Modeling and Analysis of Heterogeneous Regulation in Biological Networks.

Irit Gat-Viks[*†] Amos Tanay[*†] Ron Shamir[*]

June 6, 2004

Abstract

In this study we propose a novel model for the representation of biological networks and provide algorithms for learning model parameters from experimental data. Our approach is to build an initial model based on extant biological knowledge, and refine it to increase the consistency between model predictions and experimental data. Our model encompasses networks which contain heterogeneous biological entities (mRNA, proteins, metabolites) and aims to capture diverse regulatory circuitry on several levels (metabolism, transcription, translation, post-translation and feedback loops among them).

Algorithmically, the study raises two basic questions: How to use the model for predictions and inference of hidden variables states, and how to extend and rectify model components. We show that these problems are hard in the biologically relevant case where the network contains cycles. We provide a prediction methodology in the presence of cycles and a polynomial time, constant factor approximation for learning the regulation of a single entity. A key feature of our approach is the ability to utilize both high throughput experimental data which measure many model entities in a single experiment, as well as specific experimental measurements of few entities or even a single one. In particular, we use together gene expression, growth phenotypes, and proteomics data.

We tested our strategy on the lysine biosynthesis pathway in yeast. We constructed a model of over 150 variables based on extensive literature survey, and evaluated it with diverse experimental data. We used our learning algorithms to propose novel regulatory hypotheses in several cases where the literature-based model was inconsistent with the experiments. We showed that our approach has better accuracy than extant methods of learning regulation.

[*] School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel. {iritg,amos,rshamir}@post.tau.ac.il.

[†] These authors contributed equally to this work.

1 Introduction

Biological systems employ heterogeneous regulatory mechanisms that are frequently intertwined. For example, the rates of metabolic reactions are strongly coupled to the concentrations of their catalyzing enzymes, which are themselves subject to complex genetic regulation. Such regulation is in turn frequently affected by metabolite concentrations. Metabolite-mRNA-enzyme-metabolite feedback loops have a central role in many biological systems and exemplify the importance of an integrative approach to the modeling and learning of regulation.

In this work we study steady state behavior of biological systems that are stimulated by changes in the environment (e.g., lack of nutrients) or by internal perturbations (e.g., gene knockouts). Our model of the system contains variables of several types, representing diverse biological factors such as mRNAs, proteins and metabolites. Interactions among biological factors are formalized as regulation functions which may involve several types of variables and have complex combinatorial logic. Our model combines metabolic pathways (cascades of metabolite variables), genetic regulatory circuits (sub-networks of mRNAs and transcription factors protein variables), protein networks (cascades of post-translational interactions among protein variables), and the relations among them (metabolites may regulate transcription, enzymes may regulate metabolic reactions). We show how such models can be built from the literature and develop computational techniques for their analysis and refinement based on a collection of heterogeneous high-throughput experiments. We develop algorithms to learn novel regulation functions in lieu of ones that manifest inconsistency with the experiments.

Most current approaches to the computational analysis of biological regulation focus on transcriptional control. Both discrete (e.g., [3]) and probabilistic methods (e.g., [9]) use gene expression data and attempt to learn a regulatory structure among genes and to create a predictive model that fits the data. The computational models used in these studies involve numerous simplifying assumptions on the nature of genetic regulation. Among the more problematic of these simplifications are a) the use of mRNA levels to model the activity of transcription factor proteins, b) the lack of consideration for the state of the medium in which the experiment was done and c) the assumption of acyclic regulation structure that prevents the adequate modeling of feedback loops. As a consequence of these limitations, simple genetic networks tools are rarely used in practical biological settings. A more fruitful approach for learning regulation involves the coarser notion of regulatory modules, with [14] or without [1, 17] explicit learning of regulatory functions that define them. Module-based methods are relatively robust to noise and in some cases can tolerate the gross simplification described above. However, models generated by these methods are coarse and limited in their level of detail.

Our study aims to overcome some of the limitations of prior art by taking an approach that is innovative in combining several key aspects:

- We model a variety of variables types, extending beyond gene network studies, that focus on mRNA, and metabolic pathways methods, that focus on metabolites. Consequently, our model can express the environmental conditions and the effects of translation regulation and post translational modifications.
- Our approach allows handling feedback loops as part of the inference and learning process. This is crucial for adequate joint modeling of metabolic reactions and genetic regulation.
- We build an initial model based on prior knowledge, and then aim to improve (expand) this model based on experimental data. A similar approach was employed in [16] for transcription regulation only. We show that formal modeling of the prior knowledge allows the interpretation of high throughput experiments on a new level of detail.
- Our algorithms learn new transcription regulation functions by analyzing together gene expression, protein expression and growth phenotypes data.

Our methodologies and ideas were implemented in a new software tool called MetaReg. It facilitates evaluation of a model versus diverse experimental data, detection of variables that manifest inconsistencies between the model and the data, and learning optimized regulation functions for such variables. We used MetaReg to study the pathway of lysine biosynthesis in yeast. We performed an extensive literature survey and organized the knowledge on the pathway into a model consisting of about 150 variables. In the process of model construction, we reviewed the results of many low throughput experiments and included in the model the most plausible regulation function of each variable. We assessed the model versus a heterogeneous collection of experimental results, consisting of gene expression, protein expression and phenotype growth sensitivity profiles. In general, the model agreed well with the observations, confirming the effectiveness of our strategy. In several important cases, however, inconsistencies between measurements and model predictions indicated gaps in the current biological understanding of the system. Using our learning algorithm we generated novel regulation hypotheses that explain some of these gaps. We also showed that our method attains improved accuracy in comparison to extant network learning methods.

The paper is organized as follows. In Section 2 we introduce the model and define some notation. In Section 3 we show how to take feedback loops into account and how to use the model to infer the system state given an environmental stimulation. In Section 4 we introduce our mathematical formulation of experimental data and model scoring scheme and in Section 5 we develop optimization algorithms for the learning of regulation functions. Section 6 presents our results on the lysine pathway and its regulation. Some proofs and experimental details appear in an appendix.

2 The model

We first define a formal model for biological networks. A *model* M is a set U of *variables*, a set $S = \{1, \ldots, k\}$ of discrete *states* that the variables may attain, and a set of *regulation functions* $f_v : S^{|N(v)|} \rightarrow S$ for each $v \in U$. f_v defines the state of a regulated variable v (called a *regulatee*) as a function of the states of its *regulator* variables $N(v) = \{r_v^1, \ldots, r_v^{d_v}\}$. We define the set of *stimulators* U_I to include all variables with zero indegree. The *model graph* of M is the digraph $G_M = (U, A)$ representing the direct dependencies among variables, i.e., $(u, v) \in A$ iff $u \in N(v)$. For convenience we assume throughout that regulation functions can be computed in constant time.

A *model state* s is an assignment of states to each of the variables in the model, $s : U \rightarrow S$. A model *stimulation* is an assignment of states to all the model stimulators, $q : U_I \rightarrow S$.

In this paper we shall use the model regulation functions primarily for the determination of modes. For a model M and state s, we say that *s agrees with M on v* if $f_v(s(r_v^1), \ldots, s(r_v^{d_v})) = s(v)$. We call a model state s of M a *mode* if s agrees with M on every $v \in U \setminus U_I$. A mode is thus a steady state of the system. States representing non-steady state behavior of the system, which may be adequate for the representation of temporal processes, are outside the scope in this work. Since our biological models represent a combination of diverse regulation mechanisms, operating in different time scales (metabolic reactions are orders of magnitude faster than transcription regulation), a realistic temporal model is a considerable challenge that should be carefully dealt with in future work. The steady state assumption is in wide use (e.g., [3, 9]) and was proved flexible enough in our empirical studies. Figure 1 illustrates a simple model and its modes.

