Discovering Gene Expression Pathways Using Causal Networks and Experimental Data

Ricardo Silva RBAS@CS.CMU.EDU
Center for Automated Learning and Discovery, Carnegie Mellon University, PA 15213 USA

Abstract

The discovery of gene regulation mechanisms is an extremely valuable alternative to solve several medical and biological problems, ranging from the growth of more resistant and nutritive crops to the cure of cancer. However, given the amount of data and genes on existence, it seems clear that this task turns out to be considerable difficult for these more ambitious goals. We develop a Bayesian network learning algorithm specialized to the specific assumptions we have on this problem. A real world example of gene regulation pathways in the metabolism of galactose on yeast cells is analyzed, exemplifying the kind of outcome that can be expected from our approach.

1. Introduction

Understanding the role played by genes on the cell metabolism leads to important applications such as prevention, identification and treatment of diseases, as well as modifying the genetic structure of living beings in advantageous ways to humans. For instance, Golub et al. (1999) report that patterns of gene activation can be used to trace the evolution of a cancer tumor and aid class prediction (i.e., the cancer type) with better accuracy than traditional techniques that rely on the morphological appearance of the tumor. Ultimately, we would like to know what sets of genes should be controlled to achieve a desired effect.

Creating and modifying current models would require performing tedious and expensive cycles of theoretical formulation of hypothesis followed by experimental analysis. Evidently there is an important place for machine learning procedures for aiding model building. This is not an easy problem. One expects to find a high number of genes present on a normal cell of a given specie and they may interact in long causal chains. This means that identifying the regulatory paths consists on searching on a very large combinatorial space (Edwards & Glass, 2000). Such a search must be conducted wisely, but scarce prior knowledge is often provided.

This paper introduces an approach for causal network induction from experimental data measuring patterns of gene expression. Causal networks provide a suitable language for this domain and also provide mechanisms for dealing with intervening actions, i.e., situations where some variables are *controlled* instead of only observed (Spirtes, Glymour & Scheines, 2000; Pearl, 2000).

In Section 2, the discovery problem is introduced in full detail, as well as the assumptions that make viable finding a solution for it. We also describe the specific type of data that our approach can handle in order to derive useful conclusions. Section 3 provides an extensive description of the algorithm and the kind of information that it is able to provide. Section 4 is a real world experiment conducted on a specific metabolic network of yeast cells. We relate our work with previous approaches for discovery of gene regulation pathways in Section 5, while Section 6 provides suggestions of future work.

2. Problem description

The fundamental mechanisms of gene regulation metabolism have a well-known general structure (Weinzierl, 1999): genes are sequences of DNA. They are copied (transcribed) into messenger RNA, mRNA, which is then spliced to create new mRNA sequences. The mRNA sequences are then translated into chains of amino acids, or proteins. The proteins in turn interact with one another — bind, fold, intertwine, etc. — to form "activated" proteins. Some of the activated proteins in turn control which sequences of DNA are transcribed. The degree of activity of a gene is called g*ene expression*, and the sequence of interactions mediating the effect of one gene in another is usually referred as a regulatory pathway. Our problem can than be rephrased as finding the regulatory pathways for given types of cells.

Other factors in the cell may exert influence over this process. For examples, nutrients such as galactose may be essential for production of some proteins. Figure 1 illustrates a model of this process using a graphical representation. The graph represents a cell under equilibrium, where expression rates are constant through time.

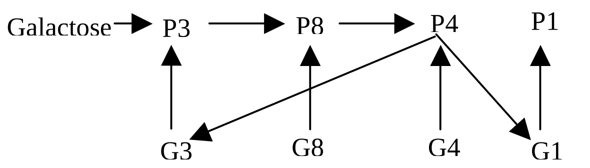

Figure 1. in this simplified model of cell metabolism, *G* nodes represent genes that are either activated or deactivated. *P* nodes represent concentrations of proteins. On this figure, galactose is a nutrient that is required to activate *P3*.

2.1 Modeling

The following definitions are used in our model:

Definition 1 (causal system) *A causal system is a set of random variables.*

Definition 2 (cause) *X is a cause of Y if there are two values x and x' for X such that* p(*Y* | *X is assigned value x*) ≠ p(*Y* | *X is assigned value x'*) *where* p(.) *is a probability function. Notice that this assignment is a manipulation of X, not only conditioning on an observed value of X. In practice, expected values are compared instead of probability functions. For example, we say* G_i *is a cause of* G_j *if the expected value of expression of* G_j *changes when we "knockout" (delete) gene* G_i.

