, g_1) = 1 \text{ iff } infl(g_1) = P$.

Even if $g_0$ is regulated by the function $f^{g_0}$, usually, due to experimental noise, not all conditions will manifest $f^{g_0}$. We say that condition $j$ is consistent with $f^{g_0}$ if state($g_0, j$) = $f^{g_0}(g_n, \ldots, g_1)$, where the states of $g_n, \ldots, g_1$ are taken in condition $j$.

The control pattern of $f^{g_0}$ is the binary vector $c_n, \ldots, c_1$. For example, RAARR is the control pattern for a function with $c_5 = c_2 = c_1 = R$ and $c_4 = c_3 = A$. The state pattern of the variables of $f^{g_0}$ is state($g_n, \ldots, g_1$). For example, 10100 corresponds to state($g_5$) = 1, state($g_4$) = 0 etc.

Biological motivation. We present below several biological examples that explain the motivation for defining chain functions. The Trp operon of E. Coli is a classic example (Neidhardt, 1996). If the promoter of the Trp operon is bound by a repressor (TrpR), the expression of the tryptophan-producing enzymes is prevented. The blocking of expression is regulated in the following way: to bind to its promoter DNA, TrpR must have two tryptophan molecules (L-Trp) bound to it. This is an example of negative control, where removal of the ligand switches the Trp operon on. This example corresponds to a chain function with $n = 2$ (see Figure 1A), where $g_0$, the regulatole, is the Trp operon, $g_1$ is TrpR, and $g_2$ is L-Trp. $c_2$, the control bit of the L-Trp, is $A$, since L-Trp activates TrpR. $c_1 = R$, since TrpR represses the transcription of the regulatole. The activity of L-Trp ($g_2$) depends only on its presence. Thus, if L-Trp and TrpR are present (the state pattern is 11), then $a(g_2) = 1$ and thus $infl(g_2) = P$, which implies that $a(g_1) = 1$, and so $infl(g_1) = N$, so we expect no expression of $g_0$. One can compute similarly the expression level for any other state pattern.

Another well known example of a generic regulation switch is galactose utilization in the yeast S. cerevisiae (Jones et al., 1992). This process occurs in a biochemical pathway that converts galactose into glucose-6-phosphate. The transporter gene gal2 encodes a permease that transports galactose into the cell. A group of enzymatic genes, gal1, gal7, gal10, gal5 and gal6, encode the proteins responsible for galactose conversion. The regulators gal4p, gal3p and gal80p control the transporter, the enzymes, and to some extent each other (Xp denotes the protein product of gene X). In the following, we describe the regulatory mechanism, assuming that glucose is absent in the medium. gal4p is a DNA binding factor that activates transcription. In the absence of galactose, gal80p binds gal4p and inhibits its activity. In the presence of galactose in the cell, gal80p binds gal3p. This association releases gal4p, so that gal4p actually activates transcription. This mechanism can be viewed as a chain function, where $(g_4, g_3, g_2, g_1) = (\text{galactose, gal3, gal80, gal4})$, and the corresponding control pattern is ARRA. The known regulatoles are gal1, gal7, gal10, gal5, gal6 and gal2 (see Fig. 1B).

In general, two fundamental mechanisms by which gene regulatory proteins control gene transcription are negative regulation via transcriptional repressors, and positive regulation via transcriptional activators. Inducing ligands can turn a gene ‘on’ by either activating transcriptional activator or repressing transcriptional repressor. Likewise, inhibitory ligands can turn ‘off’ a gene either by inactivating an activator or activating a repressor. These mechanisms are simple cases of chain functions. Examples in Escherichia coli include the lac operon repression by the λ repressor and lactose, araBAD operon activation by araC and arabinose, and the CAP activator in the presence of cAMP (Neidhardt, 1996). More complex regulation functions, such as the signal transduction controlling the SOS
response in E.coli (Neidhardt, 1996), and gene expression during the development of the drosophila’s embryo (Mannervik et al., 1999), might be also viewed as chain functions.

In more complex situations, one simple chain function may not be enough. Some systems should be modeled by several chains combined by boolean operators (e.g. the general amino acids control chain, which operates in conjugation with the arginine specific regulatory chain (Jones et al., 1992)). Several regulators which have the same functionality may be modeled as alternative regulators in a single node along the chain. (e.g. Fus3 and Kss1 in the S. cerevisiae pheromone response). In addition, we might need more levels of discretization. The key concept in chain functions is that activity level of a regulatee is determined by a chain of influences. This concept is not limited to a boolean model (see also concluding remarks). The chain functions as defined here can be used as basic building blocks for modeling more sophisticated regulation systems.

