Available online at www.sciencedirect.com
SCIENCE DIRECT®
Metabolic Engineering 6 (2004) 326–339
METABOLIC ENGINEERING
http://www.elsevier.com/locate/ymben

A Boolean algorithm for reconstructing the structure of regulatory networks

Sarika Mehra,[a] Wei-Shou Hu,[a,*] and George Karypis[b]

[a] *Department of Chemical Engineering and Materials Science, University of Minnesota, 421 Washington Avenue SE, Minneapolis, MN 55455-0132, USA*

[b] *Department of Computer Science and Engineering, University of Minnesota, 200 Union Street SE, Minneapolis, MN 55455-0132, USA*

Received 1 October 2003; accepted 21 May 2004

Available online 15 July 2004

Abstract

Advances in transcriptional analysis offer great opportunities to delineate the structure and hierarchy of regulatory networks in biochemical systems. We present an approach based on Boolean analysis to reconstruct a set of parsimonious networks from gene disruption and over expression data. Our algorithms, Causal Predictor (CP) and Relaxed Causal Predictor (RCP) distinguish the direct and indirect causality relations from the non-causal interactions, thus significantly reducing the number of miss-predicted edges. The algorithms also yield substantially fewer plausible networks. This greatly reduces the number of experiments required to deduce a unique network from the plausible network structures. Computational simulations are presented to substantiate these results. The algorithms are also applied to reconstruct the entire network of galactose utilization pathway in *Saccharomyces cerevisiae*. These algorithms will greatly facilitate the elucidation of regulatory networks using large scale gene expression profile data.

Keywords: Regulatory networks; Boolean; Reverse engineering algorithm

1. Introduction

Biological processes are a manifestation of biochemical reaction networks of molecular synthesis and transformation, as well as regulatory networks controlling the expression of both regulatory elements and the biochemical reaction network. The regulatory network consists of a large number of regulatory elements of interacting genes and proteins organized in hierarchical trees. Cellular events, including physiological, differentiation and developmental, involve the interplay of these regulatory networks. In many cases, a subset of the cellular network for a particular event may be relatively isolated or localized, and can be analyzed separate from the global network. Examples include, the yeast mating type pheromone regulation (Gustin et al., 1998) and *Bacillus* sporulation regulation (Sonenshein, 2000). The elucidation of the structure and organization of these networks, at both local and global levels, can provide us with much insight into control of cellular events, and holds the key to harness the vast biochemical potential of living systems. With the advent of the post-genomic era a variety of large-scale gene expression profiling tools have enabled us to survey the temporal expression pattern of the regulatory elements. Deciphering this information for reconstructing the hierarchy of the regulatory elements is becoming even more urgent.

Various approaches for modeling of regulatory elements have been proposed. The primary aim of all these methods is to identify the interacting elements and construct the regulatory map to different degrees of detail. This is known as reverse engineering the network. A number of approaches are based on observing the temporal expression patterns of the different regulatory elements (Liang et al., 1998; Akutsu et al., 1999; Chen et al., 1999; D'Haeseleer et al., 1999; Weaver et al., 1999; Wahde and Hertz, 2000; Holter et al., 2001; Vance et al., 2002; Wolkenhauer, 2002; Gardner et al., 2003). Other kind of models use expression profiles from different

*Corresponding author.
E-mail address: acre@cems.umn.edu (W.-S. Hu).

1096-7176/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.ymben.2004.05.002

perturbation experiments (Akutsu et al., 1998; Maki et al., 2001; Wagner, 2001; Aburatani et al., 2003; Tegner et al., 2003). Differential equation-based approaches model the interactions as non-linear terms (Wahde and Hertz, 2000) or as a linear additive model from time profile data (Chen et al., 1999; D'Haeseleer et al., 1999; Weaver et al., 1999; Wolkenhauer, 2002). Although such an approach provides the most knowledge in terms of mechanistic details, it also inevitably results in a large number of equations and a large number of kinetic parameters whose values are difficult to determine. Furthermore, the number of parameters, typically far exceeds the number of time points for which data is available, making the problem of determining the system parameters an ill-posed one. The Boolean method offers the advantage of inferring direct connections among the regulatory elements without resorting to parameter values. A Boolean model treats every element as having two binary states, inactive (0) or active (1) and the interaction between the elements are modeled as Boolean rules. This approach provides a first approximation for the complexity of the problem. Boolean networks model both temporal and perturbation data (Akutsu et al., 1998; Liang et al., 1998; Akutsu et al., 1999; Ideker et al., 2000). Various other approaches based on the Bayesian method have also been proposed (Friedman et al., 2000).

The contributions of this paper are two-fold. First, we develop new algorithms for genetic network reconstruction using gene disruption and over-expression data. These algorithms are more robust as they completely eliminate the prediction of certain kinds of false interactions that are predicted by earlier algorithms. Second, we evaluate the usefulness of Boolean models in reverse engineering biological regulatory networks.

The paper is organized as follows. Section 2 is an overview of the application of Boolean networks in representing biological systems. Section 3 provides a formal definition of the problem and describes the algorithms that we developed. Section 4 provides an experimental evaluation of these algorithms on data generated by In silico experiments, as well as a network based on the yeast galactose pathway and compares them against previously developed algorithms. Section 5 discusses the relative merits of the proposed algorithms. Finally, Section 6 provides some concluding remarks and directions for future research.

2. Boolean network and inference strategies

Boolean networks represent genetic networks as many interconnected binary elements, with each of them connected to a series of others (Kauffman, 1969). A binary element may represent a gene, a protein or an environmental factor. The basic premise is that regulatory interactions, typically sigmoid, can be approximated as a step function and that the state of each element can be described as either ON (1) or OFF (0). For gene regulation a change in the state of an element or a gene refers to the formation of the gene expression product that is capable of exerting an effect on other related elements. In general this entails transcription and the formation of translation products. The Boolean model can be extended to other forms of networks. For example, in a signaling cascade, the phosphorylated/non-phosphorylated states of proteins are treated as binary elements.

Each gene may be influenced by one or more regulatory elements called inputs. The expression of a gene, called the output, is computed from the input pattern according to logical or Boolean rules. A repressor is equivalent to a NOT function, whereas cooperatively acting activators are represented with the AND function. The ON/OFF pattern of all the elements involved at a given time is the state of the entire network. When the values of the inputs change, the system updates itself as the genes interact until the system reaches a final state, called the *attractor*. This attractor could be a limit cycle or a steady state. The Boolean model has been used to describe various biological pathways including, signaling pathways (Shymko et al., 1997; Genoud et al., 2001) and bacterial degradation (Serra and Villani, 1997).

There are several methods to identify possible Boolean network structures that are consistent with the observed experimental profile for a given number of genes. These methods can be classified into two broad categories. The first is based on time-profile data where the expression pattern at two consecutive time steps, defined as an INPUT/OUTPUT pair, is used to construct the network (Akutsu et al., 1999; Liang et al., 1998). These methods generally require a large number of data. Also, it is experimentally difficult to measure consecutive states, failing which the correct network may not be identified.

The second set of methods entail introducing a series of perturbations to the network and observing the resulting steady-state profiles of the regulatory elements. The steady-state expression profiles are then used to identify a set of parsimonious networks. This can be achieved by minimizing the total number of interactions (Maki et al., 2001; Wagner, 2001; Aburatani et al., 2003). However, this approach does not allow for redundant interactions. Redundancy is inherent in biological networks. Parsimonious networks can also be constructed by allowing for redundancy in networks, but minimizing the number of essential inputs for each gene (Ideker et al., 2000). A drawback of this approach is that it predicts false predictions. Further experimentation is required to eliminate these false predictions. As the network becomes more complex, the frequency of

false predictions increases significantly, thereby requiring a large number of experiments to reconstruct the complete network. The work reported here presents a new algorithm to reduce this frequency of false prediction.