We now describe the biological semantics of a model. V includes four *types* of variables: (a) mRNAs (b) active proteins that serve as enzymes or regulators (c) internal metabolites, which represent the metabolite derivatives in the pathway under study (d) external metabolites, which represent different environmental conditions and specify the nutritional concentrations in the medium. The external metabolites are assumed to be determined by the experimenter, and their level is unaffected

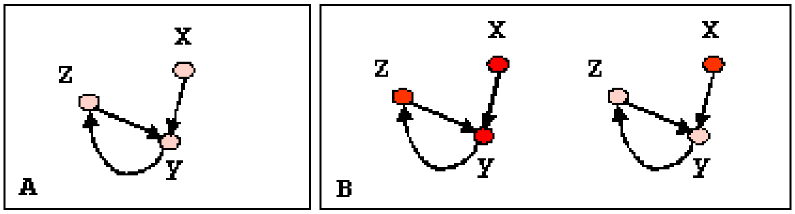


Figure 1: **A simple model**. The model includes one stimulator X, regulating a positive feedback loop of two variables Y and Z. We assume a binary state space (on-dark, off-light). f_z is the identity function and $f_y = s(x)\ AND\ s(z)$. When the stimulator state is off (A), a unique mode exists. If the stimulator state is on (B), two different modes are possible, one in which the cycle is on and the other in which the cycle is off.

by other variables in the model, so they will serve as part of our stimulator set. The levels of the mRNAs, proteins and internal metabolites are controlled by other variables via regulation functions that manifest transcriptional, translational, post-translational and metabolic control mechanisms. The stimulators determine the "boundary condition" of the model. For example, in lysine metabolism, the level of the internal lysine metabolite is influenced by lysine transport into the cell, by the yield of the lysine biosynthetic pathway, by the rate of lysine degradation, and by the rate of lysine utilization in proteins biosynthesis. The external lysine level, on the other hand, is assumed to be determined and kept fixed by the experimenter throughout the experiment.

3 Computing modes

Given a model stimulation q we would like to compute the set of model's modes whose stimulators states coincide with those of q. This will be the first step in using a model to infer the state of the system under a certain condition.

A *q-mode* of a model M and stimulation q is a mode m such that for each $v \in U_I$, $q(v) = m(v)$. We denote the set of q-modes by $Q_{q,M}$. A model M with acyclic graph G_M is called a *simple model.* We note that q-modes are unique and easily computable for simple models: Given a stimulation q and a topological ordering on the graph's nodes (which exists, since the graph is acyclic), we can compute the q-mode by calculating the state of each variable given its regulators' states. In summary:

Claim 1 *Let M be a simple model where $G_M = (U, A)$. For any stimulation q, there is a unique q-mode that can be computed in time $O(|U| + |A|)$.*

In practice, model graphs are not acyclic and feedback loops play a central role in system functionality. In cyclic models, a stimulation q may have no q-modes (in case no steady state is induced by the stimulation), a unique q-mode, or several q-modes. In order to compute the set of q-modes we will first transform a cyclic model into a simple one. Recall that a *feedback set* in a directed graph is a set of nodes whose removal renders the graph acyclic [6]. A feedback set of a model M is a feedback set for the graph G_M. Given a feedback set F, the *auxiliary model* M_F is obtained by changing the regulation functions of the variables in F to null (thus removing all incoming arcs to nodes in F). The graph G_{M_F} is updated accordingly and becomes acyclic, so M_F is simple. Given a set $F' \subseteq F$, we say that a mode m' of M_F is *(M, F')-compatible* if m' agrees with M on every $v \in F'$. In particular, a mode of M_F which is (M, F)-compatible is also a mode for M, since the steady state requirements hold for every $v \in U \setminus F$ (by definition of M_F modes) and for all $v \in F$ (due to the compatibility). Given a mode for M_F, it is easy to check if it is (M, F')-compatible by calculating f_v for each $v \in F'$. The following algorithm calculates the q-modes of M by using a feedback set F and a topological ordering of G_{M_F}:

Mode Computation Algorithm

```
• Generate each possible state assignment to $F$. For the assignment $s_F : F \rightarrow S$ do the following:
    – Generate a stimulation $q'$ for $M_F$ by joining $q$ and $s_F$.
    – Use the topological ordering to compute a (unique) $q'$-mode $m'$.
    – If $m'$ is $(M, F)$-compatible, add it to $Q_{q,M}$.
```

Hence, we have shown:

Proposition 2 *Given a model M, a feedback set F, and a stimulation q, the q-modes can be computed in $O(k^{|F|}(|U| + |A|))$ time.*

We note that the minimum feedback set problem is NP-hard [12], but approximation algorithms are available [15]. The complexity of our algorithm is exponential in the size of the feedback set, but this is tolerable for the current models we have analyzed. Much larger systems may require heuristics that avoid the exhaustive enumeration of feedback set states we are currently using.

4 Experimental conditions and their inferred modes

An ultimate test for a model is its ability to predict correctly the outcome of biological experiments. We formally represent the data of such experiments as *conditions*. A condition e is a triplet (e_q, e_p, e_s). e_q is a model stimulation defining the environment in which the experiment was performed. e_p is a partial assignment of states to variables in $U \setminus U_I$, and is called a *perturbation*. A perturbation defines a set of variables whose regulation was kept as a particular constant during the experiment. For example, knockout experiments fix the state of mRNAs to zero. e_s is a set of measurements of the states of some variables, and is called an *observed partial state*. We define $e_s(v) = -1$ for variables that were not measured in the experiment. Low throughput experiments (like northern blot or ELISA) typically measure one or few variables in a given condition. High throughput experiments (e.g., gene expression arrays or protein expression profiles) may measure the states of all variables of a particular type. A different type of high throughput experiments are growth sensitivity mutant arrays [4]. Each such array corresponds to many conditions, all with the same stimulation (representing the environment of the experiment), but with different perturbations (different knocked-out genes), and only a single measured variable: the growth level. We will assume that this level corresponds to the yield of the metabolic pathway under study.

Given a condition e we wish to use a model M to compare the possible modes induced by the stimulation e_q with the observed partial state. If the condition involves a perturbation, we first have to update our model accordingly. For simplicity assume this is not the case. We then apply the algorithm from the previous section and compute the set of all e_q-modes. In case more than one exists, we expect the correct one to be most similar to the observed partial state. To assess this similarity we introduce a score function that equals the sum of squared differences between the observed partial state e_s and a e_q-mode. Precisely, given a condition e and an e_q-mode s, we define the discrepancy $D(s, e)$ as $\sum_{v \in U, e_s(v) \neq -1} (s(v) - e_s(v))^2$. The mode with smallest discrepancy will be considered as our *inferred mode*. Its score is called the *model discrepancy* on condition e, i.e., $D(M, e) = \min_{s \in Q_{e_q,M}} D(s, e)$. If no e_q-mode exists, $D(M, e)$ is set to a large constant K. Note that models with loosely defined regulation functions may have a large number of modes per stimulation and consequently suffer from over-fitting of the inference.

5 Learning regulation functions

Given a model and experimental conditions, we wish to optimize one particular regulation function in the model and in this way derive an improved model with lower discrepancy. In this section we discuss the resulting function optimization problem, and show that this problem is NP-hard. We translate the function optimization problem to a combinatorial problem on matrices, and provide a polynomial-time greedy algorithm for it. Finally, we show that the greedy algorithm guarantees a 1/2-approximation for a maximization variant of the function optimization problem.

We focus on one model variable v and fix the set of v's regulators $\{r_v^1, ...r_v^{d_v}\}$. Let $E = \{e^i\}$ be the set of experimental conditions. In order to simplify the presentation, we assume throughout this section that experimental conditions have empty perturbation sets. Given a function $g : S^{d_v} \rightarrow S$ we define $M(g, v)$ to be the model M with the single change that $f_v = g$. The *discrepancy score of* g is defined as $\sum_i D(M(g, v), e^i)$.

Problem 1 The function optimization problem. *The problem is defined with respect to a model* M*, a set of conditions* E *and a variable* $v \in U$*. The goal is to find a regulation function* $f_v = g$ *with an optimal discrepancy score. In other words, we wish to compute argmin*$_g \sum_i D(M(g, v), e^i)$.

In most extant gene networks models [9, 3, 16], an optimal regulation function can be easily learned given the topology of the network. This is done using the multiplicities (or probabilities) of different combinations of observed states for the regulators and regulatee. The main difficulty with our version of the learning problem is that the states of regulators are frequently not observed, and have to be inferred together with the regulation function. This is the case, for example, when using gene expression data and trying to learn the regulation of an mRNA variable by unobserved transcription factor (protein) variables. A naive algorithm can test all $k^{k^{d_v}}$ functions for the best discrepancy, but this strategy is impractical even for modest k and d_v ($3^{3^3} > 10^{12}$). In fact, the optimization problem is NP-hard (for a proof, see the appendix).

