Reconstructing Chain Functions in Genetic Networks

I. Gat-Viks, R. Shamir, R.M. Karp, and R. Sharan

Pacific Symposium on Biocomputing 9:498-509(2004)

RECONSTRUCTING CHAIN FUNCTIONS IN GENETIC NETWORKS

I. GAT-VIKS, R. SHAMIR
School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel.
{*iritg,rshamir*}*@tau.ac.il.*

R. M. KARP, R. SHARAN
International Computer Science Institute, 1947 Center St., Berkeley CA 94704.
{*karp,roded*}*@icsi.berkeley.edu.*

Abstract

Deciphering the mechanisms that control gene expression in the cell is a fundamental question in molecular biology. This task is complicated by the large number of possible regulation relations in the cell, and the relatively small amount of available experimental data. Recently, a new class of regulation functions called *chain functions* was suggested. Many signal transduction pathways can be accurately modeled by chain functions, and the restriction to chain functions greatly reduces the vast search space of regulation relations. In this paper we study the computational problem of reconstructing a chain function using a minimum number of experiments, in each of which only few genes are perturbed. We give optimal reconstruction schemes for several scenarios and show their application in reconstructing the regulation of galactose utilization in yeast.

1 Introduction

The regulation of mRNA transcription is key to cellular function. High throughput genomic technologies, such as DNA microarrays, enable a global view of the transcriptome, and provide the means to reconstructing *regulatory relations* among genes, that is, inferring the set of genes that cooperate in the regulation of a given gene and the particular logical function by which this regulation is determined. This paper studies the number and complexity of biological experiments that are needed in order to infer certain regulatory relations.

An *experiment* involves knocking out or over-expressing certain genes, and measuring the expression levels of all other genes. The *order* of an experiment is the number of genes that are perturbed. 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. Akutsu et al. [1] showed that even for a boolean

network model, the number of experiments that are needed for reconstructing a network of N genes is prohibitive: The lower and upper bounds on the number of experiments of order $N-1$ that are needed, are $\Omega(2^{N-1})$ and $O(N \cdot 2^{N-1})$, respectively. Even with no more than d regulators for each regulated gene, the number of required experiments of order d is still $\Omega(N^d)$ and $O(N^{2d})$, respectively [1].

The inherent complexity of genetic network inference led researchers to seek ways around this problem. Ideker et al. [2] studied how to dynamically design experiments so as to maximize the amount of information extracted. Friedman et al. [3] used Bayesian networks to reveal parts of the genetic network that are strongly supported by the data. Tanay and Shamir [4] suggested a method of expanding a known network core using expression data. Several studies used prior knowledge about the network structure, or restrictive models of the structure, in order to identify relevant processes in gene expression data [5,6,7,8].

Recently, a biologically motivated model of regulation relations based on *chain functions*, was suggested in order to cope with the problem of genetic network inference [9]. In a chain function, the state of the regulated gene 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 defer formal definitions till later). The chain model further assumes that variable states are boolean. The latter assumption is a drastic simplification of real biology, yet it captures important features of biological systems and was frequently used in previous studies [2]. The class of chain functions has several important advantages [9]: These functions reflect common biological regulation behavior, so many real biological regulatory relations can be elucidated using them (examples include the SOS response mechanism in *E. coli* [10] and galactose utilization in yeast [11]). Moreover, by restricting consideration to chain functions, the number of candidate functions drops from double exponential to single exponential only.

In this paper we study the computational problems arising when wishing to reconstruct chain functions using a minimum number of experiments of the smallest possible order. We address both the question of finding the set of regulators of a chain function, which is typically much smaller than the entire set of genes, and the question of reconstructing the function given its regulators. We give optimal reconstruction schemes for several scenarios and show their application on real data. Our analysis focuses on the theoretical complexity of reconstructing regulation relations (number and order of experiments), assuming that experiments provide accurate results, and that the target function can be studied in isolation from the rest of the genetic network.

The paper is organized as follows: Section 2 contains basic definitions related to chain functions. In Section 3 we give worst-case and average-

case analyses of the number of experiments needed in order to reconstruct a chain function. Both low-order and high-order experimental settings are considered. In Section 4 we study the reconstruction of composite regulation functions that combine several chains. Finally, in Section 5 we describe a biological application of our analysis to reconstruct the regulation mechanism of galactose utilization in yeast. For lack of space, some proofs are shortened or omitted.