3. Boolean network prediction algorithm

3.1. Problem definition

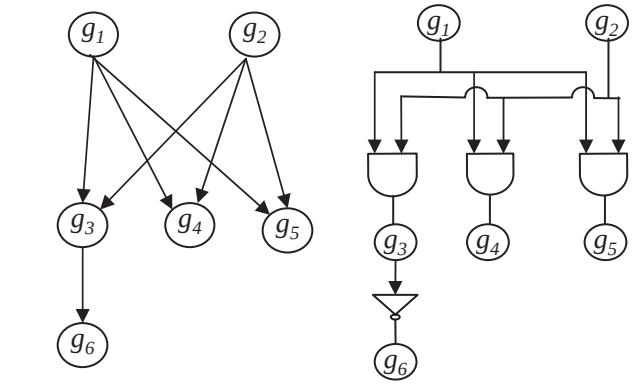
The objective of this algorithm is to reverse engineer a Boolean network from a set of observations obtained from different perturbations to the network. These perturbations could be the knockout of a gene, or its constitutive expression independent of the biological regulators. Specifically, given a set of N genes, we consider a $(N+1) \times N$ matrix, called binary expression matrix (B) that stores the Boolean state of the various elements (columns) under different conditions (rows). Specifically, the first row of B stores the initial unperturbed state (wild-type) of the elements, and each subsequent row $l+1$ stores the steady-state expression of the elements obtained after an experiment in which gene l has been perturbed. The network is reverse engineered from this expression matrix, by identifying the input elements for each gene, $g_n, n = 1, 2, \ldots, N$, one by one.

3.1.1. Assumptions

We assume that there is no polar effect due to any perturbation. Therefore the observed expression profile is the result of the introduced perturbation. This also implies that the network is self-contained, meaning that a change in expression level of any element is caused by one of the other elements in the network, either directly or indirectly. We also assume that the observed expression profiles represent the steady-state levels of the elements under each perturbation. In this context "steady state" applies to the steady state of the Boolean network. We are interested in the steady state of the system on the time-scale of observation.

3.1.2. Working example

The topology of our example regulatory network and the corresponding Boolean representation are shown in Fig. 1a and b. This regulatory structure, called the multi-input motif, is characterized by a set of regulators that bind together to a set of genes and is present in various transcriptional regulatory networks in *Saccharomyces cerevisiae* (Lee et al., 2002) and *Escherichia coli* (Shen-Orr et al., 2002). The corresponding binary expression matrix, B, is shown in Fig. 1c. The first row of the matrix represents the steady-state value of all the components of the pathway in the wild type. In each of the subsequent rows, the expression of one of the elements is perturbed by being turned ON (denoted as +) or turned OFF (denoted as −) from its initial state in row 1.



(a) (b)

(c)

Row Index		g_1	g_2	g_3	g_4	g_5	g_6
1	Wild type	1	1	1	1	1	0
2	g_1 OFF	-	1	0	0	0	1
3	g_2 OFF	1	-	0	0	0	1
4	g_3 OFF	1	1	-	1	1	1
5	g_4 OFF	1	1	1	-	1	0
6	g_5 OFF	1	1	1	1	-	0
7	g_6 ON	1	1	1	1	1	+

Fig. 1. Illustration of the working example. (a) Topology of the example regulatory network. (b) Boolean representation of the network. (c) Binary expression matrix, B, corresponding to the example network.

3.2. Overall methodology for reverse engineering Boolean networks

There are two characteristics of a Boolean network: topology and the rules of interaction. The topology of a network can be constructed by identifying the inputs of each element or alternatively, the outputs of each element. The rules of interaction are determined by the logic rule or the Boolean function governing each element of the network.

Reconstructing the topology is the most critical part of reverse engineering networks. Our reverse engineering algorithm reconstructs the topology by identifying the inputs of each element in a three-step process. The first step is to identify all those elements that are correlated. However, correlation does not imply causality and therefore the correlated elements need to be filtered for the most probable set of inputs. This filtering constitutes the second step. Finally, the network is constructed from the set of probable inputs by identifying the minimum number of inputs required to explain the set of observations. A detailed description of the overall methodology follows.

3.2.1. Identify correlated genes

To identify the potential inputs for g_n, we compare the expression level of g_n in every pair of rows i and j, where $j>i$, excluding row $n+1$ in which g_n itself has been perturbed. For each pair of rows i and j, where the expression of g_n differs, we identify the other genes $g_m (m \neq n)$ whose expression level also changes between the two rows. This set of genes, denoted by $S_{i,j}(g_n)$, consists of elements that are correlated with g_n and are probable inputs. For each gene there are thus multiple sets of probable inputs. Because the network is self-contained, each set $S_{i,j}(g_n)$ must contain at least one element whose change has altered the expression of g_n. In other words, at least one element in each set $S_{i,j}(g_n)$ must be a regulatory factor for g_n. For example, consider the gene expression matrix in Fig. 1c. To determine the set of inputs for g_3 we compare all rows pair wise between which g_3 has changed state, except row 4. One such pair of rows is *1* and *2*, and the corresponding set of correlated elements is $S_{1,2}(g_3) = \{g_1, g_4, g_5, g_6\}$. $S_{1,3}(g_3)$, $S_{2,5}(g_3)$ and others are similarly constructed. The complete list of all $S_{i,j}(g_n)$ is shown in Fig. 2a. Such an approach of identifying correlated elements has been proposed in other publications as well (Ideker et al., 2000).

3.2.2. Identify potential inputs from correlated genes using first-principle based filtering

Not all correlated genes are inputs. Two elements, g_i and g_n, can be correlated because of a *direct causality* relation where changes in g_i causes changes in g_n directly. By comparison, if g_i activates g_n via another regulatory factor, the correlation is an *indirect causality* relation. In addition, two genes g_i and g_n can also be correlated because of *identical regulatory inputs* or g_n can cause g_i, directly or indirectly by a relation known as *causality reversed*. Finally, the correlation between two elements may be *coincidental* where no causal connection exists. For the network shown in Fig. 1a, g_1 has a direct causality relation with g_3 and an indirect causality relation with g_6; g_4 and g_5 are correlated through an identical input, whereas there is no correlation between g_1 and g_2. To reverse engineer the network, we need to identify only those elements that are correlated to g_n in a direct or indirect causality relation. Failure to eliminate the rest of the correlations will give rise to falsely predicted network structure.

Two different algorithms are presented to eliminate the non-causal elements from the set of probable inputs. The first algorithm called the Causal Predictor (CP) ensures that all interactions that are predicted are only direct or indirect causality relations. However, in certain cases, this requires neglecting certain probable inputs. In contrast, the Relaxed Causal Predictor (RCP) algorithm relaxes the conditions used in CP such that some additional potential inputs are included. Both these approaches are explained below.

Consider an element $g_k \in S_{i,j}(g_n)$. The correlation between g_k and g_n could be causal, or it could be due to identical regulatory inputs, causality reversed or coincidental. To distinguish between the two possibilities, we consider the set $S_{1,k+1}(g_k)$ that is obtained by comparing the first row, corresponding to the wild type state, and the row $k+1$, where element k was perturbed. If g_n has changed state upon perturbation of g_k, it is definitely directly or indirectly affected by g_k. Thus, $g_n \in S_{1,k+1}(g_k)$ is a sufficient condition for g_k to be a regulatory input for g_n. However, this may not be a necessary condition. If g_n does not change state upon the perturbation of an element $g_l \in S_{i,j}(g_n)$, it does not imply that g_l is not an input for g_n. The effect of g_l may be complemented by another element. However, g_k is more probable to be an input for g_n than g_l. By removing g_l from all the sets of probable inputs, we can increase the probability of predicting the right network architecture. We therefore impose a "feasibility" constraint on all sets, $S_{i,j}(g_n)$ such that any element $g_k \in S_{i,j}(g_n)$ is removed from $S_{i,j}(g_n)$ if $g_n \notin S_{1,k+1}(g_k)$. The resulting sets are called $S^1_{i,j}(g_n)$. As shown in Fig. 2a, g_5 appears as a probable input of g_4. However, when g_5 is turned OFF, no change in g_4 is observed compared to its initial state, or $g_4 \notin S_{1,6}(g_5)$. Hence, we do not consider g_5 as a probable input in the sets $S^1_{i,j}(g_4)$. The modified sets of probable inputs are shown in Fig. 2b. This approach where the probable regulatory inputs are represented by the sets $S^1_{i,j}(g_n)$ is referred to as the CP algorithm. Here, all the elements satisfy the feasibility constraint. Therefore, the predicted network contains only direct and indirect causality relations. Further experiments are required to differentiate between these. However, the essential structure of the underlying network is captured.