Proposition 3 *The function optimization problem is NP hard.*

We shall translate the function optimization problem to a combinatorial problem on matrices and develop an approximation algorithm to solve it. First, we define an auxiliary matrix and show how to construct it. We define $Q^v_{q,M}$ as the set of model states s which satisfy for all $u \in U_I$, $s(u) = q(u)$ and agree with M on all $u \in U \setminus U_I, u \neq v$. Note that $Q^v_{q,M}$ is a superset of the set of q-modes $Q_{q,M}$ in which we relax the requirement for agreement on v. Given an instance of the learning problem, we form a matrix W^v with a column for each condition and a row for each assignment of states to v and its regulators. We define the matrix entry $w^v_{i,((x_1,...,x_{d_v}),x)}$ as $\min\{D(s, e^i_s)|s \in Q^v_{e^i_q,M}, s(\overline{r}) = \overline{x}, s(v) = x\}$ or K if the minimization is done over an empty set. where K is a large constant, $\overline{r} = (r_v^1, \ldots r_v^{d_v})$, $\overline{x} = (x_1, \ldots, x_{d_v})$. In the following algorithm, we show how to compute W^v by relaxing the requirement for v compatibility in the mode computation algorithm. Later we shall show how to use W^v to compute the discrepancy score.

Matrix Construction Algorithm

- Initialize all entries in W^v to K.

- Form a feedback set F such that $v \in F$.
- For each condition i and for each assignment s_F of states of the feedback set do:
  - generate a stimulation q' for M_F by joining e^i_q and s_F.
  - use a topological ordering on G_{M_F} to compute a (unique) q'-mode m' for M_F.
  - If m' is $(M, F \setminus v)$-compatible, compute its discrepancy x.
  - Replace the entry $w^v_{i,((m'(r^1_v),\ldots,m'(r^{d_v}_v)),m'(v))}$ by x if the latter is smaller.

Lemma 4 *Given a model M, a set of conditions E and a feedback set F such that $v \in F$, the Matrix Construction Algorithm correctly computes the matrix W^v in $O(k^{d_v+1}|E| + k^{|F|}(|U| + |A|)|E|)$.*

Proof: Matrix entries are computed by minimization of discrepancies over all $(M, F \setminus v)$-compatible modes that have a given regulator/regulatee states. But $(M, F \setminus v)$-compatible modes are exactly the modes in $Q^v_{e^i_q,M}$ which are used in W^v's definition. Therefore, the algorithm correctly computes W^v. The algorithm spends $O(k^{d_v+1}|E|)$ (the size of W^v) time in initialization and $O(k^{|F|}(|U| + |A|)|E|)$ time to compute all mode discrepancies. ■

Lemma 5 *The discrepancy score of a regulation function g equals $\sum_{i=1}^{|E|} \min_{\overline{x}\in S^{d_v}} w^v_{i,(\overline{x},g(\overline{x}))}$.*

Proof: We will show that for each i, $min_{\overline{x}\in S^{d_v}} w^v_{i,(\overline{x},g(\overline{x}))} = D(M(g,v), e^i)$.
$D(M(g,v), e^i) = \min_{s\in Q_{e^i_q,M(g,v)}}[D(s, e^i_s)] = \min_{\overline{x}\in S^{d_v}} \min_{s\in Q_{e^i_q,M(g,v)},s(\overline{r})=\overline{x}}[D(s, e^i_s)] =$
$\min_{\overline{x}\in S^{d_v}} \min_{s\in Q^v_{e^i_q,M(g,v)},s(\overline{r})=\overline{x},s(v)=g(\overline{x})}[D(s, e^i_s)] = min_{\overline{x}\in S^{d_v}} w^v_{i,(\overline{x},g(\overline{x}))}$. ■

By the last lemma, the scores of all possible regulation functions can be derived from the matrix W^v. To find the optimal function we first translate the problem to the following combinatorial problem:

Problem 2 The Rows Subset Cover Problem. *We are given a non-negative integer valued $n \times m$ matrix W and a partition of the rows to disjoint subsets $B_1, \ldots, B_l$. A* row subset *R is a set of rows $b^R_1 \in B_1, b^R_2 \in B_2, \ldots, b^R_l \in B_l$. Our goal is to find a row subset with maximal score $c(R) = \sum_{j=1}^{m} \max_{i=1}^{l} w_{b^R_i,j}$.*

In our settings, rows are pairs $(\overline{x}, x)$ and columns are conditions. The subsets B_j are the sets of rows with identical regulator states $\overline{x}$. To formulate the function optimization problem as a row subset cover problem we rewrite $w_{ij} = K - w^v_{ij}$. A selection of $b_i = (\overline{x}, x)$ corresponds to the setting of $f_v(\overline{x}) = x$.

The previous discussion implies that for constant value of d_v and k, the row subset cover problem is NP-hard. A *Greedy Row Subset Algorithm* applies naturally to this problem: We start with an arbitrary row subset S, and repeatedly substitute a row to improve the score, i.e., setting $S \leftarrow (S \setminus \{b^S_i\}) \cup \{b'_i\}$ where $b'_i \in B_i$ and the new S has improved score. The algorithm terminates in a local optimum when no single row substitution can improve the score. Since the score increases at each iteration and all scores are integers bounded by K, the greedy algorithm will terminate after $O(nmK)$ steps. For the function optimization problem, $O(|E||U|k^2)$ is an upper bound on the maximal score and hence on the number of steps. Each step costs $O(|E|k^{d_v+1})$ in order to find an improving substitution, and thus the total cost is $O(|E|^2|U|k^{d_v+3})$.

Proposition 6 *The greedy algorithm guarantees a 1/2-approximation for the Row Subset Cover Problem.*

Proof: Given a row subset R, the score $c(R)$ can be expressed as a sum of terms of the form $c_j(R) = max_{i=1}^{l}[w_{b_i^R,j}]$. We partition the columns according to $argmax_{i=1,...,l}[w_{b_i^R,j}]$ by defining $P_i^R = \{j|w_{b_i^R,j} \geq w_{b_{i'}^R,j},\ \forall i' \neq i\}$ and transforming $P_1^R, ..., P_l^R$ into a partition by arbitrarily breaking ties. We now have $c(R) = \sum_{i=1}^{l} \sum_{j \in P_i^R}[c_j(R)]$.

Let A be an optimal row subset, and let D be the output of the greedy algorithm. To prove the approximation ratio, we will show that $c(A) - c(D) \leq c(D)$. We first rewrite this inequality using two different column partitions, $\sum_{i=1}^{l} \sum_{j \in P_i^A}[c_j(A) - c_j(D)] \leq \sum_{i=1}^{l} \sum_{j \in P_i^D}[c_j(D)]$. In fact, we will show that the inequality holds separately for each term, i.e., $\sum_{j \in P_i^A}[c_j(A) - c_j(D)] \leq \sum_{j \in P_i^D}[c_j(D)]$.

If $b_i^A = b_i^D$, for each $j \in P_i^A$, $c_j(A) \leq c_j(D)$ since the maximal value for A in column j is $w_{b_i^A,j}$, and solution D can use the same value or a better one. Therefore, $\sum_{j \in P_i^A}[c_j(A) - c_j(D)] \leq 0$ and the inequality holds.

Assume now that $b_i^A \neq b_i^D$. We define a new row subset E_i, which is the same as D except for replacing b_i^D with b_i^A. The replacement decreases the score in certain columns (denoted L^-) and increases it in others (denoted L^+). Call the decrease in the set L^- the *loss* and call the increase in L^+ the *gain* caused by the replacement. The key to the proof will be the fact that the loss must be equal or larger than the gain, or else the greedy algorithm would have not stopped at D. Since $L^- \subset P_i^D$ we have

$$\sum_{j \in P_i^D}[c_j(D)] \geq \sum_{j \in L^-}[c_j(D) - c_j(E_i)] = loss. \tag{1}$$

Next look at L^+ and P_i^A. For $j \in L^+ \setminus P_i^A$ we have $c_j(E_i) - c_j(D) > 0$. For $j \in L^+ \cap P_i^A$ we have $c_j(E_i) - c_j(D) = c_j(A) - c_j(D)$. For $j \in P_i^A \setminus L^+$ we have $c_j(A) - c_j(D) < 0$. Overall we get:

$$gain = \sum_{j \in L^+}[c_j(E_i) - c_j(D)] \geq \sum_{j \in P_i^A}[c_j(A) - c_j(D)]. \tag{2}$$

In summary, $\sum_{j \in P_i^D}[c_j(D)] \geq \sum_{j \in L^-}[c_j(D) - c_j(E_i)] \geq \sum_{j \in L^+}[c_j(E_i) - c_j(D)] \geq \sum_{j \in P_i^A}[c_j(A) - c_j(D)]$. The first and third inequalities follow by (1) and (2), respectively. The second inequality is the observation that the loss exceeds the gain. ■

In practice, we find regulation functions by executing the matrix construction algorithm and applying the greedy algorithm to the obtained matrix. Note that our approximation holds only for the maximization problem. Developing any constant ratio approximation for the minimization problem in its current form is NP-hard. This follows from Proposition 3, since the matrix corresponding to the model in that proof has a minimum score of zero, and thus the decision problem with score zero is NP-hard.