Direct effectors. Genetic networks are frequently represented as wiring diagrams, which show ‘who regulates whom but not how’. The direct effectors of $g_0$ are defined as the minimal set of variables with the property that given any combination of their states, the state of $g_0$ is independent of any other variable. A wiring diagram is a directed graph in which the parents of a regulatee are its direct effectors. It is easy to see that every regulator in a chain function is a direct effector of the regulatee (a proof appears in Appendix A), and no variable outside $g_n, \ldots, g_1$ is a direct effector. An arrow in a chain function diagram reflects influence between regulators which are both direct effectors of $g_0$, and should not be confused with arcs in the wiring diagram, which represent direct transcription effect of the parent on the child.

Chain functions and scores in genetic networks

Chain layers. Any control pattern may be separated into layers, by truncating the control pattern after each $R$. For example, the pattern $ARRARAAA$ has four layers: $l_4 = AR, l_3 = R, l_2 = AR$ and $l_1 = AAA$. The first layer has two possible layer types $A \ldots A$ or $A \ldots AR$, and all other layers must have the type $A \ldots AR$. For brevity, the former will be called type $A$ and the latter type $R$. Note that the number of $A$s in a type $R$ layer may be zero. Define a permutation on a chain function as a reordering of the regulators without changing the control pattern. For example, there are two different permutations for the chain function $f$ with $R_2 = \{x, y\}$ and control pattern $RR$: $(x, y)$ and $(y, x)$. These two permutations yield different functions: If the states of $x$ and $y$ are $0$ and $1$ respectively, then $f(x, y) = 0$ and $f(y, x) = 1$. Similarly, if the control pattern is $RA$, the two permutations yield different functions. However, it is easy to verify that if the control pattern of $f$ is $AA$ or $AR$, the two permutations yield the same function. Thus, if $x$ and $y$ belong to the same layer, they can be permuted without changing the function, and otherwise their permutation yield different functions. This can be generalized as follows: Given a chain function $f_R^n(g_n, \ldots, g_1)$, define a class as a consecutive group of regulators out of $g_n, \ldots, g_1$ that can be arbitrarily permuted while keeping the control pattern, without changing the function. We can show that the layers partition the regulators into a minimal number of classes (see Appendix A). This implies that the order of regulators inside layers is insignificant. Hence, we may focus on the interaction between layers. The incoming influence from the previous layer, and the state of regulators inside the layer, (in fact, the conjunction of their states), determine the outgoing influence of a layer on the next one.

Layers can be interpreted biologically as follows: In case the influence on the downstream elements depends on the cooperation of several factors, this part in the ordered chain constitutes a layer. Prominent examples are transcription factor complexes (e.g. Jones et al. (1992)) and the signal activation cascades (e.g. the MAPK cascade in yeast (Roberts et al., 2000)). As another example, many arginine biosynthetic genes are regulated by arginine specific repression of arg80, arg81 and arg82, which constitute a type $R$ layer (Jones et al., 1992).
The number of chain functions. A trivial upper bound on the number of chain functions of \( n \) variables is \( O(2^n \cdot n!) \). This follows since each control bit can be A or R, and there are \( n! \) possible permutations of the variables. This bound is exponentially smaller than the total number of \( n \)-variable boolean functions, which is \( \Theta(2^{2^n}) \), but it ignores the equivalence classes formed by the layers. In Appendix A, we study the problem of counting the exact number of chain functions of \( n \) variables, and provide the following tight asymptotic bound:

**THEOREM 1.** The number of chain functions with \( n \) control variables is \( \Theta(n! \cdot (\log_2 e)^{n+1}) \).

For example, the total number of boolean functions is 256, 16500, 4.29 \cdot 10^9 and 1.84 \cdot 10^{19} for \( n = 3, 4, 5 \) and 6, respectively. In contrast, the corresponding numbers of chain functions are 26, 150, 1082 and 9366. Thus, the number of chain functions with \( n \) variables is \( \leq |\{ \text{control}\}^n| \cdot (\log_2 e)^n \).