If no element $g_k \in S_{i,j}(g_n)$ satisfies the feasibility constraint, the corresponding set $S^1_{i,j}(g_n)$ will be empty. Algorithm CP eliminates this set from the sets of probable inputs for g_n. However, at least one element of $S_{i,j}(g_n)$ is an input for g_n because of the self-contained assumption. An alternative approach would be to relax the feasibility constraint such that any element $g_k \in S_{i,j}(g_n)$ is eliminated from $S_{i,j}(g_n)$ only if $g_n \notin S_{1,k+1}(g_k)$ and the resulting set is non-empty. We implement this approach and the corresponding algorithm is referred to as the RCP algorithm. The sets derived from $S_{i,j}(g_n)$ using the modified feasibility constraint are referred to as $S^2_{i,j}(g_n)$. Although, algorithm RCP cannot ensure that all predicted interactions are direct or indirect causality relations, it guarantees that some potentially important probable inputs are not missed out.

(a)

Regulated element	Corresponding set of correlated inputs
g_1	$S(g_1) = \{\text{empty}\}$
g_2	$S(g_2) = \{\text{empty}\}$
g_3	$S_{1,2}(g_3) = \{g_1, g_4, g_5, g_6\}$, $S_{1,3}(g_3) = \{g_2, g_4, g_5, g_6\}$, $S_{2,5}(g_3) = \{g_1, g_5, g_6\}$, $S_{2,6}(g_3) = \{g_1, g_4, g_6\}$, $S_{2,7}(g_3) = \{g_1, g_4, g_5\}$, $S_{3,5}(g_3) = \{g_2, g_5, g_6\}$, $S_{3,6}(g_3) = \{g_2, g_4, g_6\}$, $S_{3,7}(g_3) = \{g_2, g_4, g_5\}$
g_4	$S_{1,2}(g_4) = \{g_1, g_3, g_5, g_6\}$, $S_{1,3}(g_4) = \{g_2, g_3, g_5, g_6\}$, $S_{2,4}(g_4) = \{g_1, g_5\}$, $S_{2,6}(g_4) = \{g_1, g_3, g_6\}$, $S_{2,7}(g_4) = \{g_1, g_3, g_5\}$, $S_{3,4}(g_4) = \{g_2, g_5\}$, $S_{3,6}(g_4) = \{g_2, g_3, g_6\}$, $S_{3,7}(g_4) = \{g_2, g_3, g_5\}$
g_5	$S_{1,2}(g_5) = \{g_1, g_3, g_4, g_6\}$, $S_{1,3}(g_5) = \{g_2, g_3, g_4, g_6\}$, $S_{2,4}(g_5) = \{g_1, g_4\}$, $S_{2,5}(g_5) = \{g_1, g_3, g_6\}$, $S_{2,7}(g_5) = \{g_1, g_3, g_4\}$, $S_{3,4}(g_5) = \{g_2, g_4\}$, $S_{3,5}(g_5) = \{g_2, g_3, g_6\}$, $S_{3,7}(g_5) = \{g_2, g_3, g_4\}$
g_6	$S_{1,2}(g_6) = \{g_1, g_3, g_4, g_5\}$, $S_{1,3}(g_6) = \{g_2, g_3, g_4, g_5\}$, $S_{1,4}(g_6) = \{g_3\}$, $S_{2,5}(g_6) = \{g_1, g_3, g_5\}$, $S_{2,6}(g_6) = \{g_1, g_3, g_4\}$, $S_{3,5}(g_6) = \{g_2, g_3, g_5\}$, $S_{3,6}(g_6) = \{g_2, g_3, g_4\}$, $S_{4,5}(g_6) = \{g_3, g_4\}$, $S_{4,6}(g_6) = \{g_3, g_5\}$

(b)

Regulated element	Corresponding set of most probable inputs
g_1	$S^1(g_1) = \{\text{empty}\}$
g_2	$S^1(g_2) = \{\text{empty}\}$
g_3	$S^1_{1,2}(g_3) = \{g_1\}$, $S^1_{1,3}(g_3) = \{g_2\}$, $S^1_{2,5}(g_3) = \{g_1\}$, $S^1_{2,6}(g_3) = \{g_1\}$, $S^1_{2,7}(g_3) = \{g_1\}$, $S^1_{3,5}(g_3) = \{g_2\}$, $S^1_{3,6}(g_3) = \{g_2\}$, $S^1_{3,7}(g_3) = \{g_2\}$
g_4	$S^1_{1,2}(g_4) = \{g_1\}$, $S^1_{1,3}(g_4) = \{g_2\}$, $S^1_{2,4}(g_4) = \{g_1\}$, $S^1_{2,6}(g_4) = \{g_1\}$, $S^1_{2,7}(g_4) = \{g_1\}$, $S^1_{3,4}(g_4) = \{g_2\}$, $S^1_{3,6}(g_4) = \{g_2\}$, $S^1_{3,7}(g_4) = \{g_2\}$
g_5	$S^1_{1,2}(g_5) = \{g_1\}$, $S^1_{1,3}(g_5) = \{g_2\}$, $S^1_{2,4}(g_5) = \{g_1\}$, $S^1_{2,5}(g_5) = \{g_1\}$, $S^1_{2,7}(g_5) = \{g_1\}$, $S^1_{3,4}(g_5) = \{g_2\}$, $S^1_{3,5}(g_5) = \{g_2\}$, $S^1_{3,7}(g_5) = \{g_2\}$
g_6	$S^1_{1,2}(g_6) = \{g_1, g_3\}$, $S^1_{1,3}(g_6) = \{g_2, g_3\}$, $S^1_{1,4}(g_6) = \{g_3\}$, $S^1_{2,5}(g_6) = \{g_1, g_3\}$, $S^1_{2,6}(g_6) = \{g_1, g_3\}$, $S^1_{3,5}(g_6) = \{g_2, g_3\}$, $S^1_{3,6}(g_6) = \{g_2, g_3\}$, $S^1_{4,5}(g_6) = \{g_3\}$, $S^1_{4,6}(g_6) = \{g_3\}$

(c)

Regulated element	Minimum inputs from sets $S^1(g_n)$ and $S^2(g_n)$
g_1	$S_{min}(g_1) = \{\text{empty}\}$
g_2	$S_{min}(g_2) = \{\text{empty}\}$
g_3	$S_{min}(g_3) = \{g_1, g_2\}$
g_4	$S_{min}(g_4) = \{g_1, g_2\}$
g_5	$S_{min}(g_5) = \{g_1, g_2\}$
g_6	$S_{min}(g_6) = \{g_3\}$

(d)

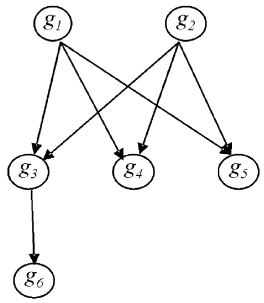


Fig. 2. Reconstructing the topology of the network. (a) First step of topology reconstruction, Sets $S_{i,j}(g_n)$ of correlated elements for each element g_n. (b) Most probable set of inputs $S^1_{i,j}(g_n)$ obtained by filtering the corresponding set of correlated elements. (c) Set of input elements under the parsimonious condition. (d) Topology of the predicted network using CP and RCP algorithms.

3.2.3. *Construction of networks from set of probable inputs: parsimonious condition*

All the elements in the sets $S^1_{i,j}(g_n)$ or $S^2_{i,j}(g_n)$ are probable inputs for g_n. If we construct a regulatory network by including all the elements in $\bigcup S^1_{i,j}(g_n)$(or $\bigcup S^2_{i,j}(g_n)$) as inputs for g_n, the resulting network will be very complex with a large number of edges. A better approach is to capture the essential structure of the network by identifying the minimum number of inputs that can explain the observations. This is achieved by determining the smallest set $S_{\min}(g_n)$, from the sets $S^1_{i,j}(g_n)$ such that at least one element of $S_{\min}(g_n)$ is present in each set $S^1_{i,j}(g_n)$. This smallest set represents the minimum inputs for gene g_n that are required to explain the expression matrix. A similar approach is followed to find the minimum sets corresponding to the sets $S^2_{i,j}(g_n)$. The task of identifying $S_{\min}(g_n)$ is analogous to a minimum set covering problem in graph-theory applications that can be solved using the greedy algorithm (Cormen, 2001). The greedy

approach progresses iteratively, by selecting at each step the element(s) that is present in the maximum number of sets $S^1_{i,j}(g_n)$. The element is subsequently removed from all the sets and the next highest occurring element is identified. The algorithm terminates when all the sets are empty and hence have been explained. For example, $S_{\min}(g_3) = \{g_1, g_2\}$ is the smallest set identified using the Greedy approach that can explain all the sets $S^1_{i,j}(g_3)$. Fig. 2c shows the calculated $S_{\min}$ sets for all the genes. More than one alternative $S_{\min}$ sets may be found for each gene g_n. The predicted network is assembled by putting together the minimum sets for each gene. In the event of multiple sets, each of the alternative sets can form a different network. The total number of different networks is given by the product of the number of minimum sets for each gene. We call these predicted networks as inferred networks. For the example shown in Fig. 1a, the sets $S^2_{i,j}(g_n)$ are identical to the sets $S^1_{i,j}(g_n)$, and therefore the CP and RCP algorithms yield the same result (Fig. 2d). In this case only one network is predicted, and remarkably the entire network can be reverse engineered.