Definition 3 (direct cause) *Let S be a causal system such that* $S = X \cup Y \cup \boldsymbol{O}$, *where X and Y are random variables and* ***O*** *is a set of random variables. If there are two values x and x' for a random variable X, an instantiation o of* ***O*** *such that* p(*Y* | ***O*** = ***o*** *and X is assigned x*) ≠ p(*Y* | ***O*** = ***o*** *and X is assigned x'*), *we say X is a direct cause of Y with respect to causal system S. We can also say that Y is a direct effect of X with respect to S.*

The structure of our model can be formalized as follows.

1. There are three types of components: genes, proteins and cellular environmental factors;
2. These components and the causal relations between them can be represented as a causal Bayesian network, where the usual Markov and Faithfulness conditions apply (Spirtes et al., 2000). The components are nodes in the causal network..
3. The expression level of the genes can be measured, but not the concentration of proteins. The proteins are treated as hidden (but modeled) variables, where we specify in advance the ones that are directed affected by genes;
4. In the true graph, each edge is directed and represents a direct cause;
5. Each gene G_i has only one direct effect: a protein labeled P_i. No other edges going out of G_i are allowed. Since one gene may produce multiple proteins, this node in fact is a simplification, acting as a surrogate for the collective effect its proteins;
6. No P_i directly causes its own gene G_i;
7. Proteins may cause each other and genes;
8. Cycles are acceptable;
9. The rate of expression of a gene is an additive function of its causes;

2.2 Assumptions about available data

There are many ways to obtain measurements of gene expression. In particular, we are interested in *experimental data*, where genes are knocked-out and the effects of these deletions are measured. These experiments are used as oracles able to answer questions such as "Does gene G_i cause gene G_j"? We may also be interested in measuring the effect of some cellular environmental conditions, such as presence of galactose. In this case, we assume these factors are always externally manipulated by the researcher in charge of the experiment, and therefore are treated as *exogenous variables*.

Detecting if a given variable is a direct cause of another is difficult because no measurements of proteins are available. In general, the approach introduced in the next section is able to detect many situations where specific proteins and genes lie on a same direct path, but usually most of the edges will remain unindentified. In some cases, experiments where more than one gene is knocked-out will be available. The following definition formalizes cases where we are able to experimentally detect lack of direct causation.

Definition 4 (direct causality detection experiment) *Assuming that* G_i *causes* G_j, *we say K is a direct causality detection experiment with respect to genes* G_i *and* G_j *if it contains the outcomes of one experiment where we knock-out* $G_i \cup \boldsymbol{C}$ *and one where we knock-out only* ***C***, *where* $\boldsymbol{C} \subset \{Causes(G_j)\} \backslash \{G_i, G_j\}$ *and* $Causes(G_j)$ *is the set of all genes in our causal system that causes* G_j. *We also say that* ***C*** *is the experimental condition set of K. The outcome of a direct causality detection experiment is "yes" if the expected value of* G_j *is different in both experiments, and "no" if they are equal.*

Notice that if *K* answers "yes", it does not mean G_i is a direct cause of G_j with respect to our causal system. We can only say that if all possible direct causality experiments between G_i and G_j answer "yes". Therefore, if we perform some experiment *K* that answers "no", we can say that G_i is not a direct cause of G_j and the set **C** blocks the causal effects of G_i on G_j.

Having available all possible direct causality detection experiments for a pair of genes is very unlikely, since its number increases exponentially with the size of causal system and they are still very expensive.

Table 1. Rules for inference of direct path relationships.

Rule 1: $\forall i, j\ [\mathrm{E}(G_i) \neq \mathrm{E}_{Gj}(G_i)] \Leftrightarrow G_j \rightarrow \ldots \rightarrow G_i$

Knocking out one gene changes the expectation of a second one if and only if the first gene is an indirect cause of the second.

Rule 2: $\forall i, j\ \exists l, m\ [\mathrm{E}(G_i \mid X_j = l) \neq \mathrm{E}(G_i \mid X_j = m)] \Leftrightarrow X_j \rightarrow \ldots \rightarrow G_i$

Changing the level of an exogenous variable changes the expectation of a gene if and only if the exogenous variable is a cause of the gene.

Rule 3: $\forall i, j, k$ [X_k is a cause of $G_i \wedge G_j$ causes $G_i \wedge [\forall l, m\ \mathrm{E}_{Gj}(G_i \mid X_k = l) = \mathrm{E}_{Gj}(G_i \mid X_k = m)]] \Rightarrow$

P_j is on every directed path: $X_k \rightarrow \ldots \rightarrow G$

If an exogenous variable is a cause of a gene, but the exogenous has no effect when a second gene is knocked out, then the protein for the second gene must be a part of every directed path from the exogenous variable to the first gene.