SCORING FUNCTIONS

We assume the regulatee \( g \) is fixed. Our goal is to find the best explanation for the regulation of \( g \), given the expression data. This requires a score, or a scoring function, which evaluates how well a regulation function fits the data. Several scores, including mutual information, rSpec and BDE (see Results for more details) were suggested in previous studies. Here we propose and analyze two new scores: One evaluates a particular set of regulators of \( g \), without attempting to determine the regulation function itself. The other evaluates a particular function for a given set of regulators. The scores are designed to test any regulators set or any candidate regulation function. In particular, the development and use of the scores are completely independent from our study of chain functions.

**Regulators specificity.** We first wish to evaluate the specificity of a set of regulators \( R_n \) to a certain regulatee \( g \). We present here a hypothesis-testing approach to this question.

Let \( M \) be a matrix summarizing the expression data, where rows correspond to the \( r = |V| \) states of \( g \), and columns correspond to the \( c \leq 2^d \) states of \( R_n \) which appear in the data. \( m_{ij} \) is the number of co-occurrences of the \( i \)th state of \( g \) with the \( j \)th state pattern of \( R_n \) in the same condition.

Consider the null hypothesis \( H_0 \) that the state of \( g \) and the regulators’ state pattern are independent. Rejection of \( H_0 \) indicates that the state of \( g \) depends on the regulators’ state pattern, so there is high correlation of the regulators and the regulatee. To test the hypothesis, we use the G-test of independence (Sokal and Rohlf, 1995). The logarithm of the generalized likelihood ratio statistic \( \lambda(M) \) of the above hypothesis is \( \ln \lambda(M) = -\sum_{i=1}^r \sum_{j=1}^c m_{ij} \cdot \log \frac{m_{ij}}{m} + \sum_{j=1}^c m_{ij} \cdot \log \frac{m_{ij}}{m} + \sum_{i=1}^r m_i \cdot \log \frac{m_i}{m} \), where \( m_{ij} = \sum_j m_{ij} \), \( m_i = \sum_j m_{ij} \) and \( m = \sum_{i=1}^r \sum_{j=1}^c m_{ij}. \) A fundamental property of likelihood ratio tests in general is that the asymptotic null distribution of \(-2 \ln \lambda(M)\) is a \( \chi^2 \) distribution, where the parameter space of \( H_0 \) is \( t \)-dimensional and the parameter space of \( H_0 \) is \( t'-dimensional. \) This property is known as the Wilks phenomenon (Wilks, 1938). Accordingly, in our case the asymptotic null distribution of \(-2 \ln \lambda(M)\) is a \( \chi^2_{(c-1) \cdot (r-1)} \) distribution. Therefore, we define \( \log \text{spec} \), the specificity of the set of regulators \( R_n \) for \( g \), as the \( p \)-value that corresponds to the test statistic \(-2 \ln \lambda(M)\), and evaluate it using \( \chi^2_{(c-1) \cdot (r-1)} \).

In \( \lambda(M) \) is proportional to the mutual information between the regulators’ state pattern and the regulatee: \(- \ln \frac{\lambda(M)}{m} \) is precisely \( h(x : y) = H(x, y) - H(x) - H(y) \) (cf. Cover and Thomas (1991)). Mutual information has been used in several studies of genetic networks (e.g. Liang et al. (1998), Pe’er et al. (2002) and Friedman et al. (2000)). \( \log \text{spec} \) has the advantage of assigning a probability to the mutual information expression.

\( \log \text{spec} \) measures the unevenness of the frequencies \( m_{ij} \) for each \( i \). For a fixed number \( k \) of conditions, \( \log \text{spec} \) evaluates the way \( k \) is distributed among the \( c \times r \) cells in \( M \). When \( c \) is large, most cells will contain low frequencies and the unevenness will be low. Hence, \( \log \text{spec} \) has a bias towards small \( c \) values.

The size of matrix \( M \) defined above is bounded by \( 2^{n+1} \) for \( r = 2 \), so by a naive implementation of \( \log \text{spec} \), the total cost of the computation for a given set of \( n \) regulators and the regulatee is \( O(l \cdot n \cdot 2^{n+1}) \). The first part is the cost of building \( M \) and the second, of computing \( -2 \ln \lambda(M) \) and the \( \chi^2 \) approximation. Since typically \( n \ll 20 \), this time is moderate in practice.