The importance of filtering the correlated elements for the most probable causal relations can be best demonstrated here. If the unfiltered sets, $S_{i,j}(g_n)$ are used to construct the parsimonious networks, three alternative minimum sets are predicted for g_3, and two each for g_4 and g_5 (Fig. 3a). This results in a total of $3 \times 2 \times 2 = 12$ different networks, none of which is the correct one (Fig. 3b). Wrong connections are predicted between g_3, g_4 and g_5, which are correlated due to common inputs. By eliminating such non-causality relations, the proposed algorithms are able to predict the true inputs for these genes.

(a)

Regulated element	Minimum inputs from sets $S_{i,j}(g_n)$
g_1	$S_{min}(g_1)$ = {empty}
g_2	$S_{min}(g_2)$ = {empty}
g_3	$S_{min}(g_3)$ = {g_4, g_5}, {g_4, g_6}, {g_5, g_6}
g_4	$S_{min}(g_4)$ = {g_3, g_5}, {g_5, g_6}
g_5	$S_{min}(g_5)$ = {g_3, g_4}, {g_4, g_6}
g_6	$S_{min}(g_6)$ = {g_3}

(b)

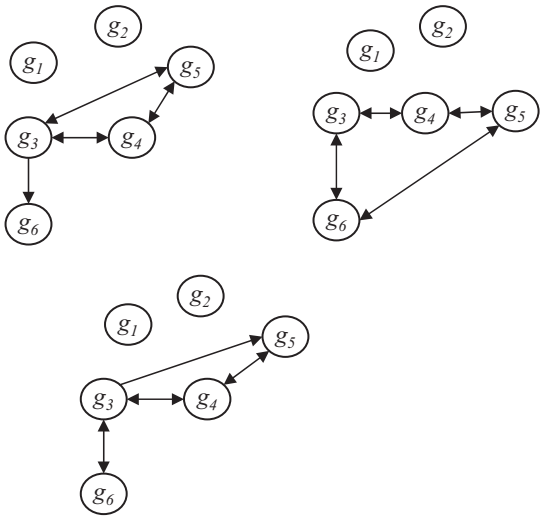


Fig. 3. Reconstructing the topology of networks without filtering for non-causal elements. (a) Set of input elements under the parsimonious condition from the sets $S_{i,j}(g_n)$. (b) Topology of predicted networks. Only 3 out the 12 predicted networks are shown.

3.2.4. Construction of boolean function

To determine the Boolean function governing each interaction in the inferred network, a list of outputs for all possible input combinations, or the truth table, is constructed. For an element with n inputs, the corresponding truth table has 2^n distinct input combinations. The output corresponding to each input combination is obtained from the binary expression matrix. This is explained with an example below. However, not all instances of input state combinations maybe present in the expression matrix resulting in alternative Boolean functions. Some of these Boolean functions may not be biologically relevant (Raeymaekers, 2002) and hence can be eliminated. Additional experiments can be designed to distinguish between the rest.

Fig. 4 shows the truth table for determining the logical relation between the predicted inputs for each gene in the network shown in Fig. 2d. The output state of g_3 when both of the predicted inputs g_1 and g_2 are ON, is determined from the state of g_3 corresponding to the row in the Boolean expression matrix (Fig. 1c) where g_1 and g_2 are 1. Similarly, the output state of g_3 is

g_1	g_2	g_i
1	1	1
1	0	0
0	1	0
0	0	0/1

g_3	g_6
1	0
0	1

$g_i = g_1$ **AND** g_2

or

$g_i = g_1$ **AND** g_2 **OR** (**NOT** g_1 **AND NOT** g_2)

where $i = 3, 4, 5$.

Fig. 4. Truth table for each element of the predicted network. The output state is determined from the binary expression matrix. Incomplete truth table is indicated by a 0/1 value in the output state.

determined when either g_1 or g_2 is 1. However, there is no row in the expression matrix where g_1 and g_2 are 0. The corresponding state of g_3 may thus be either 0 or 1, and therefore g_3 maybe regulated by g_1 and g_2 in an AND relation ($g_3 = g_1$ AND g_2) or the logic function describing their relation may be more complex (Fig. 4).

3.3. Practical considerations

Experimentally, gene expression profiles are likely to be obtained by DNA microarray or quantitative PCR. Consider an expression matrix, where the gene expression levels of all genes are determined with respect to a common reference condition. To construct a Binary expression matrix, the expression ratios of each gene under different perturbation experiments are normalized with respect to its own mutant experiment. Next, the gene is assigned a state of 0 when it is knocked out, and a state of 1 when it is over-expressed. The state of the gene under all the other conditions is determined depending on a user-defined threshold value (e.g., a two-fold change with low p-values). Alternatively, the cut-off value can be based on the mean and standard deviation of the distribution of the expression ratios under different perturbations. If the expression ratio is greater than the significant cut-off a state greater than its state in the mutant experiment is assigned and vice versa.

In certain cases, the structure of the final expression matrix can validate the discretization procedure. For example, a gene should have the lowest defined state under the perturbation experiment when the gene itself has been deleted. Similarly, if a gene has been assigned the same state in the wild type experiment and the perturbation experiment when it was itself perturbed, then all the other elements in the network should also have the same state in the two experiments. Else, the 'self-contained' assumption is violated.

However, such discretization of the data into only two levels may not always be feasible or appropriate. The algorithm described above can be extended to analyze gene expression data with multiple states. In this case the change of state could be from any one level to another, irrespective of the magnitude of the change. In the case the data is discretized into three levels: 0, 1 and 2, change of states can be between (0, 1), (1, 2) or (0, 2). However the change between 1 and 0 is not distinguished from a change between 2 and 0.

4. Experimental evaluation

To evaluate the performance of the algorithms we need a set of networks whose structure is completely known. However, there are not many such biological networks. Hence, we validate our approach on a set of synthetically generated networks. In addition, we also evaluate the algorithms on a network structure based on the *Saccharomyces cerevisiae* galactose-utilization regulatory network. In the application of the algorithm, the original network would be unknown and is to be reverse engineered from the expression matrix obtained experimentally.

4.1. Evaluation metrics

The effectiveness of the algorithms in reverse engineering the original network, referred as target network, is evaluated in terms of their sensitivity, specificity and *F*-factor. Sensitivity is a measure of how much of the target network can be predicted, whereas specificity represents the accuracy of the algorithm. *F*-factor balances both the sensitivity and specificity and is defined as the harmonic mean of the two quantities.

$$\text{Sensitivity} = \frac{\text{Number of common edges between inferred and target network}}{\text{Total number of edges in target network}},$$

$$\text{Specificity} = \frac{\text{Number of common edges between inferred and target network}}{\text{Total number of edges in inferred network}},$$

$$F\text{-factor} = \frac{2 * \text{Sensitivity} * \text{Specificity}}{\text{Sensitivity} + \text{Specificity}}.$$

For each set of network parameters, 200 different networks were generated. After applying the CP and RCP algorithms to predict the inferred network(s), the average sensitivity and specificity values over these 200 networks were calculated. Also, the average number of inferred networks for each target network was calculated. Furthermore, the false edges were identified. Depending on their relationship to the gene for which input genes are being determined they are classified into the following four categories: indirect causality, identical regulatory inputs, causality reversed and others.