2 Chain Functions

Chain functions were introduced by Gat-Viks and Shamir [9]. In the following we define these functions and describe their main properties. Our presentation differs from the original one, to allow succinct description of the reconstruction schemes in later sections.

Let U denote the set of all variables in a network, where $|U| = N$. These variables correspond to genes, mRNAs, proteins or metabolites. Each variable may attain one of two *states*: 1 or 0. The state of gene g, denoted by $state(g)$, indicates the discretized expression level of the gene. The intended interpretation is that $state(g)$ is 1 if gene g is capable of being activated in a given environment, and 0 otherwise. A variable normally exists in its *wild-type* state, but perturbations such as gene knockouts may change its state. Let $g_0 \in U$ be regulated by a set S of n variables. In that case we say that S is the *regulator set* of g_0, and g_0 is called the *regulatee*. A candidate regulation function for the regulatee g_0 has the form $f^{g_0} : \{0,1\}^n \rightarrow \{0,1\}$. In other words, the state of g_0 is a function of the states of its regulators.

The chain function model assumes that the functional relations are deterministic. The chain function f^{g_0} on the *regulators* $g_n, ..., g_1$ determines the state of the regulatee g_0. The order of the regulators is important, as it reflects the order of influence among them. We call g_i the *predecessor* of g_j for $i > j$, and the *successor* of g_k for $i < k$. Each regulator may activate or repress its successor, and this chain of events enables a signal to propagate from g_n to g_0, in a manner described below.

Associated with each regulator g_i is a fixed value y_i which dictates the regulatory influence of g_i on g_{i-1}. If $y_i = 0$ then g_i is an *activator*; otherwise g_i is a *repressor*. The value y_i represents an intrinsic property of the chain and is not subject to change. The *control pattern* of f^{g_0} is the binary vector $(y_n, ..., y_1)$. The function f^{g_0} can be defined using two n-long boolean vectors attributing activity and influence to each g_i. The definitions of the *activity* and *influence* are recursive. Let $a(g_i)$ denote the activity of g_i, and $infl(g_i)$ denote the influence of g_i on g_{i-1}. The influence on g_n is always 1. g_i is activated ($a(g_i) = 1$) iff it is capable of being activated and it receives a positive activation signal from its predecessor. The activation signal $infl(g_i)$, transmitted from g_i to g_{i-1}

is 1 if g_i is an activator and is itself activated, or if g_i is a repressor and is not activated (so that it fails to repress g_{i-1}). Formally,

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

$$infl(g_i) = y_i \oplus a(g_i) \quad (2)$$

Finally, the state of the regulatee g_0 is simply the influence of g_1. We define the *output* of f^{g_0} to be $state(g_0)$. A chain function is uniquely determined by its set of regulators, their order and the control pattern.

Any control pattern may be separated into *blocks* of consecutive regulators by truncating the control pattern after each 1. The first block (rightmost, ending at g_1) has two possible forms: $0 \ldots 0$ or $0 \ldots 01$. All other blocks are of the form $0 \ldots 01$.

3 Reconstruction of Chain Functions

In this section we study the question of uniquely determining the chain function which operates on a known regulatee, using a minimum number of experiments. We assume throughout that all variable states in wild-type are known (or, else, these could be measured). We further assume that all regulator states in wild type are 1, except possibly g_n. The latter assumption is motivated by the observation that in many biological examples, all regulators are expressed in wild type and the state of the regulatee is determined by the presence or absence of a metabolite g_n. (Examples include the Trp, lac and araBAD operons in *E. Coli* [10], and the regulation of galactose utilization in yeast [11]. See Section 6 for a discussion of the situation when this assumption does not hold.)

An *experiment* is defined by a set of variables that are externally perturbed (knocked-out or over-expressed). The states of the perturbed variables are thus fixed, and the states of all non-perturbed regulators are assumed to remain at the wild-type values, with the exception of the regulatee. Its state is determined by the chain function. The *order* of an experiment is the number of externally perturbed variables in it.