Rule 4: $\forall i, j, k$ [G_j causes $G_i \wedge G_k$ causes $G_i \wedge [\mathrm{E}_{Gk}(G_i) = \mathrm{E}_{Gk\ \&\ Gj}(G_i)]] \Rightarrow P_k$ is on every directed path: $G_j \rightarrow \ldots \rightarrow G_i$

If G_j *is a cause of* G_i*, but* G_j *is not a cause anymore when we knockout* G_k*, then the protein for* G_k *must be present on every directed path from* G_j *to* G_i.. This rule corresponds to a direct causality detection experiment with $C = \{G_k\}$.

Rule 5: $\forall i, j, k, l, m$ [X_k causes $G_i \wedge X_k$ causes $G_j \Rightarrow$

$(\mathrm{E}_{Gj}(G_i \mid X_k = l) - \mathrm{E}(G_i \mid X_k = l)) = (\mathrm{E}_{Gj}(G_i \mid X_k = m) - \mathrm{E}(G_i \mid X_k = m) \Leftrightarrow P_j$ is **not** on any directed path: $X_k \rightarrow \ldots \rightarrow G_i)]$

Knocking out G_j *changes the expectation of* G_i *by the same amount regardless of the value of the exogenous variable (which is a cause of* G_j*) if and only if the protein for gene* G_j *is not a part of any directed path from the exogenous variable to the* G_i. This relationship holds when we assume additivity of effects.

Rule 6: $\forall i, j, k$ [G_j causes $G_i \wedge G_k$ causes $G_i \wedge (\mathrm{E}_{Gj\&Gk}(G_i) - \mathrm{E}_{Gk}(G_i)) = (\mathrm{E}_{Gj}(G_i) - \mathrm{E}(G_i)) \Rightarrow$

P_j is **not** on any directed path: $G_k \rightarrow \ldots \rightarrow G_i$]

This rule is just a variation of Rule 5 applied to genes only. Here, we do not have equivalence, only implication: the antecedent can be false while P_j still does not lie on any directed path between P_k and G_i. That happens when there is a directed path from P_j to P_k. The next rule formalizes this case.

Rule 7: $\forall i, j, k$[G_j causes $G_i \wedge G_k$ causes $G_i \wedge (\mathrm{E}_{Gj}(G_i) - \mathrm{E}(G_i)) \neq (\mathrm{E}_{Gj\&Gk}(G_i) - \mathrm{E}_{Gk}(G_i)) \wedge$

there is no directed path from P_j to $P_k \Rightarrow P_j$ is on **some** directed path: $G_k \rightarrow \ldots \rightarrow G_i$]

If we had a directed path from P_j to P_k, we could not know if the difference is due to the direct effect of P_j on G_i or through an indirect effect that goes through P_k on a path $P_j \rightarrow \ldots \rightarrow P_k \rightarrow \ldots \rightarrow G_i$. Notice also that this rule and rules 11-13 require knowledge that usually is not directly measurable (i.e., if a protein is a cause of another) and therefore should be deduced from other rules

---

Rule 8: $\forall i, j, k$ [G_i does not cause $G_j \wedge G_k$ causes $G_j \Rightarrow$

There is no directed path from P_i to $P_k \wedge P_i$ cannot be on any path $G_k \rightarrow \ldots \rightarrow G_j$]

Rule 9: $\forall i, j, k$ [X_i does not cause $G_j \wedge G_k$ causes $G_j \Rightarrow$ there is no directed path from X_i to P_k]

Rule 10: $\forall i, j, k$ [G_i causes $G_j \wedge G_j$ causes $G_k \Rightarrow P_j$ lies on some path $G_i \rightarrow \ldots \rightarrow G_k$]

$\forall i, j, k$ [X_i causes $G_j \wedge G_j$ causes $G_k \Rightarrow P_j$ lies on some path $X_i \rightarrow \ldots \rightarrow G_k$]

Notice that it is possible to have $i = k$. From our assumptions about the data that is going to be used, we cannot directly measure if one gene causes it self. In our case, we can only discover if G_i causes itself by rules such as Rule 10.

Rule 11: $\forall i, j, k, q$ [P_j is not on any path $V_i \rightarrow \ldots \rightarrow G_q \wedge P_j$ is on some path $P_k \rightarrow \ldots \rightarrow G_q \Rightarrow$

P_k is not on any path $V_i \rightarrow \ldots \rightarrow G_q$].

Rule 12: $\forall i, j, k, q$ [P_j is not on any path $V_i \rightarrow \ldots \rightarrow G_q \wedge P_k$ is on some path $V_i \rightarrow \ldots \rightarrow G_q \Rightarrow$

P_j is not on any path $P_k \rightarrow \ldots \rightarrow G_q$].