The fitness of a regulation function. We now wish to evaluate how well a particular regulation function fits the experimental data. Let \( S \) be a state pattern of the regulators \( R_n \) and let \( f^{g_0} \) be any regulation function. \( f^{g_0} \) determines the expected state of \( g_0 \) for the pattern space \( S \). Given a set of conditions \( E = \{e_1, \ldots e_l\} \), the difference vector \( \Delta \) of a particular combination \( g_0, f^{g_0}, E, R_n \) is: \( \Delta(S) = |\{e_i | state(R_n) = S, f^{g_0}(S) = state(g_0, j)\}| - |\{e_i | state(R_n) = S, f^{g_0}(S) \neq state(g_0, j)\}|. \) Hence, \( \Delta \) counts the number of agreements (consistent cases) minus the number of disagreements in the data with \( f^{g_0} \) for the pattern \( S \). We shall refer to \( \Delta \) of a particular combination \( g_0, f^{g_0}, E, R_n \) without explicitly specifying it. The size of the \( \Delta \) vector is \( c \), the number of different state patterns \( S \).

Denote by \( d_0 \) the number of patterns \( S \) in the data with
\( \Delta(S) = 0 \). If \( e \) other \( \Delta(S) \) values appear, let \( d_1, \ldots, d_e \) be the number of times each of them appear. Now, rank the absolute values of the difference vector and to the rank of each absolute value attach the sign of the difference in \( \Delta \). In case of a tie, rank by midranks, i.e., tied values are ranked by their mean rank. Let us denote the ranks whose signs are negative by \( R_1 < \cdots < R_d \) and those with positive signs by \( S_1 < \cdots < S_k \) so that \( c = a + k + d_0 \).

Consider now testing the hypothesis \( H_0 \) of no difference between the agreement and disagreement frequencies, against the alternative that there are more agreements then disagreements. Thus, rejection of \( H_0 \) is more likely if \( k \) is large and if the positive signed ranks tend to be larger than the negative signed ranks. The Wilcoxon signed rank test (Lehmann and D’abrella, 1975) offers a simple statistic that combines these criteria in the sum of the positive signed ranks \( V_p = S_1 + \cdots + S_k \). \( H_0 \) is rejected where \( V_p \) is sufficiently large. We define \( \text{funcFit} \) as the \( p \)-value that corresponds to the test statistic \( V_p \). The \( p \)-value for \( V_p \) is available in the Wilcoxon standard signed rank table for the null distribution of \( V_p \). Beyond the range of the table, one can use the normal approximation, where the expectation and the variance of \( V_p \) are

\[
\Delta(S) = 0. \quad \text{if } e \text{ other } \Delta(S) \text{ values appear, let } d_1, \ldots, d_e \text{ be the number of times each of them appear. Now, rank the absolute values of the difference vector and to the rank of each absolute value attach the sign of the difference in } \Delta. \quad \text{In case of a tie, rank by midranks, i.e., tied values are ranked by their mean rank. Let us denote the ranks whose signs are negative by } R_1 < \cdots < R_d \text{ and those with positive signs by } S_1 < \cdots < S_k \text{ so that } c = a + k + d_0. \]

\[
\text{Consider now testing the hypothesis } H_0 \text{ of no difference between the agreement and disagreement frequencies, against the alternative that there are more agreements then disagreements. Thus, rejection of } H_0 \text{ is more likely if } k \text{ is large and if the positive signed ranks tend to be larger than the negative signed ranks. The Wilcoxon signed rank test (Lehmann and Dabrella, 1975) offers a simple statistic that combines these criteria in the sum of the positive signed ranks } V_p = S_1 + \cdots + S_k. \quad H_0 \text{ is rejected where } V_p \text{ is sufficiently large. We define } \text{funcFit} \text{ as the } p \text{-value that corresponds to the test statistic } V_p. \quad \text{The } p \text{-value for } V_p \text{ is available in the Wilcoxon standard signed rank table for the null distribution of } V_p. \quad \text{Beyond the range of the table, one can use the normal approximation, where the expectation and the variance of } V_p \text{ are}
\]

\[
\text{The scoring scheme. When we wish to find the best regulatory relation, we can, in principle, find the best regulation set using regSpec, and then use funcFit to find the best function for that set. However, as discussed above, the two scores have different biases to errors, the amount of unevenness in each column of } M, \text{ and } c \text{ value. Hence, using the two scores together and seeking regulatory relations that score high in both is advisable.}
\]