Our reconstruction algorithms are based on performing various experiments and observing their influence on the state of the regulatee. The algorithms implicitly assume that the regulation function is indeed a chain function and do not explicitly test this property.

We now devise a simple set of equations that characterize the output of a chain function as a function of the control pattern and the states of the regulators, both in the wild-type state and in states produced by perturbing some regulators. These equations are the foundation of all the subsequent reconstruction schemes:

Proposition 1 *Let f be a chain function on $g_n, \ldots, g_1$. If $state(g_i) = 1$ for $1 \leq i < n$ then $state(g_0) = state(g_n) \oplus (\oplus_{i=1}^{n} y_i)$. For any other*

state vector, if the least index of a state-0 regulator is $j \leq n$ then $f^{g_0}(g_n, ..., g_1) = \oplus_{i=1}^{j} y_i$.

Proof: By definition, $a(g_n) = state(g_n)$. For $i < n$, $state(g_i) = 1$ implies that $a(g_i) = a(g_{i+1}) \oplus y_{i+1}$. It follows by induction that $state(g_0) = state(g_n) \oplus (\oplus_{i=1}^{n} y_i)$. Similarly, if $state(g_j) = 0$ and $state(g_i) = 1$ for all $i < j$, it follows by induction that $f^{g_0}(g_n, ..., g_1) = \oplus_{i=1}^{j} y_i$. ■

3.1 Types and Blocks

A *perturbation* is an experiment that changes the state of a variable to the opposite of its state in wild-type. By our assumption on the regulator states in wild-type, the perturbation of a regulator in $\{g_{n-1}, ..., g_1\}$ is a knockout. For $S \subseteq U$, an *S-perturbation* is an experiment in which the states of all the variables in S are perturbed.

Let w be $state(g_0)$ in wild-type. Let $\bar{w}$ be the opposite state. For the reconstruction, we first classify the variables in U into two *types*: W and $\bar{W}$. A variable is in W ($\bar{W}$) if its perturbation produces output w ($\bar{w}$). Naturally, the majority of the genes have type W, since in particular all the genes that are not part of the chain function are such. By Proposition 1 we have $g_n \in \bar{W}$. We call a gene that belongs to W ($\bar{W}$) a *W-gene* (*$\bar{W}$-gene*). *W-successor*, *$\bar{W}$-successor* of a gene and *W-regulator* , *$\bar{W}$-regulator* are similarly defined.

The type of a single gene can be determined by a single perturbation of the gene. Such an experiment will be referred as a *typing experiment* throughout.

Corollary 2 *Given an ordered set of regulators $g_n, ..., g_1$, their control pattern can be reconstructed using n typing experiments.*

Consider now the block partition of the regulators. The right boundary of a block corresponds to a regulator g_j with $y_j = 1$ (unless $j = 1$, in which case $y_1 = 0$ is also possible), and any other regulator g_i in the block has $y_i = 0$.

Lemma 3 *Each block contains regulators of a single type, and two adjacent blocks contain regulators of opposite types.*

The proof follows from the fact that the type of g_i differs from the type of g_{i-1} iff $y_i = 1$. Thus, we can refer to a block as either a *W-block* or a *$\bar{W}$-block*, and the two types of blocks alternate.

3.2 Reconstructing the Regulator Set and the Function

Consider a chain function with control pattern $(y_n, ..., y_1)$ and let $g_j, \ldots, g_i$ be a block. Then $infl(g_i) = [infl(g_{j+1}) \wedge (\bigwedge_{h=i}^{j} state(g_h))] \oplus y_i$. Thus,

the behavior of the chain is determined by the boolean variable $infl(g_{j+1})$, by the control pattern, and by the conjunction of the states of its regulators. Since this conjunction is independent of the order of occurrence of these genes, no experiment based on perturbing the states of the genes can determine the order of the genes within the block. In view of this limitation, our goal is to reconstruct the control pattern, the set of genes within each block (but not the order of their occurrence) and the ordering of the blocks. Correspondingly, in the following we will use the term *successor* of a gene to denote a regulator that succeeds that gene in the chain and is not a member of its block. For convenience, we shall refer to W-genes that are not regulators of g_0 as predecessors of g_n.