4.2. Synthetic data generator

The synthetic network generator we devised is outlined in Fig. 5. A biological network can be represented as a directed graph, where each node represents a gene and the edges between them are the interactions. The number of inputs for any node is called its indegree. If feedforward or feedback loops are present in the network, it is termed cyclic otherwise it is called an acyclic network. We generated a set of acyclic and cyclic synthetic networks with the number of genes varying from 10 to 100, a maximum indegree of two, four or eight and no self-loops. For each node, a conjunctive Boolean expression is generated randomly. A Boolean expression is in conjunctive form when it is a conjunction of clauses, the variables within each clause are connected by "OR" and the clauses are related by "AND". For example, the Boolean function A XOR B, can be written as (((NOT A) OR (NOT B)) AND (A OR B)) in conjunctive form**.** We evaluate the truth table for each randomly generated conjunctive expression, and check that it is not independent of any of its immediate inputs.

Each network was simulated to steady state from a number of randomly selected initial conditions by updating all the nodes simultaneously. For each of the networks, perturbation experiments were also carried out. Specifically, if a node was active (or inactive) in the wild type, the particular network was simulated by forcing that node to a constant value of 0 (or 1). Any network that exhibits limit cycle behavior in any of the perturbation experiments was rejected. The steady states from each perturbation experiment, represented in a binary expression matrix were used to infer the target networks by our CP and RCP algorithms. The inferred networks are compared to the original target network to compute the sensitivity, specificity and *F*-factor values.

The number of genes together with the maximum indegree determines the complexity of the network. Complexity is defined as the lower bound of the number of experiments required to completely determine its topology. For a network of n elements, and maximum indegree of k, the total number of experiments is of the order of n^k (Akutsu et al., 1998). However, this is only an approximate measure of complexity. The distribution of the indegree of each node in the network is another important factor determining the degree of complexity.

4.3. Simulation results

The average sensitivity and specificity values for different network configurations obtained using the CP and RCP algorithms are listed in Table 1. The results are listed in increasing order of complexity of the target network.

We see that the sensitivity of the inferred networks decreases with increasing complexity. This is expected since we have considered here only n experiments, which become a smaller fraction of the total number of experiments required as the complexity of the networks increases. The specificity values, generally higher than sensitivity values, do not show a monotonic variation with complexity. For very complex networks, a small fraction of the network is predicted and hence the percentage of false predictions also decreases. The *F*-factor however, decreases with the complexity of network.

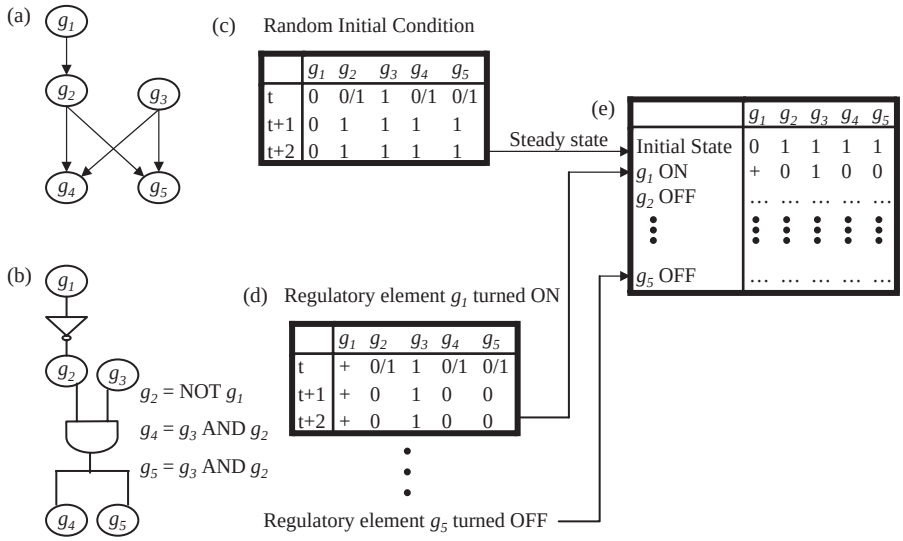


Fig. 5. Schematic of Synthetic Data Generator. (a) Random network topology for a given number of nodes ($N = 5$) and maximum indegree (2). (b) Assignment of a random Boolean function for each element. (c) Simulation of the network to a steady state from some random initial condition to generate the first row of the gene expression matrix. (d) Simulation of the network perturbed by knocking out or over-expressing each regulatory element to generate the subsequent rows of the expression matrix. (e) Corresponding gene expression matrix.

Table 1
Sensitivity and Specificity values of the predicted networks for CP and RCP algorithms for different target network architectures

Target network characteristics			Sensitivity (%)			Specificity (%)		
Number of genes	Indegree	Number of edges	Causal Predictor (CP)	Relaxed Causal Predictor (RCP)	Ideker's approach	Causal predictor (CP)	Relaxed causal predictor (RCP)	Ideker's approach
10	2	12	59±0.9	59±0.9	57±0.9	96±0.8	94±0.8	89±1.0
20	2	27	53±0.6	53±0.6	50±0.6	95±0.6	93±0.6	84±0.8
50	2	72	54±0.4	54±0.4	48±0.3	94±0.4	92±0.5	79±0.6
100	2	150	54±0.3	53±0.3	47±0.2	94±0.3	93±0.3	77±0.5
20	4	62	22±0.3	22±0.3	20±0.3	95±0.6	93±0.6	84±0.8
20	6	90	14±0.3	14±0.3	14±0.3	95±0.6	95±0.5	86±0.8
20	8	120	10±0.2	10±0.2	10±0.2	96±0.5	95±0.5	87±0.8

Note: The data shown are mean ±standard error for 200 target networks.
The values obtained using Ideker's approach are also included for comparison.

Table 2
Distribution (%) of false edges in the predicted networks into indirect causality and no causality categories

Target network characteristics		Causal predictor (CP)		Relaxed causal predictor (RCP)		Ideker's approach	
Number of genes	Indegree	Indirect causality	No causality	Indirect causality	No causality	Indirect causality	No causality
10	2	100	0	100	0	99	1
20	2	100	0	96	4	81	19
50	2	100	0	34	66	29	71
100	2	100	0	64	36	35	65
20	4	100	0	78	22	75	25
20	8	100	0	100	0	100	0

Note: The data shown are averages over 200 different target networks.
Results for different target network configurations are shown for the CP and RCP algorithms along with those obtained using Ideker's approach.

For a given network configuration, the CP and RCP algorithms are comparable in terms of average prediction statistics. However, there are differences in individual cases, which is evident when we look at the distribution of the false edges. As discussed above, the incorrect correlations can be either due to indirect causality or are non-causal in nature. The non-causal interactions are further classified into three types: identical regulatory inputs, causality reversed and others. Table 2 shows the distribution of the false edges into the two major categories: indirect causality and non-causality correlations. In case of overlap between these two categories, we have given the indirect edges precedence over the non-causal interaction. We can see that the CP algorithm can completely eliminate the non-causality kind of edges. Thus, although all the predicted regulatory connections may not be actual physical connections (reflected in less than 100% specificity values), they represent effective functional relations. In contrast, the RCP algorithm eliminates a fraction of the non-causality relations. This was as expected and has been discussed during the development of the algorithm. Note that for the network with 20 genes and an indegree of 8, there are negligible non-causal relations. Due to the highly interconnected nature of networks with a high indegree, the non-causality relations overlap with indirect interactions and hence have been classified in the latter category.

Table 3 presents the average number of networks inferred by the current algorithms for different network architectures. The number of predicted networks increases with the number of genes in the system. However, for a given configuration, the number of networks predicted for a target network varies over a wide range, as shown in Table 3. The distribution of the number of predicted networks for networks with 50 genes and indegree of 2 is shown in the histogram chart of Fig. 6. In more than 75% of the target networks the CP algorithm predicts a unique network whereas the RCP predicts a single network for 20% of the test cases.