Rule 13: $\forall i, j, k, q$ [there is a directed path from V_i to $P_k \wedge P_j$ belongs to some directed path $P_k \rightarrow \ldots \rightarrow G_q$, $\Rightarrow$

P_j belongs to some directed path $V_i \rightarrow \ldots \rightarrow G_q$]

---

3. Approach

We now present a constraint satisfaction approach for automated discovery of causal graphs under the special conditions delineated in Section 2. In Section 3.1, we describe rules that restrict the space of graphs that could explain the behavior of a desired genetic system. An algorithm for constructing a partially directed graph based on these rules is given in Section 3.2, where also we explain the semantics of its representation in full detail. The rules require knowledge of some characteristics of the population that are usually unknown and should be estimated by statistical tests on finite samples. We postpone this discussion till Section 3.4.

3.1 Constraint rules

The rules in Tables 1 describe valid conclusions about the regulatory network given some characteristics that can be estimated by measuring the expression levels of genes under experimental and control conditions. Our general strategy is detecting if:

- there are pathways between a given gene and another;
- a given protein P_j lies on a path between protein P_i, $i \neq j$, and a gene or a third protein;
- a given protein lies on a path between an exogenous factor of the system and a gene;

These rules are not currently known to be complete, but they are independent in a sense they are valid independently of their order of application. A control strategy for usage of these rules is described in Section 3.2. When considered appropriate, we also added an informal description of the semantics of some rules.

By our notation, symbols P, G and X represent proteins, genes and exogenous factors of the system as introduced in Section 2. In a few situations, V_i will be used to represent either a protein P_i or an exogenous factor $X_{(i-N)}$, where N is the number of genes. The function E(.) denotes the expectation function. The function $E_G(Y)$ represents the expected value of Y on a system where we knockout gene G. $E_{G1\,\&\,G2}(Y)$ represents the expected value of Y when we simultaneously knockout two genes, G_1 and G_2. The rules with double knockouts could be easily extended to handle multiple deletions.

The rules should be carried in separate for each state of the background conditions represented by the exogenous variables. The reason why this care should be taken is to avoid anomalous situations where, for example, G_1 causes G_2 under $X_1 = True$ and G_2 causes G_3 under $X_1 = False$. The rules in Table 1 would deduce that G_2 is on some path from G_1 to G_3, even though G_1 and G_2 never cause G_3 at the same time.

It must be clear at this point that this approach is less an induction procedure than an analytical one. The data set can be interpreted as an oracle where relatively straightforward questions are made ("Does G_1 cause G_2? Does G_1 cause G_2 when we delete G_3?"). We are not discovering new causal relations among genes: this is a direct result from the experiments and the statistical procedure with no need to refer to graphical representations. This constraint satisfaction approach is useful for deducing facts that are not so obvious:

- Which causal relations hold among the hidden proteins?
- How could I know if this protein is a direct cause of this gene/protein?
- How could I know if this exogenous factor is a cause/direct cause of this protein?
- Which answers from my oracle are inconsistent given the assumptions?

3.2 Algorithm

Some of the rules from the previous section must be applied multiple times. Also, they only detect which proteins belong to a path from one gene or exogenous factor to another gene, without giving any idea of order. The following algorithm clarifies the procedural order of rule application, describes strategies to identify some order between nodes in a path and introduces a graphical model to represent the output of this procedure.

Let *DirectedPaths* (*DP* for short) be a matrix of dimensionality $(N + Q)\ x\ N$, where N is the number of genes and Q the number of exogenous factors. Each row represents a protein, with the last Q rows representing the exogenous factors. Each column represents a gene. Each element DP_{ij} is a set representing the proteins that appear in a directed path between P_i and G_j (or $X_{(i-N)}$ and G_j). As discussed in Section 2, G_i is a cause of P_i, $1 \leq i \leq N$. If a protein does not appear on cell DP_{ij}, it means there is no path from V_i to G_j that includes P_i. If the symbol $\underline{P_k} \in DP_{ij}$, it means there is a directed path from V_i to P_j that includes P_k. If the symbol $P_k \in DP_{ij}$, it means there is no evidence against the presence of protein P_k in a directed path from V_i to P_j . Both symbols, $\underline{P_k}$ and P_k, cannot appear at the same time in the same cell.

Step 1: Initialization

If G_i or $X_{(i-N)}$ causes G_j (by application of Rules 1 and 2), add to each cell DP_{ij}, $1 \leq i \leq N + Q$, the symbols P_k, (k can be equal to i). Otherwise, set $DP_{ij} = \emptyset$.

Step 2: Build matrix of directed paths

Apply Rules 3 – 6, 8, 9 and 10 to underline or eliminate all symbols P_k from all cells DP_{ij}. Apply repeatedly Rules 7, 11, 12 and 13 to underline or eliminate symbols P_k

from all cells DP_{ij} till no modification can be performed in *DirectedPaths*. The output of this algorithm is a modified *DirectedPaths* table.