The above discussion implies that once we have typed each gene, it remains to determine, for each pair consisting of a W-gene and a $\bar{W}$-gene, which of these genes precedes the other in the chain. Let k_W,$k_{\bar{W}}$ denote the number of regulators of types W,$\bar{W}$, respectively. Note that $k_W + k_{\bar{W}} = n \leq N$, and in fact, typically, $n \ll N$, as $k_W \ll |W|$.

Suppose we perform a $\{i,k\}$-perturbation with $g_i \in W$ and $g_k \in \bar{W}$. If the result is w, then g_k precedes g_i. Otherwise, g_i precedes g_k. A 2-order experiment for determining the relative order of a W-gene and a $\bar{W}$-gene will be called a *comparison* throughout.

Proposition 4 *Given the set of regulators of a chain function and their types, $k_W k_{\bar{W}}$ comparisons are necessary and sufficient to reconstruct the function.*

Proof: The upper bound follows by comparing every W-regulator with every $\bar{W}$-regulator. The lower bound follows from the fact that, in the special case where every W-regulator precedes every $\bar{W}$-regulator, no set of comparisons can determine the relative order of a given pair consisting of a W-regulator and a $\bar{W}$-regulator, unless it includes a direct comparison between the pair. Therefore, all such comparisons must be performed. ■

We now turn to the question of reconstructing a chain function without prior knowledge of the identity of its regulators. The discussion above suggests a way to solve the problem: First, we find the gene types using N typing experiments. Next, we reconstruct the block structure by performing all possible comparisons between a W-gene and a $\bar{W}$-gene.

A more efficient reconstruction is possible when g_n is known. This is common in functions in which g_n stimulates the response. If g_n is known, then, since $g_n \in \bar{W}$, all W-regulators can be identified by comparing every W-gene with g_n, for a total of $N - k_{\bar{W}}$ comparisons. Since any $\bar{W}$-gene is a regulator, these experiments are sufficient to identify all the regulators, and we can apply Proposition 4 to complete the reconstruction.

Proposition 5 *A chain function can be reconstructed using at most N typing experiments and $k_{\bar{W}} \times (N - k_{\bar{W}})$ comparisons. Given g_n, a chain function can be reconstructed using at most $N - 1$ typing experiments and $N - n + k_W k_{\bar{W}}$ comparisons.*

Propositions 4 and 5 were a worst case analysis. Next, we describe another reconstruction algorithm, whose *expected* number of required experiments is lower. The algorithm is based on identifying g_n efficiently and using it for the reconstruction. Denote by D_g the set of W-successors of $g \in \bar{W}$ in f.

Proposition 6 *A chain function can be reconstructed using N typing experiments and an expected number of $O(N \log k_{\bar{W}} + k_W k_{\bar{W}})$ comparisons.*

Proof: **Algorithm:** We perform N typing experiments. Next, we apply a randomized scheme to identify g_n and reconstruct the chain: Each time we pick a gene $g \in \bar{W}$ at random, find its successors and their order, and remove g and all its successors from further consideration. We stop when no $\bar{W}$ genes are left, identifying g_n as the last picked gene. In order to find the successors of g, we first identify the members of D_g using at most $N - k_{\bar{W}}$ comparisons. Using D_g, we then reconstruct the part of the chain that spans g and its successors by at most $|D_g|(k_{\bar{W}} - 1)$ comparisons, as in Proposition 4.

Complexity: The set of comparisons can be divided into two parts: Those that are required to identify the sets D_g, and those required to reconstruct the chain parts induced by these sets. For the latter, $k_W k_{\bar{W}}$ comparisons are needed in total, since every pair consisting of a W-regulator and a $\bar{W}$-regulator is compared exactly once. Thus, it suffices to compute the expectation of the first part. Let $T(x)$ be this expectation, given that the current $\bar{W}$ set contains x elements, where $T(0) = 0$. Then $T(x) \leq \frac{1}{x} \sum_{q=1}^{x} (N + T(x - q))$ for $x \geq 1$. By induction $T(x) \leq 2N \log x + N$. Substituting $x = k_{\bar{W}}$ we obtain the required bound. ■

3.3 Using High-Order Experiments

In this section we show how to improve the above results when using experiments of order $q > 2$. The results in this section are mainly of theoretical interest, since high-order experiments may not be practical.