We note that in order to take condition perturbations into account, we have to consider a slightly different model in each condition. For example, if a condition was measured in a strain knocked-out for a specific gene v, we will form a modified model with altered (constant) f_v function and compute its modes and discrepancy as described above. The other algorithms (matrix generation and row selection) remain unchanged.

6 Results

We applied the *MetaReg* modeling scheme and algorithms to study lysine biosynthesis in the yeast *S. cerevisiae*. This system was selected since a) it is a relatively simple metabolic pathway, b) its regulatory mechanisms are relatively well understood, and c) several high throughput datasets which include experimental information pertinent to lysine biosynthesis are available.

6.1 A Model for Lysine Biosynthesis

We have performed an extensive literature survey and constructed a detailed model for lysine biosynthesis and related regulatory mechanisms. Lysine, an essential amino acid, is synthesized in *S. cerevisiae* from α-ketoglutarate via homocytrate and α-aminoadipate semialdehyde (αAASA) in a linear pathway in which eight catalyzing enzymes are involved. The production of lysine is controlled by several known mechanisms:

(1) Control of enzymes transcription via the general regulatory pathway of amino acids biosynthesis. Starvation for amino acids, purines and glucose, induce the synthesis of GCN4ap [1] which is a transcriptional activator of enzymes catalyzing amino acids biosynthesis in several pathways, including lysine. GCN4ap is controlled on the translation level by the translation initiation machinery. Specifically, GCN2ap (a translation initiation factor 2α kinase) is known to mediate the de-repression of GCN4m translation in nutrient-starved cells. The activity of GCN2ap is induced by high levels of uncharged tRNA under starvation conditions [5].

(2) Transcription control of several catalyzing enzymes is regulated by αAASA. The control is mediated by the LYS14ap transcriptional activator in the presence of αAASA, an intermediate of the pathway acting as a coinducer. αAASA serves as a sensor of lysine production [13].

(3) Feedback inhibition of homocytrate synthase isoenzymes (LYS20ap and LYS21ap) by lysine. The first step of the lysine biosynthetic pathway is catalyzed by LYS20ap and LYS21ap. At high levels of lysine, LYS20ap and LYS21ap are inhibited, and thus the production of the pathway intermediates and of lysine itself is reduced [8].

(4) MKS1ap down-regulates CIT2m expression and hence cytrate-synthase production which is needed for the synthesis of α-ketoglutarate. The resulting limitation of α-ketoglutarate decreases the rate of lysine synthesis. MKS1ap is activated in nutrient-starved cells [7, 18].

In Figure 2, we present the model graph of lysine biosynthesis as described above. The graph includes the lysine biosynthetic pathway, the catalyzing enzymes and their transcription control, and the translation initiation machinery controlling GCN4ap state. The model includes also external amino acids and ammonium (NH3). These are needed as stimulators to represent the environmental conditions enforced on the system. The transport of amino acids and ammonium into the cell is facilitated via specific permeases, and the level of internal amino acids and ammonium is determined by the extracellular metabolites and by the activity of these permeases. The state of internal lysine depends on the lysine transport into the cell and on the yield of the lysine biosynthetic pathway. Note that in order to study the model in relative isolation from other pathways and regulatory systems, we had to exclude some of the known relations (e.g., CIT2 and the Kreb cycle in α-ketoglutarate production, tRNAs in GCN2ap activation). The model graph contains several cycles that correspond to three distinct feedback cycles: general nitrogen control regulation (e.g. GCN2ap → GCN4ap → LYS1,9m → LYS1,9ap → ILys → GCN2ap), lysin negative regulation (LYS20ap/LYS21ap → IHomoCytrate → αAASA → ILys → LYS20ap/LYS21ap) and αAASA positive regulation (e.g. LYS14ap → LYS2m → LYS2ap → αAASA → LYS14ap). We used a feedback set F consisting of $GCN2ap$ and IαAASA in all the computations reported below. The complete and annotated list of regulation functions that are part of the model are available from our web site (www.cs.tau.ac.il/∼rshamir/metareg/).

We used the state space $S = \{0, 1, 2\}$. In our experiments, the definition of compatibility used for the calculation of q-modes was relaxed a bit to include also cases where $m'(v)$ and $f_v(m'(r_v^1), \ldots, m'(r_v^{d_v}))$ are both non-zeros (i.e., cases where inferred state was 1 and observation 2 or vice versa are not considered violation of compatibility). In other words, $D(i, j)$ was $(i - j)^2$ for all states $\{i, j\} \neq \{1, 2\}$,

[1] We use variable affixes to indicate types. m suffix: mRNA, ap suffix: active protein. Metabolites names are prefixed to indicate their type, I: internal, E: external.

but $D(1,2)$ and $D(2,1)$ were set to 0. This was done to allow more flexibility in the model and to focus more on major discrepancies.