Special attention must be paid to rules that concern double knockouts: they can be evaluated only when these experiments are available. Actually, due to the current cost of these experiments, they are likely to be unavailable.

Step 3: Build graph of directed paths and confirmed/discarded edges

DirectedPaths is a data structure used to generate the real output of this algorithm: a graph ***Gr*** that depicts which factors lie on causal chains from one gene to another. The following algorithm is responsible for the construction of such a graph. We use the symbol N_i to represent a generic node on this graph (gene, protein or exogenous factor). We use the symbol DP_{i*} to represent the *i*th row of *DirectedPaths*. We use the symbols DP_{*j} to represent the *j*th column of *DirectedPaths*.

The different types of edges that are created by this algorithm are: $\rightarrow$, $\leftrightarrow$, o—o, o$\rightarrow$, —o. Their semantics are explained on full detail on the next section. During the algorithm, we use asterisks to refer to any kind of edge endpoint: for example, $*\!\rightarrow$ represent any edge of the set {$\rightarrow$, o$\rightarrow$, $\leftrightarrow$}. We also differentiate between two types of arrows at the endpoints of an edge: ***confirmed*** and ***unconfirmed*** arrows.

1. Initialize ***Gr***:
    a. Create a node for each protein, gene and exogenous factor and name them accordingly;
    b. For each pair of proteins $<P_i, P_j>$, create an edge P_i o–o P_j. For all pairs of nodes $<X_i, P_i>$, create an edge X_i –o P_i. For all pair of nodes $<P_i, G_j>$, $i \neq j$, create an edge $P_i \rightarrow G_j$. For each pair of nodes $<G_i, P_i>$, create and edge $G_i \rightarrow P_i$, and mark its endpoint as ***confirmed***. All other arrows are labeled as ***unconfirmed***.
2. If G_i does not cause G_j (*by application of Rule 1*), then eliminate the edge $P_i \rightarrow G_j$.
3. If $DP_{ij} \subset \{P_i, \underline{P_i}, P_j, \underline{P_j}\}$, where V_i is a protein and G_i causes G_j (*by application of Rule 1*), then change the edge between P_i and G_j to a ***confirmed*** edge $P_i \rightarrow G_j$.
4. If $\{\underline{P_k}\} \subseteq DP_{ij} \subset \{P_i, \underline{P_i}, P_j, \underline{P_j}, \underline{P_k}\}$, where V_i causes G_j, V_i does not cause G_k (*by application of Rules 1/2*), then change the edge V_i *—o P_k to an edge $V_i *\!\rightarrow P_k$ and confirm the endpoint at P_k. Also change the arrow from P_k and G_j to a ***confirmed*** one.
5. If $DP_{ij} = \{P_k\}$ or $\{\underline{P_k}\}$, where V_i is an exogenous factor and V_i causes G_j (*by application of Rule 1*), then change the edge between V_i and P_k to an edge $V_i \rightarrow P_k$ with ***confirmed*** arrow.
6. For all cases where Rule 4 apply, eliminate the respective edges $P_i \rightarrow G_j$.
7. If no symbol P_k, $\underline{P_k}$ appears on DP_{i*}, and no symbol P_i, $\underline{P_i}$ appears on DP_{k*}, then eliminate the edge P_i o—o P_k.
8. If $\exists j$ such that $P_k, \underline{P_k} \notin DP_{ij}$; $P_i, \underline{P_i} \notin DP_{kj}$, P_k causes G_j and P_i causes G_j, then eliminate edge P_i o—o P_k.
9. If no symbol P_k, $\underline{P_k}$ appears on DP_{i*}, where node i is an exogenous factor, then eliminate the edge X_i o$\rightarrow$ P_k.
10. If $\exists j$ such that $P_k \notin DP_{ij}$, where node i is an exogenous factor, P_k causes G_j and X_i causes G_j, then eliminate edge X_i o$\rightarrow$ P_k.
11. If $\exists j$ such that $P_k \in DP_{ij}$, then change the edge V_i *—o P_k to $V_i *\!\rightarrow P_k$.
12. If $\exists j$ such that $P_k \notin DP_{ij}$, where node i is a protein factor, P_k causes G_j and P_i causes G_j, then change the edge P_i *—o P_k to P_i *— P_k.

Repeat

13. We say that ***Gr*** has a *semidirected path* between nodes N_i and N_j if: there is an ordered sequence $<N_i, N_k, N_{k+1}, \ldots, N_j>$ of different nodes in ***Gr***; there is an edge in ***Gr*** between each consecutive pair in this sequence such that this edge is not $\leftarrow$. If the only semidirected path from node N_i to node N_j is through the edge $N_i *\!\rightarrow N_j$, then ***confirm*** the arrow at N_j;
14. If $\underline{P_k} \in DP_{ij}$, P_k is on all semidirected paths from V_i to G_j or Rules 3, 4 apply, and if there is no semidirected path from P_q to P_k, but P_q lies on a directed path from V_i to G_j, eliminate any edge between V_i and P_q;

Until no more modifications are possible

15. Output ***Gr***.

By this algorithm, the absence of an edge does not imply there is not a directed path from one node to another: it means that one node is not a direct cause of another. By representing confirmed edges using underlines, the full semantics of all of possible edges in our graph is given in Table 2.