Proposition 7 *Given the set of n regulators of a chain function, the function can be reconstructed using $O(n + \frac{n^2 \log q}{q})$ experiments of order at most q. This is optimal up to constant factors for $q = \Theta(n)$.*

Proof: The number of possible chain functions with n regulators is $\Theta((\log_2 e)^{n+1} n!)$ [9]. Since each experiment provides one bit of information, the information lower bound is $\Omega(n \log n)$ experiments.

We give the upper bound proof for $q = n$. The proof for other values of q follows by appropriately choosing subsets of regulators of cardinality q, and reconstructing their sub-chains using the method we give next, thereby inferring the entire chain.

Let n_i be the number of regulators in block i, where blocks are indexed in right-to-left order. Our reconstruction algorithm is as follows: First, we perform n typing experiments. Next, we identify the type of the first block using one experiment of order n, in which all regulators are perturbed. We proceed to reconstruct the blocks one by one, according to their order along the chain. Note that the type of each block is now known, since the two types alternate. Suppose we have already reconstructed blocks $1, \ldots, i-1$. For reconstructing the i-th block we only consider the set of regulators that do not belong to the first $i-1$ blocks. Out of this set, let A be the subset of regulators that have the same type as block i, and let B be the the subset of regulators of the opposite type. We use standard binary search on the set A to identify the members of the i-th block, including in the perturbations also all regulators in B. This requires $O(n_i \log n)$ experiments. Thus, altogether we perform $O(n \log n)$ experiments. ■

4 Combining Several Chains

In this section we extend the notion of a chain function to cover common biological examples in which the regulatee state is a boolean function of several chains. Frequently, a combination of several signals influences the transcription of a single regulatee via several pathways that carry these signals to the nucleus, and a regulation function that combines them together. Here, we formalize this situation by modeling each signal transduction pathway by a chain function, and letting the outputs of these paths enter a boolean gate.

Define a *k-chain function f* as a boolean function which is composed of k chain functions over disjoint sets of regulators, that enter a boolean gate $G(f)$. Let f^i be the i-th chain function and let g^i_j denote the j-th regulator in f^i. The output of the function is $G(infl(g^1_1), \ldots, infl(g^k_1))$.

In the following we present several biological examples for k-chain functions that arise in transcriptional regulation in different organisms: The lac operon [10] codes for lactose utilization enzymes in *E. Coli.* It is under both negative and positive transcriptional control. In the absence of lactose, lac-repressor protein binds to the promoter of the lac operon and inhibits transcription. In the absence of glucose, the level of cAMP

in the cell rises, which leads to the activation of CAP, which in turn promotes transcription of the lac operon. In our formalism, the lac operon is controlled by a 2-chain function with an AND gate. The chains are: $f^1(g_2^1, g_1^1) = f^1$(lactose, lac-repressor), with control pattern 11, and $f^2(g_3^2, g_2^2, g_1^2) = f^2$(glucose, cAMP, CAP), with control pattern 100. Other examples of 2-chains with AND gates are the regulation of arginine metabolism and galactose utilization in yeast [11]. A 2-chain with an OR gate regulates lysine biosynthesis pathway enzymes in yeast [11].

These examples motivate us to restrict attention to gates that are either OR or AND. We first show that we can distinguish between OR and AND gates. We then show how to reconstruct k-chain functions in the case of OR and later extend our method to handle AND gates.

Denote the output of f^i by O_i. If $O_i = 1$ in wild-type, we call f^i a *1-chain* and, otherwise, a *0-chain*. A regulator g_j^i is called a 0-regulator (1-regulator) if its perturbation produces $O_i = 0$ ($O_i = 1$). Let k_0 (k_1) be the number of 0-regulators (1-regulators) in f. A block is called a 0-block (1-block), if it consists of 0-regulators (1-regulators).