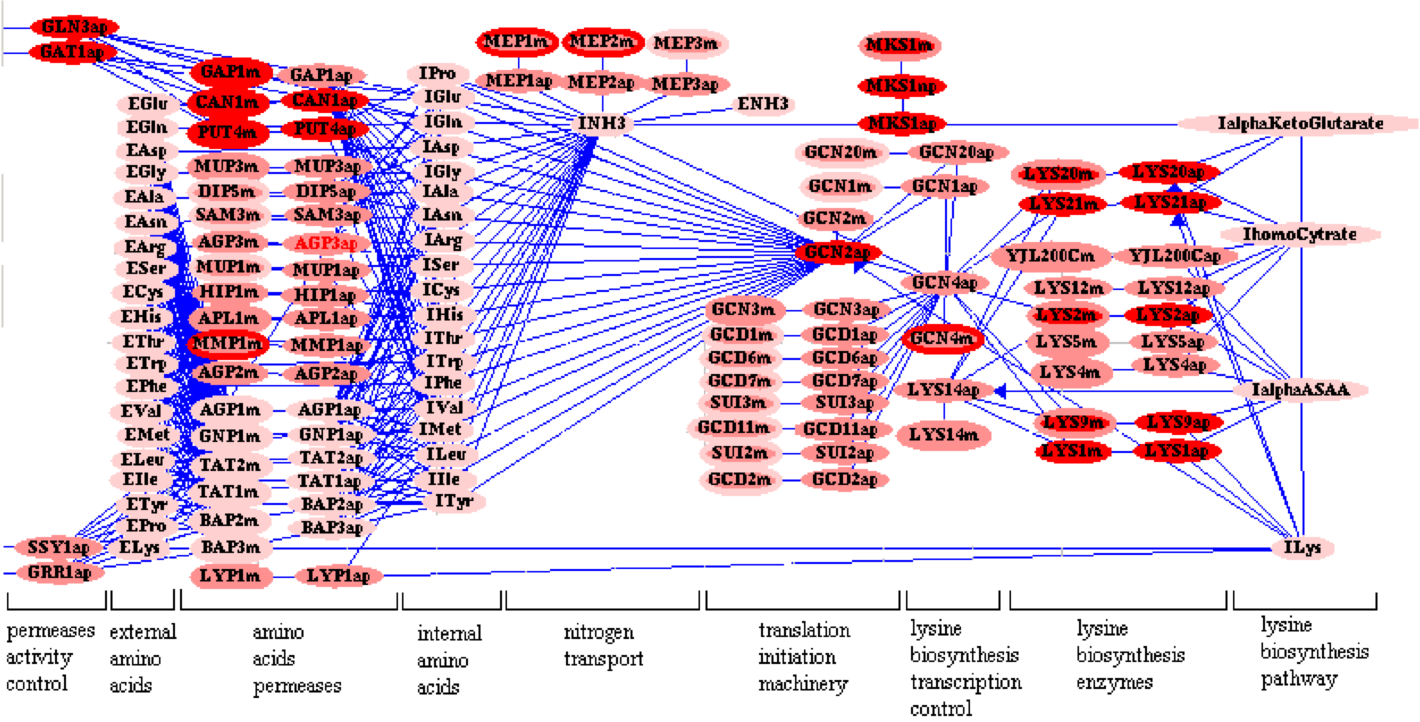


Figure 2: **The model graph of lysine biosynthesis in S. cerevisiae.** Variables are represented by nodes. Arcs lead from each regulator to its regulatees. All arc directions are at any angle to the right or straight down, unless otherwise indicated. The model includes also a regulation function for each regulated variable. These functions are not shown here. Node colors indicate the mode inferred states and the observed states in condition of nitrogen depletion after 2 days. Internal node color: inferred state. Node boundaries: observed state. Red(dark): state= 2. Dark pink(grey): 1. Light pink(light grey): 0. The representation enables us to view the disagreements as color contrasts between the observed and inferred states. For example, LYS9m (bottom right) inferred state is 2 while its observed state is 1.

6.2 Data Preparation

We formed a heterogeneous dataset from five different high-throughput experiments: (a) 10 expression profiles in nitrogen depletion medium after 0.5h, 1h, 2h, 4h, 8h, 12h, 1d, 2d, 3d, 5d of incubation [10]. (b) 5 expression profiles in amino acid starvation after 0.5h, 1h, 2h, 4h, 6h of incubation [10]. (c) 10 microarray experiments of His and Leu starvations and various GCN4 perturbations [5]. (d) protein and mRNA profiles of wild type strain in YPD and minimal media [19]. (e) 80 Growth sensitivity phenotypes [4]. The growth phenotypes were measured for each of a collection of ten gene-deletion mutant strains in eight conditions: Lys, Trp and Thr starvation, three minimal media and two YPG conditions.

To incorporate these data into our framework, we generated conditions from each of the experiments. To this end, we identified the stimulation and perturbation that match each experiment from the respective publication. We then converted the data into a set of observed states. In the Appendix, we define the conversion process of each data set. Note that some of the experiments translate directly to observed states (e.g., growth profiles) and some must be manipulated further (e.g., mRNA ratios of two conditions).

6.3 Model discrepancy

For each of the high throughput conditions in (a) through (d) we computed inferred modes and compared them to the observed states. Recall that the environment defined by the condition's stimulation gives rise to a set of possible inferred modes, and we choose the inferred mode which fits the observed states best. Typically, there are only few modes per condition in the lysine model, confirming the relatively good characterization of the system by the model.

Figure 3A summarizes the comparison between inferred modes and observed states for expression conditions. Figure 3B does the same for growth sensitivity data. In general, there is good agreement between the inferred and observed states. The matrix view highlights conditions and variables in which the observations deviate from the model predictions.

Before analyzing the deviations, we verified the specificity of the total discrepancy. Since the mode computation algorithm involves selection of one mode from several possibilities in each condition, we wanted to verify that this process does not cause over-fitting. To this end, we generated randomly shuffled data sets in which we swapped the states between variables of the same type. Figure 3C shows the discrepancy distribution obtained from this experiment, and supports the high specificity of the lysine model discrepancy.

We next examined the biological implication of two major deviations of the inference from the experimental data: First, the transcription of the translation initiation machinery (GCD1,2,6,7,11, GCN1,20, SUI2,3) is repressed in the later phases (8h-5d) of the nitrogen depletion experiment, and this effect is not predicted by the model. Moreover, the transcription of the ammonium permeases MEP1 and MEP2 is consistently activated in nitrogen depletion. To the best of our knowledge, the explanation for these observations is still unclear. However, there is some evidence for involvement of the TOR signaling pathway in the regulation of this response [2]. Second, the transcription of the lysine biosynthesis catalyzing enzymes is known to be activated by both LYS14ap and GCN4ap, but the exact combinatorial regulation function is unknown. Since they are both known to be activators, we originally modeled the regulation function of the catalyzing enzymes (LYS1,2,9,20,21) simply as the sum of LYS14ap and GCN4ap. In most catalyzing enzymes, there is a clear inference deviation in two conditions with GCN4Δ strain (Figure 3A, 3rd and 6th columns from right). In addition, the growth phenotypes of LYS14 deletion strain (Figure 3B, second row) deviate from their inferred states in all conditions with nutritional limitation of lysine. Therefore, the regulation function we originally modeled for the lysine biosynthesis catalyzing enzymes is apparently not optimal.

6.4 Learning improved regulation functions

To refine our understanding of the combinatorial regulation scheme involving LYS14ap and GCN4ap we applied our learning algorithm to the regulation functions of LYS1,2,4,5,9,12,20,21. For each one, we computed the discrepancy matrix and selected an optimal regulation function using the learning algorithm outlined in Section 5. To estimate the confidence of our learned functions we used a bootstrap procedure as follows. We generated 1000 datasets each containing a random subset of 80% of the original set of conditions. For each random dataset we recalculated the optimal regulation functions for each of the enzymes. The *confidence* of the function entry $f_v(x_1, \ldots, x_{d_v}) = y$ was defined as the fraction of times y was learned as the function value for the regulators values $x_1, \ldots, x_{d_v}$. In case of ties (several function outcomes with equal scores), we split the count among the candidate outcomes. Results are summarized in Figure 4A,B,C.

Based on the optimal functions, we identify two enzyme sets that share a regulatory program. The expression of genes in the first set (LYS1,9,20 and possibly LYS4 and LYS21) is dependent on the presence of both LYS14 and GCN4. Both transcription factors seems to drive the transcription of enzymes in this set linearly. The second set, including LYS5, LYS12 and YJL200C require LYS14 but not GCN4 for basal expression levels. For LYS5 it seems that GCN4 may not be a regulator at all, possibly since LYS5 is not a catalyzing enzyme in the pathway under study. We note that the combination of expression and growth phenotype information was crucial for deriving this conclusion. For example, when using expression data alone, the rows with LYS14p=0 are completely undefined.

6.5 Cross validation

We tested the predictive quality of *MetaReg* by performing leave-one-out cross validation. For the test, we used the set of enzymes $L = \{\text{LYS1,2,4,5,9,12,20,21m}\}$ as regulatees and GCN4ap, LYS14ap, as regulators. For each variable $v \in L$ and each condition c, we optimized the regulation function of v while fixing the rest of the model and hiding the data of c. We then used the optimized model to infer the mode in condition c without using the observed value of v. Finally, we compared the inferred state of the enzyme variable to the observed one, and counted the total number of correct outcomes (or fractions of outcomes in case the inferred mode was ambiguous and several alternatives existed). Using mRNA expression data only, the accuracy derived in this procedure was 78.3% (Figure 4D).

We compared the performance of Metareg to the following alternative methods: (a) A Bayesian networks [9] with a known structure where GCN4m and LYS4m are the parents of each variable in L. We learned the local probability parameters [11] using non-informative prior. To compute the accuracy, we ran a cross validation test by learning parameters while hiding one condition at a time. The overall accuracy obtained in this procedure was 61.4%, much lower than achieved by *MetaReg*.

(b) An independence model: Each regulatee in L has no regulators. We predict the probability of each regulatee outcome as the background distribution of its observations. To compute the accuracy, we ran the same procedure as in (a). The overall accuracy obtained in this procedure was 47.5%. We conclude that the detailed modeling of interactions among proteins, metabolites and mRNAs gives an improved accuracy to our model.

Note that in all cross validation tests, we used exactly (and only) the same gene expression data. We thus prevented metareg from utilizing the additional information it can use (e.g. phenotypes, protein expression), which is essential for discovering several aspects of the regulatory network.

7 Discussion

Models of biological regulation are becoming increasingly complex. The well established biological methodology of model development and expansion (incremental refinement) is facing major challenges with the advent of high throughput technologies and the discovery of more and more regulatory mechanisms. Computational techniques for modeling and learning biological systems are currently limited in their ability to help biologists to extend their models: De-novo reconstruction methods ignore available biological knowledge, and module-based methods do not specify concrete regulation functions. Here we aim at the construction of a computational methodology that combines well with current biological methodologies. MetaReg models can be built for almost any existing biological system, they do not assume complete knowledge of the system, and are flexible enough to integrate diverse regulatory mechanisms. Once built, the model allows easy integration of high throughput data into the analysis of the existing model. The computational tools introduced here can then be used to generate testable and easy to understand biological regulation hypotheses.