The result is a specific type of partial ancestral graph (PAG). Assuming the information provided by the graph is correct, a genetic engineer would be able to establish strategies to select the manipulations required to achieve the desired overall gene expression behavior in a cell. For example, if we know that gene G_i controls the expression of G_k only through the protein synthesized by G_j, but G_i also affects other genes that we do not want to disturb, one could consider the possibility of manipulating only G_j (if it also does not produce other undesirable side-effects).

3.3 Equivalence classes

Ideally, the graph should be composed of only directed (or double-directed) confirmed edges. But, in most cases, our algorithm is not able to do this. For some sets of assumptions, there are well-known results about how to characterize an equivalence class (Spirtes et al. 2000), but for more general cases no complete description is known. Due to lack of space, we cannot discuss this issue in full detail. The main idea is that if two genes affect exactly the same set of genes, we will not be able to identify the direction of causality between them unless some extra information is provided.

Table 2. Semantics of each type of edge.

TYPE	DESCRIPTION
$N_i \underline{\rightarrow} N_j$	N_i directly causes N_j.
$N_i \quad N_j$	N_i does not cause N_j directly and vice-versa.
$N_i \rightarrow N_j$	There is a directed path from N_i to N_j (N_i causes N_j). N_j does not cause N_i.
N_i o$\rightarrow N_j$	N_i causes N_j, but we do not know if the converse is valid.
N_i –o N_j	There is no directed path from N_j to N_i but we don not know about the opposite.
N_i o–o N_j	We cannot tell the relation of causality between these nodes.
$N_i \leftrightarrow N_j$ $N_i \leftrightarrow\!\!\!\!\!\underline{} N_j$ $N_i \underline{\leftrightarrow} N_j$	There is a directed path from N_i to N_j (N_i causes N_j) and vice-versa. If one (or both) of the endpoints is marked, this implies direct cause for that particular direction.

3.4 Statistical decisions

All rules described in Table 1 require knowledge about some characteristics of the population that are unlikely to be known with certainty. Ideker et al. (2001b) uses a multiplicative-additive error model, a likelihood ratio test is used to evaluate the hypothesis of equal expected values of gene expression under experimental and control conditions. This statistic has asymptotic χ^2 distribution with one degree of freedom under the null hypothesis. Instead of using the theoretical cut-off point for a given Type I probability of error, Ideker et al. empirically determine the cut-off point by running a comparison of two batches of cells under the same non-experimental conditions (where no difference should be noticed) and choosing the threshold where less than 0.1% of the genes of a given cell type rejected the null hypothesis. We adopt Ideker et al. framework on our evaluation in Section 4 due to its solid experimental evidence and because we use a data set where these techniques have been previously applied.

Reporting inconsistencies: our algorithm assumes that the data was generated by a model that satisfies the restrictions introduced in Section 2. However, it may be the case that the true data generating process does not correspond to our assumptions, or that sampling variability lead to inconsistent conclusions. For example, even though causal relations sometimes are not transitive (Pearl, 2000), we are assuming transitivity. If the data report conclusions such as "G_1 causes G_2, G_2 causes G_3 but G_1 does not cause G_3", we report that unexpected result to the user. Other possible inconsistencies are:

- discarding a protein from a path when it was already confirmed on another path;
- confirming a protein when it was already discarded;
- eliminating an edge between two nodes when this edge is the only possible path between two nodes that are confirmed to lie on at least one same path;
- no gene in a experimental condition set that blocks the effect of G_i in G_j causes G_j;

4. Evaluation

Ideker et al. (2001a) recently proposed a framework for building biological models of gene expression. To illustrate this framework, they designed experiments for exploration of the process of galactose utilization in the yeast *Saccharomyces cerevisiae*. A selected subset of 9 genes[1] out of more than 6,000 was manipulated by knocking-out each one at a time under two background conditions of the system: presence and absence of galactose (+gal and –gal, respectively). Using microarrays technology, it was possible to compare the rate of gene expression on each of these 18 experimental conditions and control groups composed of yeast cells in their natural condition (wild type cells). The measurements were taken with the cells in equilibrium.

The data set, however, may produce results with significant high variance: only four measurements are made for each experimental condition. The probabilistic model given in Section 3.4 alleviates part of the problem by using some global parameters that are independent of specific genes (and so measurements for each of the 6,000 genes can be used to estimate them), but ultimately the cut-off point to hypothesis testing was still determined experimentally instead of using the theoretical χ^2 cut-off.