Lemma 8 *Given a k-chain function f with gate $G(f)$ which is either AND or OR, $k \geq 2$, we can determine, using $O(N^2)$ experiments of order at most 2, if $G(f)$ is an AND gate or an OR gate.*

Proof: We perform N typing experiments. If $w = 0$ and $\bar{W} = \emptyset$ then $G(f)$ is an AND gate. If $w = 1$ and $\bar{W} = \emptyset$ then $G(f)$ is an OR gate. Otherwise, $\bar{W} \neq \emptyset$. In this situation the cases of $w = 0$ and $w = 1$ are similarly analyzed. We describe only the former.

If $w = 0$ we have to differentiate between the case of an OR gate, whose inputs are all 0-chains, and the case of an AND gate, whose inputs are one 0-chain and $(k-1)$ 1-chains. To this end we perform all comparisons of a W-gene and a $\bar{W}$-gene. Let T be the set of genes g such that the result of a $\{g, g'\}$-perturbation is w for every $g' \in \bar{W}$. Then $T \neq \emptyset$ iff $G(f)$ is an AND gate. ■

We now study the reconstruction of an OR gate. Let S be the (possibly empty) set of regulators that reside in one of the first blocks (i.e., blocks containing g_1^i), that are also 1-blocks. We observe that a perturbation of any regulator in S results in $state(g_0) = 1$ regardless of any other simultaneous perturbations we may perform. Hence, our reconstruction will be unique up to the ordering within blocks and the assignments of the regulators in S to their chains. The next lemma handles the case $w = 0$. The subsequent lemma treats the case $w = 1$.

Lemma 9 *Given a k-chain function f with an OR gate and assuming that $w = 0$, we can reconstruct f using N typing experiments and $(N - k_1)k_1$ comparisons.*

Proof: We perform N typing experiments. Then, for each 1-regulator b, we perform all possible comparisons, thereby identifying all 0-regulators that succeed b in its chain. This completes the reconstruction. ■

Lemma 10 *Let f be a k-chain function with an OR gate. Assume that $w = 1$, and let r be the number of 1-chains entering the OR gate. Then f can be reconstructed using $O(N^r + Nk_0{}^r)$ experiments of order at most $\min\{k+1, r+2\}$.*

Proof: First, we determine r, the minimum order of an experiment that will produce output 0 for f. For successive values i we perform all possible i-order experiments; r is determined as the smallest i for which we obtain output 0. In total we perform $O(N^r)$ experiments. We call the set of perturbed genes in an r-order experiment which results in output 0, a *reset combination.*

Next, we identify all 1-regulators. This is done by performing $O(Nk_0{}^r)$ experiments of order $(r+1)$ as follows: For each reset combination discovered, we perturb in addition each other gene, one at a time, and record those that produce output 1 as 1-regulators. Each reset combination identifies a set of 1-regulators. These sets form a partial order under set inclusion. Let M be a reset combination corresponding to a minimal set in the partial order of 1-regulator sets. The genes in this minimal set will be exactly the 1-regulators in the 0-chains and the 1-regulators in S. By perturbing all r regulators in M, we deactivate the 1-chains, thereby reducing the problem of reconstructing the 0-chains to that of reconstructing a $(k-r)$-chain function with an OR gate and $w = 0$. This is done by applying the reconstruction method of Lemma 9 using experiments of order at most $\min\{k+1, r+2\}$. The assignment of 1-regulators in S will remain uncertain.

The 1-chains can be now computationally inferred as follows: Pick an arbitrary reset combination and consider in turn each of its subsets of cardinality $r-1$. Fixing a subset, consider all reset combinations that contain it. The variable 0-regulators in these combinations correspond to the 0-regulators of a particular 1-chain. For each of these variable 0-regulators our experiments determine a set consisting of the 1-regulators in its chain that succeed it, plus the 1-regulators in S and in the 0-chains, which have been identified by the reset combination M, and can be removed from consideration. Performing this computation for all combinations and subsets, we will have determined, for each 1-chain, its 0-regulators, its 1-regulators and the ordering relations between them. ■

Note that for $k = 1$ the above algorithms will reconstruct a single chain. Further note that these algorithms may be used for the reconstruction of an AND gate as well, exchanging the roles of 0 and 1 in the above description. This gives rise to the following result:

Theorem 11 *A k-chain function with an OR or an AND gate can be reconstructed using $O(N^k)$ experiments of order at most $k+1$.*

5 A Biological Application

The methods we presented above can be applied to reconstruct chain functions from biological data. We describe in detail one such reconstruction of the yeast galactose chain function, for which some of the required perturbations have been performed. We show that one additional experiment suffices to fully reconstruct the regulation function.