There are many limitations that are currently not handled by our methods and are subject to future work. At the modeling level, there is a need to formulate uncertainties more rigorously and allow the biologist to state which features in the model are well established and which are not. At the inference level, metareg is currently limited to exploration of strict steady states. This makes the true modeling of some biological processes difficult, as it may be necessary to tailor the models so that attractors of size greater than one does not exist. As for the learning algorithms, we have only studied the most basic problem of learning a single regulation function. Algorithms for simultaneous learning of multiple regulation functions and for model structure refinements should still be developed.

Acknowledgment

IGV was supported by a Colton fellowship. AT was supported in part by a scholarship in Complexity Science from the Yeshaia Horvitz Association. RS was supported in part by the Israel Science Foundation (Grant 309/02) and by the EMI-CD project that is funded by the European Commission within its FP6 Programme, under the thematic area ”Life sciences, genomics and biotechnology for health”, contract number LSHG-CT-2003-503269. ”The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.”

References

[1] Z. Bar-Joseph, G.K. Gerber, T.I. Lee, N.J. Rinaldi, J.Y. Yoo, F. Robert, D.B. Gordon, E. Fraenkel, T.S. Jaakkola, R.A. Young, and D.K. Gifford. Computational discovery of gene modules and regulatory networks. *Nature Biotechnology*, 21:1337–1342, 2003.

[2] J. L. Crespo and M. N. Hall. Elucidating TOR signaling and rapamycin action: Lessons from *S. cerevisiae*. *Microb. Mol. Biol. Rev.*, 66:579–591, 2002.

[3] P. Dhaseleer, S. Liang, and R. Somogyi. Genetic network inference: from co-expression clustering to reverse engineering. *Bioinformatics*, 16:707–726, 2000.

[4] G. Giaever et al. Functional profiling of the *S. cerevisiae* genome. *Nature*, 418:387–391, 2002.

[5] K. Natarajan et al. Transcriptional profiling shows that GCN4p is a master regulator of gene expression during amino acid starvation in yeast. *Mol. Cell. Biol.*, 21:4347–4368, 2001.

[6] S. Even. *Graph Algorithms*. Computer Science Press, Potomac, Maryland, 1979.

[7] A. Feller, F. Ramos, A. Pierard, and E. Dubois. LYS80p of *S. cerevisiae*, previousely proposed as a specific repressor of LYS genes, is a pleiotropic regulatory factor identical to Mks1p. *Yeast*, 13:1337–1346, 1997.

[8] A. Feller, F. Ramos, A. Pierard, and E. Dubois. In *S. cerevisiae*, feedback inhibition of homocitrate synthase isoenzymes by lysine modulates the activation of LYS gene expression by LYS14p. *Eur. J. Biochem.*, 261:163–170, 1999.

[9] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *J. Comp. Biol.*, 7:601–620, 2000.

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

[11] D. Heckerman, D. Geiger, and D.M. Chickering. Learning baysian networks: the combination of knowledge and statistical data. Technical Report MSR-TR-94-09, Microsoft research, 1995.

[12] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103, New York, 1972. Plenum Press.

[13] F. Ramos, E. Dubois, and A. Pierard. Control of enzyme synthesis in the lysine biosynthetic pathway of *S. cerevisiae*. *Eur. J. Biochem.*, 171:171–176, 1988.

[14] E. Segal, M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat Genet.*, 34(2):166–76, 2003.

[15] P. D. Seymour. Packing directed circuits fractionally. *Combinatorica*, 15:281–288, 1995.

[16] A. Tanay and R. Shamir. Computational expansion of genetic networks. *Bioinformatics*, 17:S270–S278, 2001.

[17] A. Tanay, R. Sharan, M. Kupiec, and R. Shamir. Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genome-wide data. *Proc. Natl. Acad. Soc.*, 101:2981–2986, 2004.

[18] J. J. Tate, K. H. Cox, R. Rai, and T. G. Cooper. Mks1p is required for negative regulation of retrograde gene expression in *S. cerevisiae* but does not affect nitrogen catabolite repression-sensitive gene expression. *J. Biol. Chem*, 277:20477–20482, 2002.

[19] M.P. Washburn. Protein pathway and complex clustering of correlated mRNA and protein expression analyses in *S. cerevisiae. PNAS*, 100:3107–3112, 2003.

8 Appendix 1: Preprocessing the Data

8.1 Microarray data

Gene expression measurements using cDNA microarrays do not provide absolute mRNA levels in a single condition of interest, but only relative mRNA levels in pairs of conditions. Hence, an experiment gives for each gene the ratios of its mRNA levels in a *test* and *reference* conditions. In order to use such data in our analysis, we first have to transform experiments into absolute mRNA levels in single conditions and then use them to evaluate model discrepancy. This can be easily done if all experiments are using a common reference condition. In this simple case, we may directly use the relative measurements as the observed states in the test condition, due to the fact that all relative measurements are comparable.

In practice, not all experiments use the same reference condition. To handle this case we define the *experiments graph*, a directed multigraph in which vertices represent conditions and arcs represent experiments. For each experiment, we add an arc from the reference to the test condition vertex. Note that condition vertices can be reused. Let $l(v,i,j)$ be the logarithm of the ratio of gene v's levels in condition i and condition j. For gene v, the weight of arc (i,j) is precisely $l(v,i,j)$, as obtained from the microarray in the experiment with test condition j and reference condition i. We wish to compute a normalized log hybridization level (we will refer to it as *level*) for each mRNA variable in each condition. Assume first that the graph is connected. The idea is to fix one vertex in the graph as a common reference and compute the levels of all other conditions relative to it. We then discretize the levels to generate the observed states of each condition.

Our normalization procedure works as follows: First, using prior biological knowledge we fix the levels of the mRNA variables in one condition (the *source condition*). Second, we use a breadth-first search algorithm on the underlying undirected experiment graph in order to handle condition vertices by an ascending distance from the source condition. When reaching a condition i, there must be at least one neighbor whose levels are already determined. We compute the level of each mRNA variable v in the condition i as follows: The *contribution* of an incoming neighbor j is defined as $l(v,j)+l(v,j,i)$ where $l(v,j)$ is the level of variable v in condition j. For an outgoing neighbor j, the contribution is defined as $l(v,j)-l(v,j,i)$. We compute the level of v at condition i by averaging the contributions of all determined neighbors. In case more than one connected component exists, we must fix a common reference in each component and ensure their levels are comparable. Note that if we have absolute measurements on more than one condition in a connected component of the experiment graph, the algorithm can be adapted to take this information into account.

In order to calculate the observed states for datasets (a)-(d) (Section 6.2), we created the experiment graph shown in Figure 5. We used growth of a wild-type strain in standard conditions and YPD medium as our source condition, and fixed its levels to 1 for all genes. We assigned the computed levels in the range $(-\infty,0)$,$[0,2)$ and $[2,\infty)$ to the observed states 0,1 and 2, respectively.

8.2 Protein data

The observed states of proteins in minimal media were obtained by discretizing the concentration levels reported by [19]. We assigned the values in the range $[0,0.5]$,$(0.5,1.5]$, and $(1.5,\infty)$ to the observed states 0,1 and 2 respectively.

8.3 Phenotype Data

The observed states of the growth phenotypes were obtained by discretizing the sensitivity scores reported in [4]. Giaever et al. (2002) computed for each strain and environmental condition a sensitivity score, and assigned the sensitivity scores in each generation time a cutoff of significance. Following these cutoffs, we discretized the sensitivity scores in the range [0,10],(10,20], and (20,∞) to observed states 0,1, and 2, respectively for 5 generation phenotypes, and [0,20],(20,100],(100,∞) for 15 generation phenotypes. The growth phenotype was associated with the level of the internal lysine variable in the model.