4.4. Application of algorithms to biological networks

We apply our algorithm to a gene expression profile dataset obtained by systematic perturbation to the yeast galactose utilization pathway, one of the best-studied systems in *Saccharamyces cerevisiae*. It consists of a set of structural and regulatory genes, that enable cells to utilize galactose as a carbon source by converting galactose to glucose-6-phosphate. The structural genes GAL2, GAL1, GAL7, GAL10 and GAL5 encode the

Table 3
Average number of networks predicted by the CP and RCP algorithms for different target network architectures

Target network characteristics		Causal predictor (CP)	Relaxed causal predictor (RCP)	Ideker's approach
Number of genes	Indegree			
10	2	268 (1, 3×10^4)	272 (1, 3×10^4)	273 (1, 3×10^4)
20	2	3e3 (1, 6×10^5)	4e3 (1, 6×10^5)	4e4 (1, 8×10^6)
50	2	2e4 (1, 3×10^5)	6e8 (1, 6×10^{10})	5e13 (1, 7×10^{15})
100	2	2e7 (1, 2×10^9)	5e13 (1, 6×10^{15})	4e23 (9, 6×10^{25})
20	4	134 (1, 3×10^4)	3e2 (1, 4×10^4)	2e3 (1, 3×10^5)
20	6	8 (1, 6×10^2)	2e2 (1, 2×10^4)	358 (1, 5×10^4)
20	8	3 (1, 2×10^2)	26 (1, 3×10^3)	384 (1, 6×10^4)

Note: The data shown are averages over 200 different target networks.
The numbers in the brackets represent the minimum and maximum number of networks predicted by each algorithm. The results obtained using Ideker's approach are also shown.

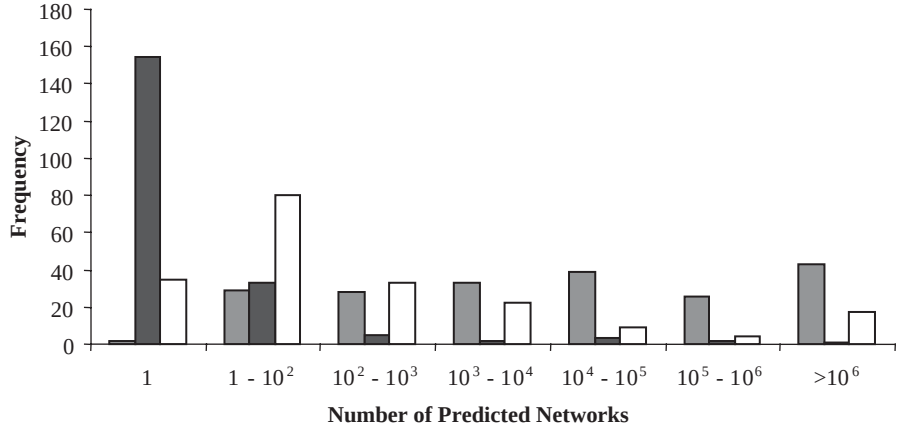
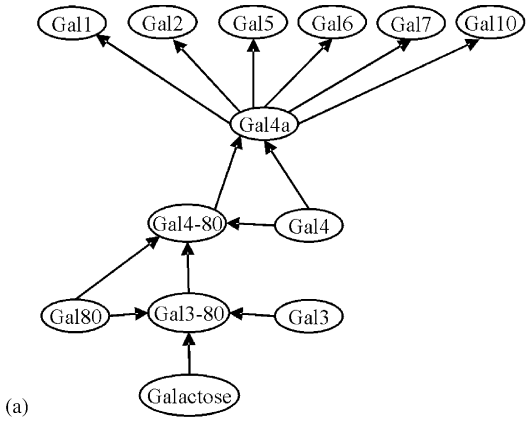


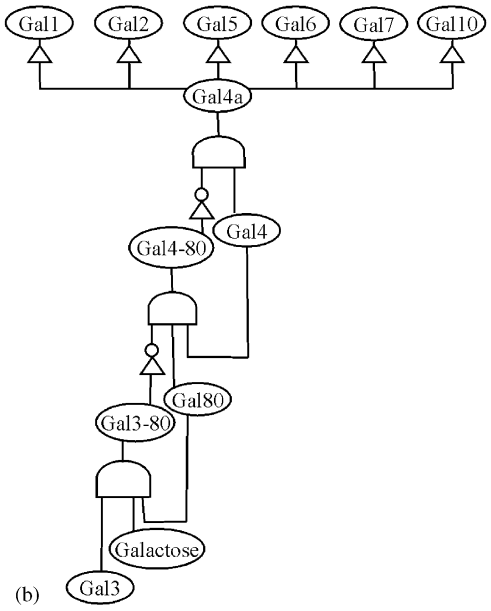
Fig. 6. Distribution of the number of predicted networks for the CP (■), RCP (□), and Ideker (▩) algorithms. A sample size of 200 target networks is used where all networks have 50 genes with a maximum indegree of 2.

galactose permease, galactokinase, UDP-glucose–hexose-1-phosphate uridylyltransferase, UDP-glucose 4-epimerase and phosphoglucomutase proteins responsible for transporting galactose into the cell and converting intracellular galactose to glucose-6-phosphate via galactose-1-phosphate and glucose-1-phosphate. We consider the regulatory network responsible for inducing the GAL structural genes to a high level in the presence of galactose as opposed to in the presence of glycerol. This transcriptional control is primarily exerted by the regulatory genes GAL4, GAL3 and GAL80. Gal4 protein is the main transcriptional activator, which induces the expression level of the GAL1, GAL2, GAL7 and GAL10 structural genes by more than 1000-fold and the GAL5 gene by 100-fold in the presence of galactose. The Gal80 protein inhibits the activity of Gal4 protein in the absence of galactose by binding to the transcription activation domain of Gal4 protein (Lohr et al., 1995). This repression is relieved by the Gal3 protein. A number of different hypothesis have been proposed to explain this mechanism. A recent report showed that Gal3 protein forms a complex with Gal80 protein in the cytoplasm, and thus prevents Gal80 protein from inhibiting Gal4 protein in the nucleus (Peng and Hopper, 2002). The role of the GAL6 gene is not entirely clear. However, the expression level of GAL1, GAL2 and GAL7 structural genes is increased in a Gal6 mutant (Zheng et al., 1997). The initial entry of galactose into cells growing on glycerol may occur by a constitutive, GAL2-independent process or by Gal2 protein mediated entry, enabled by low-level GAL2 expression in glycerol.

We first describe this network using the Boolean framework. The network topology is shown in Fig. 7a. There are in total 13 species, nine representing the mRNA state of the five structural genes, GAL1, GAL2, GAL7, GAL10, GAL5, and the four regulatory genes GAL3, GAL4, GAL80 and GAL6. For all these species, the presence of mRNA ensures the corresponding protein; therefore the same species is used to denote the mRNA and protein state. However, the Gal4 protein can exist in two different states, either as unbound protein that can activate the transcription of other genes or in a complex with the Gal80 repressor denoted by Gal4–80. We therefore call the unbound state as Gal4a (Gal4 active). Another protein–protein complex is



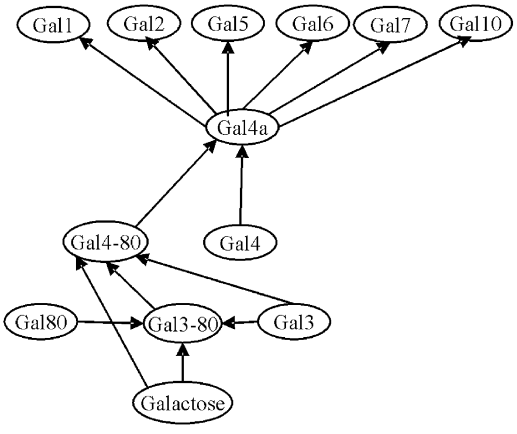
(a)



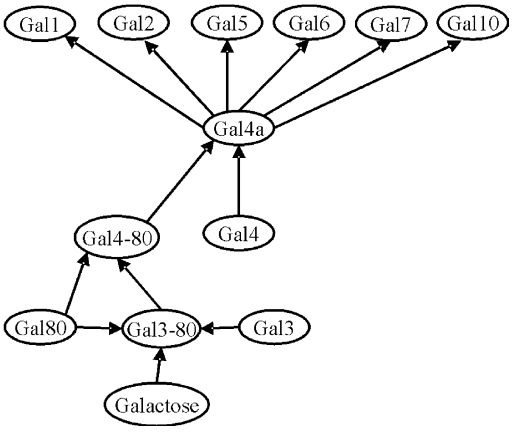
(b)