\textbf{RESULTS}

To test our methods, we applied them to the yeast galactose pathway dataset of Ideker et al. (2001). Since high throughput data of protein levels are currently unavailable, we use the mRNA expression levels to model both transcription levels and the abundance of the proteins, assuming that the amount of mRNA presented in the cell is indicative of its protein levels. The dataset contains 23 expression profiles, each corresponding to some perturbation in the galactose pathway. Guided by the current galactose system model, wild-type and nine genetically altered yeast strains were examined, each with a complete deletion of one of the nine galactose pathway genes: gal2\( \Delta \), gal1\( \Delta \), gal5\( \Delta \), gal7\( \Delta \), gal10\( \Delta \), gal3\( \Delta \), gal4\( \Delta \), gal6\( \Delta \) and gal80\( \Delta \). Each of the nine strains was also perturbed environmentally by growth in the presence of galactose (+gal), and in the absence of galactose (-gal).

Additionally, three double perturbations were performed: gal80\( \Delta \)gal2\( \Delta \) -gal, gal80\( \Delta \)gal4\( \Delta \) -gal and gal10\( \Delta \)gal1\( \Delta \) +gal. The reference to all these conditions is the wild-type, grown in +gal media. Ideker et al. computed for each gene and condition the mRNA expression ratio relative to the reference, and assigned to it a confidence value. We transformed the data into binary states as follows: For each gene and condition, if the confidence value was high (above 45 (Ideker et al., 2001)), and the ratio was above 1 (below -1), we set the state value to 1(0). For low confidence values, we assumed the expression level was identical to the wild-type expression, and set the state to 1, since in the reference condition all the galactose system genes are expected to be expressed (in the presence of galactose and absence of glucose (Jones et al., 1992)).

We used as the set of potential regulators gal4, gal3, gal80, gal1, gal2, gal5, gal6, gal7, gal10, gcn1 and galactose. As regulatees, we checked the genes gal1, gal7, gal10, gal2, gal5 and gal6, since their regulation has been well characterized previously (see Figure 1B). For analyzing a regulatee, we do not use data from strains with its complete deletion. We used \( n = 4 \) throughout.

We compared the performance of funcFit to the following alternative scores: (a) rSpec (Tanay and Shamir, 2001), which is essentially minus the logarithm of the \( p \)-value of the number of conditions for which the regulation function is consistent with the observed expression of the regulatee. (b) Mutual information (Cover and Thomas, 1991) between the observed expression level of the regulatee and the expected expression level generated by applying the regulation function on \( R_a \). (Note that it scores a particular regulation function, unlike the mutual information mentioned in the Scoring Functions Section). (c) BDE with the following informative priors: \( N_{ijk} = N_{0.9}^{0.9} \) for consistencies and \( N_{ijk} = N_{0.1}^{0.1} \) for inconsistencies, where \( N = 10 \), and with non-informative
priors where $N' = 1$. See (Heckerman et al., 1995) for definitions and a description of BDE.

Our test was as follows: For each regulatee, we checked each possible subset of $n = 4$ regulators, and for each such subset, we checked every possible regulation function, and found the best scoring one. We performed this test twice, once using all boolean functions, and once using only chain functions. We repeated this test with the scoring functions BDE, mutual information, rSpec and funcFit. In all tests, we did not allow auto-regulation, and each regulator was allowed to appear only once along the chain function. Testing was done using a C++ software implementation written in-house. It can analyze all chain (boolean) functions with $|U| = 11$ and $n = 4$, for a single regulatee, in 30(15) seconds on a standard 800MHz PC.

In Figure 2, we present the performance of the different scoring methods. For each candidate regulation set, all chain functions are scored and the best score is presented. As can be seen in the figure, mutual information and rSpec tend to score high a large portion of the regulator sets. Thus, occasionally they may infer a lot of false positive regulation functions. Moreover, a small difference in the consistency level can cause a regulation function to be ranked very high or very low. In mutual information, there are 1, 1 and 69 regulator sets whose best chain functions are in the highest score category, for gal1, gal7 and gal10 respectively. In rSpec, there are 1, 1 and 44 chain functions in the highest scores category, for the same regulatees. The main reason for this instability is that both scores take into consideration only the total number of consistent conditions, without considering their state patterns at all. Unlike these scores, BDE and funcFit consider the distribution of inconsistency among the state patterns. Moreover, funcFit and BDE have a Gaussian-like distribution of scores, which is preferable, since we always get only a few top scoring candidates. Nevertheless, BDE does not always rank the real chain function high: In BDE, there are 26, 76 and 7 chain functions above the real solution, for the three regulatees. Since BDE penalizes state patterns with noise, but does not penalize for missing state patterns, it has a bias to small $c$ values. This may explain its poorer performance in comparison with funcFit. funcFit is the only score which consistently ranks the real solution high: there are only 7, 8 and 7 chain functions equal to or above the real solution, for the same three regulatees. Qualitatively similar results were obtained when repeating the same analysis with all boolean functions, and with the BDE score using different $N'$ values as well as non-informative priors.