The galactose utilization in the yeast *S. cerevisiae* [11] occurs in a biochemical pathway that converts galactose into glucose-6-phosphate. The transporter gene gal2 encodes a protein 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, promoting transcription. This mechanism can be viewed as a chain function, where $f^1(g_4, g_3, g_2, g_1) = f^1(galactose, gal3, gal80, gal4)$, and the corresponding control pattern is 0110. The gal7, gal10 and gal1 regulatees are also negatively controlled by another chain f^2 containing MIG1 and glucose. The two chains are combined by an AND gate. We focus here on the reconstruction of f^1, since the other chain has no influence in the experiments that we describe below (as those were conducted in the presence of glucose). f^1 consists of 3 blocks, where in wild-type (in the presence of glucose and galactose) gal3, gal80 and gal4 are in state 1 (using the same discretization procedure employed by Ideker et al. [9]).

Assuming we know the group of four regulators, we need according to Proposition 4 a total of 4 typing experiments and 3 comparisons (since only gal80 is of type W) to reconstruct the chain. Notably, all 4 typings and 2 of the 3 comparisons were performed by Ideker et al. [12], yielding the correct results. The missing experiment is a comparison of gal80 and gal3. A correct result of this experiment will lead to full reconstruction of the chain function.

6 Concluding Remarks

In this paper we studied the computational problems arising when wishing to reconstruct regulation relations using a minimum number of ex-

periments, assuming that the experiments provide correct results. We restricted attention to common biological relations, called chain functions, and exploited their special structure in the reconstruction. We also suggested an extension of that model, that combines several chain functions, and studied the implied reconstruction questions. On the practical side, we have shown an application of our reconstruction scheme for inferring the regulation of galactose utilization in yeast.

The task of designing optimal experimental settings is fundamental in meeting the great challenge of regulatory network reconstruction. While this task entails coping with complex interacting regulation functions, we chose here to focus on the reconstruction of a single regulation relation of a single regulatee. We also made two strong assumptions that simplify the analysis considerably: (1) The function can be studied in isolation. Hence, upon any perturbation, none of the other regulators change their states; (2) the wild type state of all regulators (except possibly g_n) is 1. Our study could serve as a component in a more general scheme for dealing with entire networks, whose regulation relations possibly interact with one another.

Acknowledgments

R. M. Karp and R. Shamir were supported by a grant from the US-Israel Binational Science Foundation (BSF). R. Sharan was supported by a Fulbright grant. I. Gat-Viks was supported by a Colton fellowship.

References

1. T. Akutsu et al. *Theor. Comp. Sci.*, 298:235–251, 2003.
2. T. Ideker, V. Thorsson, and R.M. Karp. In *Proc. of the Pacific Symposium in Biocomputing*, pages 305–316, 2000.
3. N. Friedman et al. *J. Comp. Biol.*, 7:601–620, 2000.
4. A. Tanay and R. Shamir. *Bioinformatics*, 17, Supplement 1:270–278, 2001.
5. D. Hanisch et al. *Bioinformatics*, 18, Supplement 1:145–154, 2002.
6. T. Ideker et al. *Bioinformatics*, 18, Supplement 1:233–240, 2002.
7. E. Segal et al. *Bioinformatics*, 17, Supplement 1:243–252, 2001.
8. D. Pe'er, A. Regev, and A. Tanay. *Bioinformatics*, 18, Supplement 1:258–267, 2002.
9. I. Gat-Viks and R. Shamir. *Bioinformatics*, 19, Supplement 1:108–117, 2003.
10. F. C. Neidhardt, editor. ASM Press, 1996.
11. E. W. Jones, J. R. Pringle, and J. R. Broach, editors. Cold Spring Harbor Laboratory Press, 1992.
12. T. Ideker et al. *Science*, 292:929–933, 2001.