To evaluate our approach, we use the same data set and the same hypothesis testing procedures, as introduced in Section 3.4. One of the reasons this problem was chosen is due to the existence of known partial models for metabolism of galactose for this type of cell, which makes comparisons easier to perform.

Under +gal conditions, our network was composed of a cycle among genes *gal3*, *gal4*, *gal7* and *gal8*, with their respective proteins (none of the other genes had any

[1] The genes are *gal1*, *gal2*, *gal3*, *gal4*, *gal5*, *gal6*, *gal7*, *gal10* and *gal80*.

detectable causal effect). The following statements, which were not obvious from the original data set, were generated:

- no gene in our causal system causes *Pgal4*;
- galactose causes *Pgal4*, *Pgal7* and *Pgal10*, but we could not deduce if it causes or not *Pgal3*;
- *Pgal4* is a direct cause of *gal2*;

Some examples of inconsistencies that were generated:

- *gal10* causes various other genes (e.g., *gal3*), but the double knock-out *gal1+gal10* does not cause any of these genes, and *gal1* alone does *not* cause any of these genes. Ideker et al. (2001) state that the lack of significant different gene expression in *gal3* due to the *gal1+gal10* experiment is an evidence that *gal1* proteins intermediate all the effects from *gal10*, but our representation cannot fit this;
- in a few situations, we found out that accepting that *gal4* does not cause *gal3* (by the statistical test), we come out with warnings as "*gal4* does not cause *gal3*, while G_j causes *gal3*. However, *gal4* also causes G_j". While it is true that causal relations are not always transitive it is at least suspicious when this phenomenon happens. A similar warning is reported for the effect of galactose in *gal3*.

By assuming that *gal4* and galactose cause *gal3*, nearly all inconsistencies disappear. It remains the case about what to do in this situation: right now, we cannot say if this problem is due to sampling variation or incorrect assumptions. The p-values for both causal relationship is quite low, raising the belief that the probability of incorrectness of the data is negligible, but the fact that many inconsistencies could be solved by neglecting this evidence is still suspicious. We are in process of coming to a definitive conclusion. In both cases, the network remains basically the same, but in the latter graph it is confirmed that galactose is a cause of *Pgal3*.

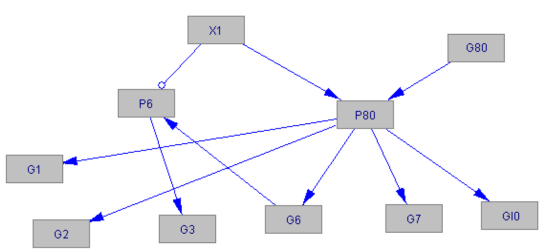

Figure 2. The regulatory network obtained under absence of galactose (represented by variable *X1*).

For the –gal condition, the network is even sparser: only genes *gal6* and *gal80* causes changes in gene expression. Given theses facts, it was trivial to conclude that *Pgal6* is a direct cause of *gal3* and *Pgal80* a direct cause of *gal1*, *gal2*, *gal6*, *gal7* and *gal10*. Due to available double knockouts, it was also possible to discover that galactose is a direct cause of Pgal80. However, by the graph obtained, we ran into another inconsistency.

Gal80 is now an indirect cause of *gal3*, which is not supported by the data. Notice that the graph does show an edge between *Pgal80* and *Pgal6*, which would happen if there was no inconsistency. That is also the reason why the edge between *Galactose* and *Pgal6* has a circle as one endpoint. Again, the probability of *gal80* having a causal relation with *gal3* is low according to the data. We are currently looking for a way to understand the reason of this inconsistency.

The results obtained with this domain are relatively limited. Only one direct cause was discovered in the first graph, and besides some constraints concerning *Pgal4*, the cycle between the remaining proteins can go in any direction. This happened because these proteins cause the same effects (with the exception of *Pgal4*). We know by our discussion on equivalence classes that this situation leads to undetermined edges. If this situation happens frequently in regulatory networks is still an unanswered question. But even under these limitations, the proposed algorithm contributes with a set of tools for the practitioner biologist. Not only the detection of edges is a useful outcome, but also the generation of possible inconsistencies and suggestions for minimizing the number of experiments required for detecting direct causality relations.

5. Related work

Clustering has been consistently applied in gene expression analysis (Eisen et al., 1998). For instance, the work described on (Spellman et al., 1998) is a well-know reference on analysis of yeast cells using a specific clustering method. However, even though clustering may be useful to detect genes that are correlated, one may want to find out how this correlation happens. Questions such as detecting the influence of common causes and direction of gene regulation cannot be answered by cluster analysis alone.