(c)

	Gal1	Gal2	Gal3	Gal4	Gal4a	Gal5	Gal6	Gal7	Gal10	Gal80	Gal4-80	Gal3-80	Galactose
Wild type	1	1	1	1	1	1	1	1	1	1	0	1	1
Gal1-	-	1	1	1	1	1	1	1	1	1	0	1	1
Gal2 -	1	-	1	1	1	1	1	1	1	1	0	1	1
Gal3-	0	0	-	1	0	0	0	0	0	1	1	0	1
Gal4-	0	0	1	-	0	0	0	0	0	1	0	1	1
Gal4a-	0	0	1	1	-	0	0	0	0	1	0	1	1
Gal5-	1	1	1	1	1	-	1	1	1	1	0	1	1
Gal6-	1	1	1	1	1	1	-	1	1	1	0	1	1
Gal7 -	1	1	1	1	1	1	1	-	1	1	0	1	1
Gal10 -	1	1	1	1	1	1	1	1	-	1	0	1	1
Gal80 -	1	1	1	1	1	1	1	1	1	-	0	0	1
Gal4-80 +	0	0	1	1	0	0	0	0	0	1	+	1	1
Gal3-80 -	0	0	1	1	0	0	0	0	0	1	1	-	1
Galactose -	0	0	1	1	0	0	0	0	0	1	1	0	-



(d)



(e)

Fig. 7. Application of the algorithms to analyze gene expression of *Saccharomyces cerevisae* galactose pathway. (a) Network Topology. (b) Boolean representation of the pathway. (c) Expression matrix generated by simulating the Boolean network. (d) The single network predicted by algorithm CP. (e) One of the 8 different networks predicted by RCP. The network reproduces the original expression matrix on simulation.

formed by the Gal3 and Gal80 proteins in the presence of galactose denoted as Gal3–80. Finally, galactose is also included as a variable in the model.

The Boolean rule governing each species is shown in Fig. 7b. The mRNA state for the structural genes is directly regulated by Gal4a. The presence of the active form of Gal4 protein is contingent upon the presence of the corresponding mRNA and the absence of the Gal4–80 complex. The corresponding Boolean rule for Gal4a incorporates both these conditions. The Gal3–80

complex is formed preferentially over the Gal4–80 complex. In Boolean language, this implies that Gal4–80 complex is formed only when the Gal3–80 complex is not present, given that the Gal4 and Gal80 species are available. The Gal3 and Gal80 proteins are present in the absence of galactose and their expression is modestly induced in the presence of galactose via Gal4a. Thus, to correctly model the expression levels of these two genes, we would need a three state (0, 1 and 2) model, where state 1 represents the constitutive expression in the absence of Gal4a and 2 represents the state attained on activation by Gal4a. In our Boolean model, we assume that these two proteins are constitutively active or in state 1. Similarly, to include the effect of the Gal6 protein we need 3 states for the structural genes, where in the absence of Gal6 the structural genes attain a state of 2 and in the presence of Gal6 and Gal4a a state 1. However, both states 1 and 2 are active. In the Boolean representation of the network, the effect of Gal6 cannot be included. Although, some assumptions need to be made; the Boolean model does capture the essential nature of the pathway.

To confirm that the network shown above can indeed give rise to the induction effect of galactose, it was simulated using the initial condition where the Gal structural genes are OFF and galactose is added to the yeast cells. A steady state was finally achieved with all structural genes being turned ON. Based on this network, we also simulate perturbation experiments where each species is perturbed from its wild type state. This could be achieved experimentally by knocking out the corresponding gene, inactivating the protein or expressing the protein independent of its native control. The steady state corresponding to each of these perturbation experiments is represented in the expression matrix, shown in Fig. 7c. Our algorithms are next evaluated for their ability to reconstruct the target network from the expression matrix. The CP algorithm outputs a single network (Fig. 7d) with a sensitivity and specificity of 86%, whereas the RCP predicts 8 different networks with an average sensitivity of 87% and average specificity of 93%. One of these networks is shown in Fig. 7e. The 7 other networks predicted by the RCP algorithm differ from the one shown in a single edge. Instead of predicting an interaction from Gal80 to Gal4-80, an edge is predicted from one of Gal1, Gal2, Gal5, Gal6, Gal7, Gal10 or Gal4a to Gal4–80.

A truth table is constructed for each node of the predicted networks to determine the logical function governing the corresponding inputs. If the truth table is incomplete, and multiple logical rules are possible, we choose the rule that is biologically the most relevant (Raeymaekers, 2002). The corresponding network is then simulated for each perturbation experiment and compared to the original expression matrix. This can be used as a consistency check to further prune down the different predicted networks. Thus, the simulation of only one (Fig. 7e) out of the 8 different networks predicted using RCP, could reconstruct the original expression matrix. We also simulate the experiments where each element is perturbed in the absence of galactose. These simulations correspond to perturbations in two elements of the network. An expression matrix is then constructed with steady-state profiles from both single and double perturbation experiments. The entire network can now be reconstructed by the CP and RCP algorithms. In addition, only a single network is predicted.

4.5. Comparison with other approaches

We have compared our approach to other similar approaches based on analyzing gene disruption and over-expression data under the Boolean framework. One such approach was proposed by Ideker et al. (2000). We applied an algorithm based on their approach to our synthetic data set as well as the gene expression matrix generated using the yeast galactose pathway structure. Our algorithms show a consistent increase in the specificity values over the Ideker algorithm for the synthetic data set, as shown in Table 1. Our algorithms also outperform in predicting the yeast galactose pathway. An approach based on Ideker's algorithm will infer 120 different networks with an average sensitivity and specificity of 72.3% as opposed to 8 predicted networks with an average sensitivity of 87% and specificity of 93% using RCP. Thus, a more accurate network can be predicted using our approach. In addition the current algorithms reduce the number of inferred networks drastically as compared to Ideker's approach, while still predicting a better network. As shown in Fig. 6, Ideker's algorithm predicts a single network for only 1% of the synthetic test cases.

The trend of sensitivity and specificity values for our implementation of Ideker's algorithm (Table 1) compare well with those presented in their paper (Ideker et al., 2000). However, there are differences in the absolute values. The differences could be due to various reasons. First, the networks and the governing Boolean functions are generated randomly from the vast space of possible network structures. Also, the sensitivity and specificity values are very sensitive to the initial condition chosen to simulate the networks and thus the initial state of the genes. For example, for a given gene that is governed by an AND function, we can predict all its inputs if it is active in the wild type state; but only some or none of the inputs will be predicted if the gene is inactive in the wild type state. This variability can also be inferred from the standard errors.

Another similar approach based on predicting a network with minimum number of interactions (Wagner, 2001) has been reported. This algorithm does

not allow for redundant control. In contrast, our approach predicts both the direct and indirect interactions while still implementing the parsimonious condition. In addition, the Wagner approach was mainly developed for acyclic networks. Therefore, we could not obtain a direct comparison with our synthetic networks that are both acyclic and cyclic.

5. Discussion

Our algorithms offer a significant advantage in terms of the scalability with increasing complexity of the networks. The CP and RCP algorithms attempt to distinguish the direct and indirect causality relations from the non-causality interactions. This eliminates the falsely predicted network structures and is reflected in the increased specificity values. Moreover, in certain cases, eliminating the false interactions leads to identification of additional true interactions, and therefore an increase in the sensitivity values. The gain in the sensitivity and specificity values increases as the number of genes increase from 10 to 100 for an indegree of 2 (Table 4). The algorithm also benefits networks with large indegree. Comparable gain in sensitivity and specificity is attained for all networks with indegree ranging from 2 to 8. The apparent decrease in gain with increasing indegree of networks is misleading. Single knockout experiments produce very scarce perturbations in the networks with high indegree, hence only very few interactions can be predicted (note the very low values for the average number of edges in predicted networks). The percentage gain values are normalized with respect to the percentage of edges in predicted networks. This eliminates the corresponding bias due to size of the predicted networks. The percentage of networks for which the current algorithms show an improvement also increases with the number of genes. For the network with 100 genes, all the networks show an increase in the sensitivity and specificity values.