Our next test aimed to see the effect of restricting inference to chain functions only. In Figure 3 we present the maximum funcFit scores distribution for the chain functions set and for all boolean functions. As expected, when using all boolean functions, the distribution tends to spread to higher values. Moreover, by using chain functions only, the real solution is ranked higher: In gal10, there are 16 boolean functions and only 8 chain functions with scores equal to or above the real one. In gal1 and gal7, the corresponding numbers are 16 and 7. Qualitatively similar results were obtained using the other scoring methods. In principle, when using all boolean functions, the distribution may tend to spread much more drastically to high values. However, the specific dataset that we analyzed was not large enough to manifest this difference: Although there are theoretically 65,536 boolean functions and 150 chain functions, actually only 200 boolean functions and 40 chain functions are effectively different (on average), because on average only 7.5 different state patterns appear in the data (out of 16 possible ones) for each group of regulators. In larger datasets with more state patterns, the advantage of the chain functions should be more pronounced.
Chain functions and scores in genetic networks

CONCLUDING REMARKS

In this paper, we propose a biologically relevant class of regulation functions. We also suggest two scoring methods by which one can evaluate candidate regulatory relations, and demonstrate their advantage over extant scores. We tested our method on experimental gene expression data, in trying to infer gene regulation relations. We showed the utility of using chain functions, and the advantage of our scores over several extant methods.

Clearly, more extensive testing of our methods on additional datasets and pathways is needed. By tests on large datasets we expect to demonstrate the fuller advantage of using a restricted set of relevant regulation functions. We expect to identify more regulation functions and refine our results by allowing more than two levels of discretization and assigning a probability distribution over those levels. In addition, we expect that the special structure of chain functions can be exploited in the design of follow-up experiments.

The ability to score and restrict regulatory relations are fundamental components in the grand challenge of reconstructing regulatory networks. In order to extend this
work towards global network reconstruction, the chain function model should be extended. It should allow several chain functions combined by a boolean operator. Handling functions with unknown number of regulators should be addressed. Cases where there are several regulators whose regulation chains have common parts, should also be considered.

ACKNOWLEDGEMENT

We thank Noga Alon for pointing us to the literature on Ordered Bell numbers. We thank Amos Tanay and Ori Gurel for helpful discussions. This study was supported by a pilot grant from the McDonnell foundation and by the Israeli Science Foundation (grant no. 309/02).

REFERENCES


APPENDIX A: PROPERTIES OF CHAIN FUNCTIONS

In this section, we prove some properties of chain functions. We study the problem of counting the exact
number of chain functions with \( n \) variables, and provide a tight asymptotic bound. We use the same terminology as in the Chain Functions Section.

**Lemma 1.** Every regulator in a chain function is a direct effector of the regulatee.

**Proof.** To show that \( g_i \) is a direct effector of \( g_0 \), consider a combination of states for \( g_0 \ldots g_{i+1} \) which creates a positive influence on \( g_i \) (for example, all the variables with \( A(R) \) control bit have the state \( 1(0) \)). If the states of \( g_{i-1}, \ldots, g_1 \) are all \( 1 \), then the state of \( g_0 \) is dependent on the state of \( g_i \), and thus \( g_i \) is a direct effector of \( g_0 \).

**Lemma 2.** The layers partition the regulators into a minimal number of classes.

**Proof.** We shall show first that the regulators in a layer form a class. Then, we shall show that a successive pair of regulators with the control pattern \( RA \) or \( RR \) must be in different classes, and thus we must truncate the classes after each \( R \).

We start with the first claim. A consecutive pair of regulators inside a layer always has the pattern \( AA \) or \( AR \). Exchanging the order of the two regulators might influence the state of \( g_0 \) only if the two regulators have different states: in the \( AA \) control pattern, the state patterns 10 and 01 both yield negative influence, irrespective of the previous influences. Likewise, in the \( AR \) control pattern, the state patterns 10 and 01 both yield positive influence. Thus, any two consecutive regulators inside a layer are exchangeable. Therefore, any permutation of regulators in a layer might be reached by a series of successive pair exchanges without changing the function.