Boolean networks can be used to describe many natural phenomena if one is willing to assume that expression levels are realistically represented by binary variables. The REVEAL algorithm by Liang, Fuhrman and Somogyi (1998) consists in constructing Boolean networks using information-theoretical measures, but without mechanisms to handle noisy data. Even though we did not propose modeling dynamical aspects, we still are able to unveil the cell behavior under stable states (i.e., when gene expression rate does not change on time) without requiring discretization of local effects and non-noisy data.

Friedman et al. (2000) provide one of the first accounts of applying Bayesian networks to gene expression data. Friedman et al. do not use any biological prior knowledge or constraints: they couple with the search space complexity by means of a domain-independent heuristic for parental selection. However, their experiments demonstrated a considerable sensitivity to the kind of parameterization used to represent the local parent-child distributions. Friedman et al. also do not account for the effect of hidden variable such as proteins and using experimental data in conjunction with observational data.

Graphical representations have also been used out of the context of automated discovery and data mining. Ideker et al. (2001) manually incorporate results from experimental analysis into a network representation. Our algorithm can be interpreted as a complement to this manual task, including explicit representation of proteins and deducing information that is not directly obtained only from the experiments and statistical tests. Under a machine learning nomenclature, this can be viewed as a specific form of analytical learning (Mitchell, 1997).

6. Conclusions

The algorithm introduced in this paper performs a constrained search over a space of graphical models representing genetic regulatory pathways. Constraint-based approaches for Bayesian network learning are known to be especially sensitive to sampling variation. However, it is interesting to notice that in our domain the causal conclusions are derived directly from a statistical test that does not have to assume any kind of graphical representation. Assuming that the results of these tests represent the true state of nature (as biologists would, given only the data), constraint-satisfaction algorithms are a powerful tool for identifying inconsistencies between assumptions, data or even additional background knowledge that could be used as an oracle exactly in the same way data was used in our case study in Section 4. It is unclear at this moment how Bayesian estimation methods (Cooper and Herskovits, 1992) would be able to add robustness without requiring an analytical learning procedure to check for inconsistencies.

Due to the novelty of this approach, more extensive evaluations should be performed. We are especially interested in discover if it is common or not to have a group of genes causing exactly the same set of other genes. Incorporation of background knowledge is a way to partially overcome this problem, as well as extra experiments with multiple knockouts. By future experiments and improvements, we expect that eventually derivatives of this procedure will make its way as another standard tool used to further advance our comprehension of the mechanisms of life.

Acknowledgements

David Danks contributed with initial ideas and rules that were essential for this work. I would also like to thank Clark Glymour, Peter Spirtes, Greg Cooper and the Institute For Systems Biology for the helpful discussions and suggestions that helped to improve this work.

References

Cooper, G., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*, 309-347.

Edwards, R., & Glass, L. (2000). Combinatorial explosion in model gene networks. *Chaos*, *10*, 691-704.

Eisen, M., Spellman, P., Brown, P. & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences* 95 (pp.14683-14868).

Friedman, N., Linial, M., Nachman, I., & Pe'er D. (2000). Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, *7*, 601-620.

Golub, T.; Slonim, D.; Tamayo, P.; Huard, C.; Gaasenbeek, M.; Mesirov, J.; Coller, H.; Loh, M.; Downing, J.; Caligiuri, M.; Bloomfield, C. & Lander, E. (1999). Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science*, *286*, 531-537.

Ideker, T.; Thorsson, V.; Ranish, J.; Christmas, R.; Buhler, J.; Bumarner, R.; Aebersold, R. & Hood, L. (2001). Integrated genomic and proteomic analysis of a systematically perturbed metabolic network. *Science* (to appear).

Ideker, T.; Thorsson, V.; Siegel, A. & Hood, L. (2001). Testing for differentially-expressed genes by maximum-likelihood analysis of microarray data. *Journal of Computational Biology*, *7*, 805-817

Liang, S., Fuhrman, S., & Somogyi, R. (1998). REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. *Proceedings of the Pacific Symposium on Biocomputing* (pp.18-29).

Mitchell, T. (1997). *Machine Learning*. McGraw Hill.

Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press.

Spellman, P., Sherlock, G., Zhang, M., Iyer, V., Anders, K., Eisen, M., Brown, P., Botsein, D. & Futcher, B. (1998). Comprehensive indentification of cell cycle-regulated genes of the yeast *Saccharomyces_cerevisiae* by microarray hybridization. *Molecular Biology of the Cell*,*9*, 3273-3297.

Spirtes, P., Glymour, C., & Scheines, R. (2000b). *Causation, Prediction and Search*. 2nd edition, MIT Press.

Weinzierl, R. (1999). *Mechanisms of Gene Expression*. Imperial College Press.