9 Appendix 2

Proposition 7 *The function optimization problem is NP hard.*

Proof: The decision version of the problem is to determine if there is a function g with $\sum_i D(M(g,v),e^i)$ $\leq L$. We shall reduce 3SAT to it. The idea is to encode a truth assignment of n variables using a function on $\lceil \log n \rceil$ regulators, where each assignment of regulators values encodes one variable and the regulatee value represents the truth value for that variable. For each clause we will add a model variable and a condition, such that the model discrepancy will be zero iff the function encoding the truth assignment will set at least one of the clause literals to its required truth value. For an example of the construction see Figure 6.

We now describe the construction formally. We are given an instance of 3SAT with a set of m clauses $C_1, ..., C_m$, involving the variables $x_1, ..., x_n$. Define first some auxiliary variables: For the j-th literal of clause i, we let a_{ij} equal the index of the variable, and we set $b_{ij} = 0$ if that variable is negated and $b_{ij} = 1$ otherwise. In our example (Figure 6), $C_2 = \neg x_1 \vee \neg x_2 \vee x_3$ and $a_{20} = 1, a_{21} = 2, a_{22} = 3, b_{20} = 0, b_{21} = 0, b_{22} = 1$.

We are now ready to construct the model M. All states are Boolean. The model variables include a) a set of $d_v = \lceil \log_2 n \rceil$ variable pairs r_i, r'_i that constitute a *variable encoding gadget.* b) a variable c_i for each clause C_i and c) a *truth assignment* variable v, to which we will apply function optimization. The variable encoding gadget is built so that each combination of states of its variables corresponds to one of the SAT variables. We introduce a positive feedback loop between r_i and r'_i so that in each mode both can have either 0 or 1 state. The regulation function of each clause variable c_i encodes the logic of that clause. We let $N(c_i) = (r_1, \dots, r_{d_v}, v)$ and set f_{c_i} to zero in all but three particular regulators states, corresponding to each of the clause literals, in which the regulatee value is set to 1. The positive regulators state for the j'th clause is determined by the binary representation of a_{ij} in the variable encoding gadget and by requiring v to attain the corresponding b_{ij} value. In other words, f_{c_i} is true only when the variable encoding gadget encodes one of the variables in C_i and v has a value which is compatible with that variable sign in C_i. v's regulators are all the r_i's. To finish the construction, we must specify the set of conditions E. It consists of one condition e^i per clause C_i, in which c_i is observed as 1 and all other variables are hidden. Finally, set $L = 0$.

We claim that there is a function g s.t. $\sum_i D(M(g,v),e^i) = 0$ iff the given instance of 3SAT is satisfiable. We map between truth assignments $T = \{t_j\}$ and functions g^T by setting $g^T(x_1, \dots, x_{d_v}) \equiv t_j$ where j is the integer with binary encoding $x_1, \dots, x_{d_v}$. We will prove that all clauses are satisfied by T iff $\sum_i D(M(g^T,v),e^i) = 0$.

Let T be a truth assignment that satisfies the 3SAT instance. Then in each clause C_i there is a variable a_{ij} with sign b_{ij} that is true. Consider the encoding of the index a_{ij} in the variables $r_1, \dots, r_{d_v}$. By the definition of g^T, the state of v equals $t_{a_{ij}}$, and hence by construction f_{c_i} will equal one, giving zero discrepancy for the condition i. Hence the overall discrepancy is zero.

Suppose conversely that the total discrepancy is zero for the function g^T, and consider the truth assignment T. As the discrepancy for experiment e^i is zero, there is an index a_{ij} whose encoding along with b_{ij} correspond to a value 1 of f_{c_i}. Hence, clause C_i is satisfied. ■

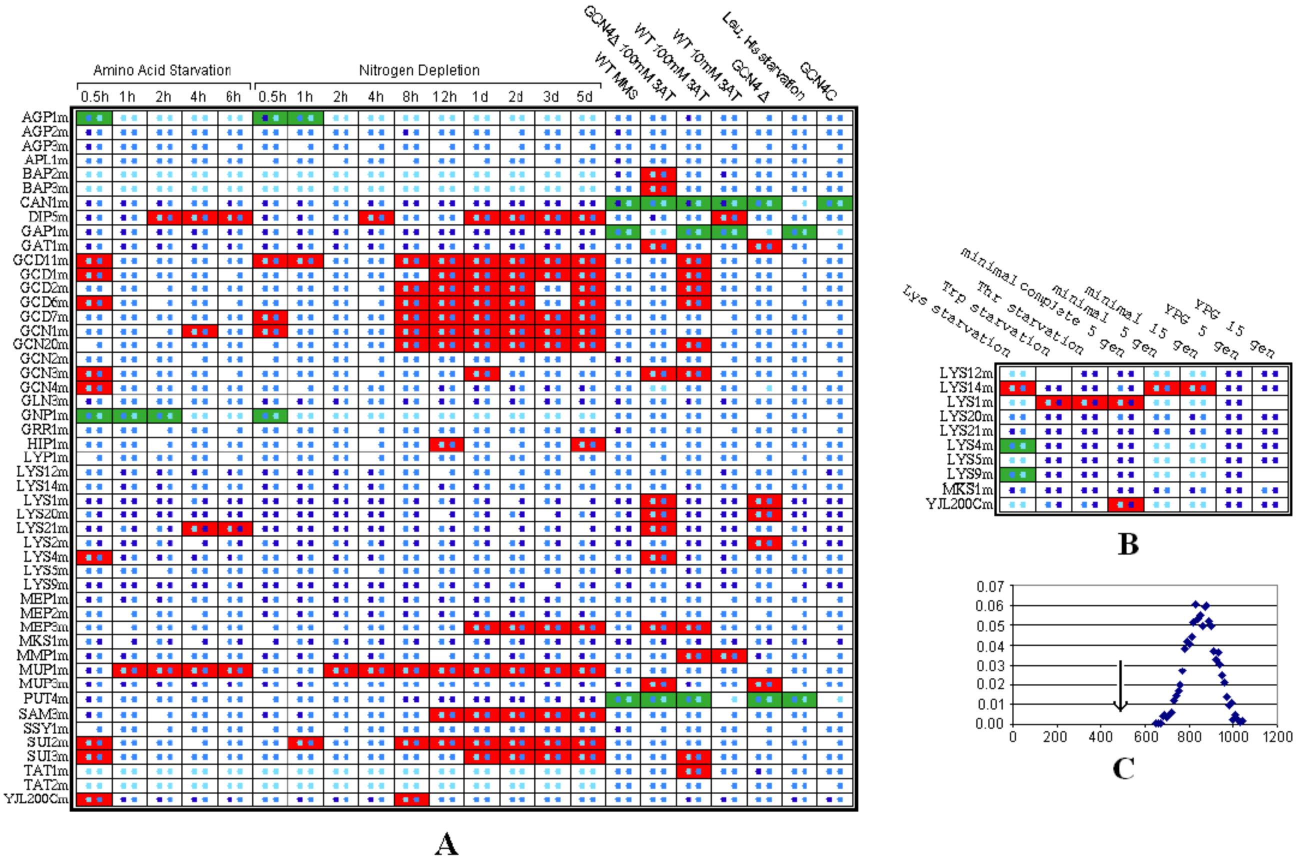
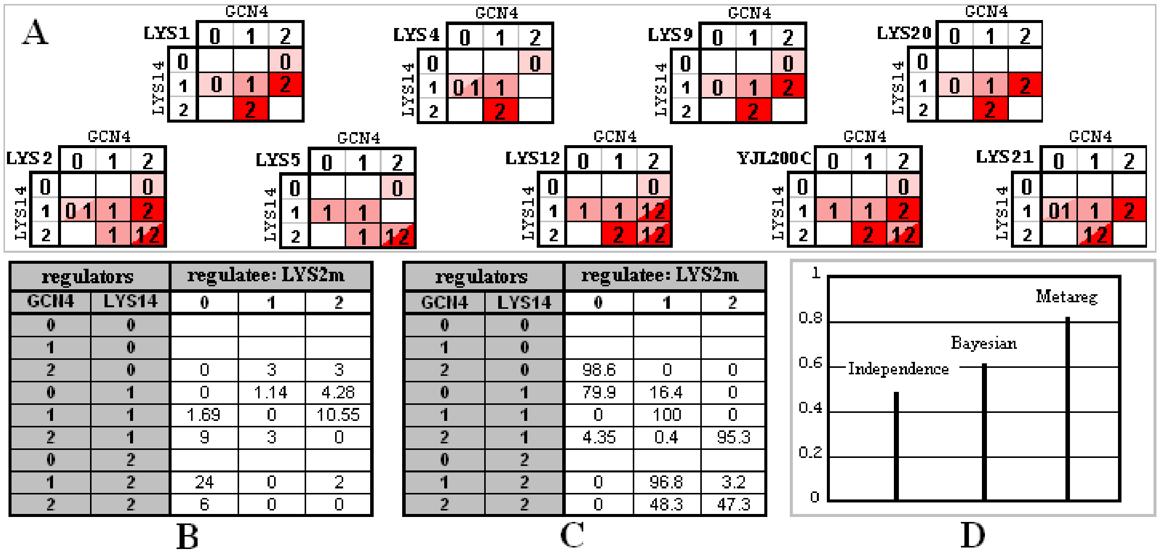