Performance of the algorithms depends on the structure of the target network. For example, if the elements in a network are connected to each other by direct or indirect causality relations only, all three algorithms should predict similar network structures. In contrast, networks with non-causality interactions, such as the one shown in Fig. 1a, will be predicted correctly (sensitivity and specificity values of 100%) only by our CP and RCP algorithms. Ideker's approach would predict incorrect structures such as shown in Fig. 3. Regulatory networks commonly found in biological systems are expected to include non-causality relations. Therefore, we expect the current algorithms to yield a more significant improvement in realistic biological systems, than the average statistics shown in Tables 1 and 2. The average sensitivity and specificity values presented in Table 1 are based on random networks, which include networks with both causal and non-causal interactions. Specifically, the dramatic improvement observed for the network in Fig. 1a, which represents a structure found in a wide variety of transcriptional regulatory networks, demonstrates this point.

6. Conclusions

We present two new algorithms to elucidate the structure of regulatory networks from expression data obtained by perturbing each of the genes in a network. Our results show that these algorithms perform better than existing approaches and significantly reduce the number of miss-predicted edges. Importantly, one of the algorithms can completely eliminate the non-causal interactions. Moreover, the proposed algorithms drastically reduce the number of equivalent networks that are being predicted. One of the major challenges in

Table 4
Comparison of the three algorithms in terms of *F*-factor for different target network configurations

Target network		Average number of edges in		*F*-factor (%)			Gain (%)	Normalized % gain
Number of genes	Indegree	Target network	Predicted network	Causal predictor (CP)	Relaxed causal predictor (RCP)	Ideker's approach		
10	2	12	8	73	73	69	5	8
20	2	27	16	68	68	63	9	15
50	2	72	44	68	68	60	15	24
100	2	150	91	68	68	58	17	29
20	4	62	15	35	35	33	7	29
20	6	90	15	25	25	24	5	31
20	8	120	14	19	19	18	4	33

Note: The average number of edges in the predicted network are a maximum over all three algorithms.
Gain reflects the percentage improvement in terms of *F*-factor values achieved by the CP and RCP algorithms over Ideker's approach. Normalized gain is obtained by dividing the gain by the percentage of edges in the predicted networks compared to target networks.

reverse engineering of any particular regulatory network is to experimentally differentiate between the various plausible networks predicted. By drastically reducing the number of such plausible networks, our algorithms should greatly facilitate the elucidation of the regulatory structure.

Acknowledgments

This work was supported in part by a grant from the National Institute of Health, USA (GM55850-05A1).

References

Aburatani, S., Tashiro, K., Savoie, C.J., Nishizawa, M., Hayashi, K., Ito, Y., Muta, S., Yamamoto, K., Ogawa, M., Enomoto, A., Masaki, M., Watanabe, S., Maki, Y., Takahashi, Y., Eguchi, Y., Sakaki, Y., Kuhara, S., 2003. Discovery of novel transcription control relationships with gene regulatory networks generated from multiple-disruption full genome expression libraries. DNA Res. 10, 1–8.

Akutsu, T., Kuhara, S., Maruyama, O., Miyano, S., 1998. Identification of genetic networks by strategic gene disruptions and gene over expressions. Proceedings of Ninth ACM-SIAM Symposium on Discrete Algorithms (SODA '98), pp. 695–702.

Akutsu, T., Miyano, S., Kuhara, S., 1999. Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. Pac. Symp. Biocomput. 4, 17–28.

Chen, T., He, H.L., Church, G.M., 1999. Modeling gene expression with differential equations. Pac. Symp. Biocomput. 4, 29–40.

Cormen, T.H., 2001. Introduction to Algorithms. MIT Press, Cambridge, MA.

D'Haeseleer, P., Wen, X., Fuhrman, S., Somogyi, R., 1999. Linear modelling of mRNA expression levels during CNS development and injury. Pac. Symp. Biocomput. 4, 41–52.

Friedman, N., Linial, M., Nachman, I., Pe'er, D., 2000. Using Bayesian networks to analyze expression data. J. Comput. Biol. 7, 601–620.

Gardner, T.S., di Bernardo, D., Lorenz, D., Collins, J.J., 2003. Inferring genetic networks and identifying compound mode of action via expression profiling. Science 301, 102–105.

Genoud, T., Trevino Santa Cruz, M.B., Metraux, J.P., 2001. Numeric simulation of plant signaling networks. Plant Physiol. 126, 1430–1437.

Gustin, M.C., Albertyn, J., Alexander, M., Davenport, K., 1998. MAP kinase pathways in the yeast *Saccharomyces cerevisiae*. Microbiol. Mol. Biol. Rev. 62, 1264–1300.

Holter, N.S., Maritan, A., Cieplak, M., Fedoroff, N.V., Banavar, J.R., 2001. Dynamic modeling of gene expression data. Proc. Natl. Acad. Sci. USA 98, 1693–1698.

Ideker, T.E., Thorsson, V., Karp, R.M., 2000. Discovery of regulatory interactions through perturbation: inference and experimental design. Pac. Symp. Biocomput. 5, 305–316.

Kauffman, S.A., 1969. Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. 22, 437–467.

Lee, T.I., Rinaldi, N.J., Robert, F., Odom, D.T., Bar-Joseph, Z., Gerber, G.K., Hannett, N.M., Harbison, C.T., Thompson, C.M., Simon, I., Zeitlinger, J., Jennings, E.G., Murray, H.L., Gordon, D.B., Ren, B., Wyrick, J.J., Tagne, J.B., Volkert, T.L., Fraenkel, E., Gifford, D.K., Young, R.A., 2002. Transcriptional regulatory networks in *Saccharomyces cerevisiae*. Science 298, 799–804.

Liang, S., Fuhrman, S., Somogyi, R., 1998. Reveal, a general reverse engineering algorithm for inference of genetic network architectures. Pac. Symp. Biocomput. 18–29.

Lohr, D., Venkov, P., Zlatanova, J., 1995. Transcriptional regulation in the yeast GAL gene family: a complex genetic network. FASEB J. 9, 777–787.

Maki, Y., Tominaga, D., Okamoto, M., Watanabe, S., Eguchi, Y., 2001. Development of a system for the inference of large scale genetic networks. Pac. Symp. Biocomput. 446–458.

Peng, G., Hopper, J.E., 2002. Gene activation by interaction of an inhibitor with a cytoplasmic signaling protein. Proc. Natl. Acad. Sci. USA 99, 8548–8553.

Raeymaekers, L., 2002. Dynamics of Boolean networks controlled by biologically meaningful functions. J. Theor. Biol. 218, 331–341.

Serra, R., Villani, M., 1997. Modelling Bacterial Degradation of Organic Compounds with Genetic Networks. J. Theor. Biol. 189, 107–119.

Shen-Orr, S.S., Milo, R., Mangan, S., Alon, U., 2002. Network motifs in the transcriptional regulation network of *E. coli*. Nat. Genet. 31, 64–68.

Shymko, R.M., Meyts, P.D., Thomas, R., 1997. Logical analysis of timing-dependent receptor signalling specificity: application to the insulin receptor metabolic and mitogenic signalling pathways. Biochem. J. 326, 463–469.

Sonenshein, A.L., 2000. Control of sporulation initiation in *Bacillus subtilis*. Curr. Opin. Microbiol. 3, 561–566.

Tegner, J., Yeung, M.K., Hasty, J., Collins, J.J., 2003. Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. Proc. Natl. Acad. Sci. USA 100, 5944–5949.

Vance, W., Arkin, A., Ross, J., 2002. Determination of causal connectivities of species in reaction networks. Proc. Natl. Acad. Sci. USA 99, 5816–5821.

Wagner, A., 2001. How to reconstruct a large genetic network from n gene perturbations in fewer than n(2) easy steps. Bioinformatics 17, 1183–1197.

Wahde, M., Hertz, J., 2000. Coarse-grained reverse engineering of genetic regulatory networks. Biosystems 55, 129–136.

Weaver, D.C., Workman, C.T., Stormo, G.D., 1999. Modeling regulatory networks with weight matrices. Pac. Symp. Biocomput. 112–123.

Wolkenhauer, O., 2002. Mathematical modelling in the post-genome era: understanding genome expression and regulation—a system theoretic approach. Biosystems. 65, 1–18.

Zheng, W., Xu, H.E., Johnston, S.A., 1997. The cysteine-peptidase bleomycin hydrolase is a member of the galactose regulon in yeast. J. Biol. Chem. 272, 30350–30355.