Next, we show the second claim. In the \( RA \) control pattern, the state pattern 10 yields negative influence while the state pattern 01 yields positive influence. Likewise, in the \( RR \) control pattern, the state pattern 10 yields positive influence while the state pattern 01 yields negative influence. Thus, such pairs are unexchangeable.

In the rest of the appendix, we study the problem of chain functions counting. Define the composition of a layer as the subset of regulators out of \( g_n, \ldots, g_1 \), which correspond to the layer.

**Lemma 3.** A chain function is uniquely determined by the sizes, order and composition of its layers, and the type of pattern in the first layer.

**Proof.** To prove the lemma, we show that any change in the number, order or composition of the layers, or the type of the first layer, is not function preserving. First, we prove that different types of the first layers cannot correspond to the same function: Given the state pattern \( 000 \ldots 0 \) for all regulators, if the first layer is of type \( A \), it has negative influence and the state of the regulatee is 0. If it is of type \( R \), that state is 1, so the function value is changed.

Next we prove that any change in the number, order or composition of layers is not function preserving. Let \( f^R \) and \( f^A \) be two chain functions whose number, order or composition of layers is different. The layers of \( f^R \) are denoted by \( l_0', \ldots, l_1' \), and the layers of \( f^A \) are denoted by \( l_0, \ldots, l_1 \). We denote by \( l_x \) and \( l'_x \) the first (least indexed) layers whose composition differs, so that the layers \( l_{x-1}', \ldots, l_1' \) are identical to the layers \( l_{x-1}, \ldots, l_1 \), and \( l_x' \) is different from \( l_x \). Such layers \( l_x \) and \( l_x' \) exist by our assumptions. Suppose, w.l.o.g., that \( l_x' \) contains a variable \( v \) which is not included in \( l_x \). Assume that \( l_x \) and \( l_x' \) have the same type \( R \) (The proof for the other layer type is similar). Consider the following state pattern: All variables in layers \( l_1, \ldots, l_1 \) are in state 1 and all the rest (including variables that appear in only one of the functions) are in state 0. Thus, \( l_x \) is positively influenced by layer \( l_x+1 \) and is negatively influencing its successor. However, using the same state pattern, \( l_x' \) contains the variable \( v \) which has state 0. Thus, \( l_x' \) has positive influence on \( l_{x-1}' \). Since layers \( l_{x-1}', \ldots, l_1' \) have the same composition as \( l_{x-1}, \ldots, l_1 \) and the state pattern in both is all 1-s, the final function value is changed.

We now count the total number of chain functions with \( n \) variables. Let \( S_k^n \) be the number of partitions of \( n \) variables into exactly \( k \) nonempty sets. \( S_k^n \) may be computed recursively by the formula \( S_k^n = kS_k^{n-1} + S_{k-1}^{n-1} \), where \( S_1^n = 1 \), \( S_0^n = 0 \) and \( S_k^0 = 0 \) for \( y > x \). In each step we add a variable to one of the \( k \) existing sets, or we put the variable in the new set. Thus, the number of partitions of \( n \) variables into any number of ordered nonempty sets is \( b(n) = \sum_{k=1}^{n} k! \cdot S_k^n \), which is known as an ordered Bell number, which is asymptotically \((1 + O(1)) \cdot \frac{n^e}{e^n} \cdot (\log_2 e)^{n+1} \) (Wilf, 1994, p. 175–176). For each partition of the variables, there are two possible types of first layer. Thus we conclude:

**Theorem 4.** The number of chain functions with \( n \) control variables is \( 2 \cdot b(n) \).

Hence the number is \( \Theta(n!) \cdot (\log_2 e)^{n+1} \). For example, for \( n = 2 \), there are \( 2 \cdot b(2) = 6 \) different functions. Indicating the chain functions as \( f_{R_2,	ext{c}2}(g_2, g_1) \), these are \( f_{R,R}(x, y) \), \( f_{R,R}(x, y) \), \( f_{R,A}(x, y) \), \( f_{R,A}(x, y) \), \( f_{A,A}(x, y) \) (equivalent to \( f_{A,A}(x, y) \), since \( AA \) is one layer), and \( f_{A,R}(x, y) \) (equivalent to \( f_{A,R}(x, y) \), since \( AR \) is one layer).