Figure 3: **Model Discrepancy (A) Discrepancy matrix for the expression data.** Columns correspond to conditions and rows correspond to mRNA variables. Each cell contains two small squares: observed (left) and inferred (right) states of the row variable in the column condition. State colors: Cyan (light gray):0, light blue (gray):1, dark blue (black):2. The background color of the cells emphasizes critical disagreement, where the inferred state is zero and the observed state is not (green or light gray), or vice versa (red or gray). **(B) Discrepancy matrix for the phenotype data.** Each cell represents a condition, which is a combination of certain environmental nutrients and one gene deletion. Columns correspond to the nutritional environment (i.e., the medium), and rows correspond to the knocked-out variable. Each cell contain two small squares: observed (left) and inferred (right) state of the internal lysine metabolite (the ILys variable) in this condition. Colors are as in (A). **(C). Distribution of model discrepancy scores for randomly shuffled data sets.** X axis: total model discrepancy. We generated the distribution by computing model discrepancy for 50 random data sets. The discrepancy of the real data set is 494 (arrow), much lower than the minimal discrepancy measured in the shuffled sets.



LYS1	0	1	2
0			0
1	0	1	2
2		2	

LYS4	0	1	2
0			0
1	0 1	1	
2		2	

LYS9	0	1	2
0			0
1	0	1	2
2		2	

LYS20	0	1	2
0			
1	0	1	2
2		2	

LYS2	0	1	2
0			0
1	0 1	1	2
2		1	12

LYS5	0	1	2
0			0
1	1	1	
2		1	12

LYS12	0	1	2
0			0
1	1	1	12
2		2	12

YJL200C	0	1	2
0			0
1	1	1	2
2		2	12

LYS21	0	1	2
0			
1	01	1	2
2		12	

B

regulators		regulatee: LYS2m		
GCN4	LYS14	0	1	2
0	0			
1	0			
2	0	0	3	3
0	1	0	1.14	4.28
1	1	1.69	0	10.55
2	1	9	3	0
0	2			
1	2	24	0	2
2	2	6	0	0

C

regulators		regulatee: LYS2m		
GCN4	LYS14	0	1	2
0	0			
1	0			
2	0	98.6	0	0
0	1	79.9	16.4	0
1	1	0	100	0
2	1	4.35	0.4	95.3
0	2			
1	2	0	96.8	3.2
2	2	0	48.3	47.3

Figure 4: **Learning regulation functions.** (A) The optimal transcription regulation function of each lysine biosynthesis pathway enzyme as a function of the states of the regulators GCN4ap and LYS14ap. Each cell presents the state of a regulatee given the states of its regulators GCN4ap (column) and LYS14ap (row). Cell colors indicate the regulatee states. Red (dark gray): state= 2. Dark pink (gray): 1. Light pink (light gray): 0. We show only entries with over 90% confidence. For combinations of regulators states that have lower confidence or were never present in the inferred modes, we leave the corresponding entries of the optimal regulation function undefined. (B). The discrepancy change for each function modification relative to the optimum regulation function of LYS2. Rows: combination of regulators (GCN4ap,LYS14ap) states. Columns: regulatee (LYS2) states. The cell $((x, y), z)$ represents the difference in discrepancy of a regulation function in which $f(x, y) = z$ and all other values are as in the optimal LYS2m function. All values are relative to the discrepancy of the optimal regulation function as shown in A. (C) Confidences for the LYS2 function. Rows and columns are as in (B), values are the percent of times in which the value was learned out of 1000 bootstrap experiments. (D) The accuracy of the independence, Bayesian and MetaReg methods on the lysine biosynthesis pathway. The accuracy is computed by cross validation on all expression conditions and the lysine biosynthesis pathway enzymes.

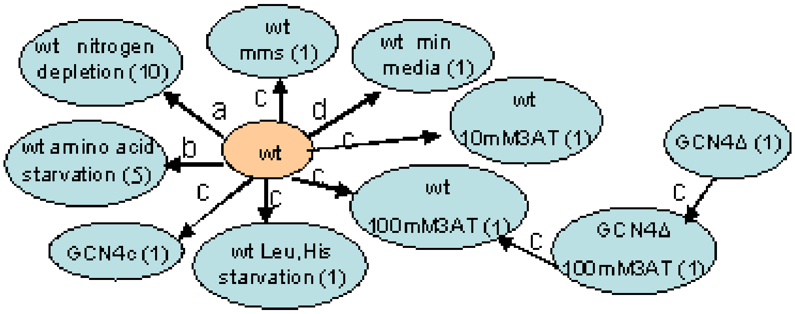
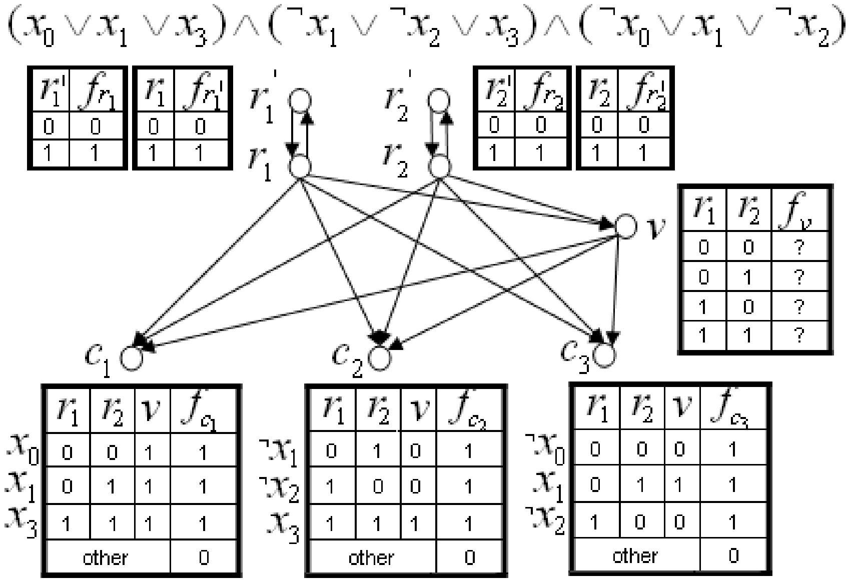


Figure 5: **Experiments graph.** The conditions and experiments used in our analysis. Ovals represent groups of conditions vertices, shown together to simplify graphical representation (the number of conditions represented by each oval is shown in parentheses). Arcs represent differential gene expression experiments and are annotated by their respective publication code: a,b:[10]. c:[5]. d:[19]. wt: wild type in standard conditions and YPD medium.



$$(x_0 \vee x_1 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_0 \vee x_1 \vee \neg x_2)$$

r_1'	f_{r_1}
0	0
1	1

r_1	$f_{r_1'}$
0	0
1	1

r_2'	f_{r_2}
0	0
1	1

r_2	$f_{r_2'}$
0	0
1	1

r_1	r_2	f_v
0	0	?
0	1	?
1	0	?
1	1	?

	r_1	r_2	v	f_{c_1}
x_0	0	0	1	1
x_1	0	1	1	1
x_3	1	1	1	1
	other			0

	r_1	r_2	v	f_{c_2}
$\neg x_1$	0	1	0	1
$\neg x_2$	1	0	0	1
x_3	1	1	1	1
	other			0

	r_1	r_2	v	f_{c_3}
$\neg x_0$	0	0	0	1
x_1	0	1	1	1
$\neg x_2$	1	0	0	1
	other			0

Figure 6: **Construction of a reduction model.** Top: two variable encoding gadgets. Bottom: Three clause variables. Middle: The truth assignment variable whose function is optimized.