JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 13, Number 2, 2006

Pp. 165–181

A Probabilistic Methodology for Integrating Knowledge and Experiments on Biological Networks

IRIT GAT-VIKS, AMOS TANAY, DANIELA RAIJMAN, and RON SHAMIR

ABSTRACT

Biological systems are traditionally studied by focusing on a specific subsystem, building an intuitive model for it, and refining the model using results from carefully designed experiments. Modern experimental techniques provide massive data on the global behavior of biological systems, and systematically using these large datasets for refining existing knowledge is a major challenge. Here we introduce an extended computational framework that combines formalization of existing qualitative models, probabilistic modeling, and integration of high-throughput experimental data. Using our methods, it is possible to interpret genomewide measurements in the context of prior knowledge on the system, to assign statistical meaning to the accuracy of such knowledge, and to learn refined models with improved fit to the experiments. Our model is represented as a probabilistic factor graph, and the framework accommodates partial measurements of diverse biological elements. We study the performance of several probabilistic inference algorithms and show that hidden model variables can be reliably inferred even in the presence of feedback loops and complex logic. We show how to refine prior knowledge on combinatorial regulatory relations using hypothesis testing and derive p-values for learned model features. We test our methodology and algorithms on a simulated model and on two real yeast models. In particular, we use our method to explore uncharacterized relations among regulators in the yeast response to hyper-osmotic shock and in the yeast lysine biosynthesis system. Our integrative approach to the analysis of biological regulation is demonstrated to synergistically combine qualitative and quantitative evidence into concrete biological predictions.

Key words: biological systems, probabilistic modeling, high throughput data.

1. INTRODUCTION

THE INTEGRATION OF BIOLOGICAL KNOWLEDGE, high throughput data, and computer algorithms into a coherent methodology that generates reliable and testable predictions is one of the major challenges in today's biology. The study of biological systems is carried out by characterizing mechanisms of biological regulation at all levels, using a wide variety of experimental techniques. Biologists are continuously refining models for the systems under study, but rarely formalize them mathematically. High-throughput techniques have revolutionized the way by which biological systems are explored by generating massive

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

amounts of information on the genomewide behavior of the system. Genomewide datasets are subject to extensive computational analysis, but their integration into existing biological models is currently done almost exclusively manually. To rigorously integrate biological knowledge and high-throughput experiments, one must develop computational methodologies that accommodate information from a broad variety of sources and forms and handle highly complex systems and extensive datasets.

Recent studies on computational models for biological networks have attempted *de novo* reconstruction of a network on genes (e.g., Friedman *et al.* [2000]), used prior knowledge on network topology (e.g., Hartemink *et al.* [2002] and Imoto *et al.* [2004]), or combined transcription factor location and sequence data to learn a clustered model for the genomewide behavior of the system (Bar-Joseph *et al.*, 2003; Segal *et al.*, 2003; Beer and Tavazoie, 2004). Other studies built detailed models manually, utilizing existing biological knowledge (Chen *et al.*, 2000; Covert *et al.*, 2004) but lacked computational methods for model reassessment in light of additional evidence.

In this study, we describe a new mathematical framework for representing biological knowledge and integrating it with experimental data. Our methodology allows biologists to formalize their knowledge on a system as a coherent model and then to use that model as the basis for computational analysis that predicts the system's behavior in various conditions. Most importantly, our framework allows the learning of a refined model with improved fit to the experimental data.

In previous works (Tanay and Shamir, 2001; Gat-Viks *et al.*, 2004), we have introduced the notions of model refinement and expansion and studied it when applied to discrete deterministic models. Here we study these problems in the more general settings of probabilistic models. The probabilistic approach allows us to model uncertainty in prior biological knowledge and to distinguish between regulatory relations that are known at a high level of certainty and those that are more speculative. The probabilistic model also allows us to mix noisy continuous measurements with discrete regulatory logic. Our model expresses diverse biological entities (e.g., mRNAs, proteins, metabolites) and biological relations (e.g., transcription and translation regulation, posttranslational modifications). We formalize our model as a probabilistic factor graph (Kschischang *et al.*, 2001), accommodating undelayed feedback loops which are essential in many biological systems.

Having established our methodology for probabilistic modeling, we develop algorithms for inferring the system's state given partial data. For example, we can infer the activity of proteins given gene expression data. We use inference algorithms as the basis for learning refined regulatory functions. We develop a formulation of the learning problem in our network model, which is based on deterministic hypothesis testing. Our approach to the learning of regulatory models uses regulatory features with clear biological meaning and allows the derivation of p-values for learned model features.

We tested the performance of our algorithms on simulated models and on two complex pathways in *S. cerevisiae*: the regulation of lysine biosynthesis and the response to osmotic stress. In both cases, our models successfully integrate prior knowledge and high throughput data and demonstrate improved performance compared to extant methods. In particular, our results suggest a novel model for regulation of genes coding for components of the HOG signaling pathway and robustly learn logical relations among central transcription factors downstream of the Hog1 kinase. Our results show that integration of prior biological knowledge with high-throughput data is a key step toward making computational network analysis a practical part of the toolbox of the molecular biologist.

The paper is organized as follows: In Section 2 we introduce our mathematical formulation for prior biological knowledge and experimental data. In Section 3, we show how to infer the state of hidden variables. Sections 4 and 5 present our learning methodologies: Section 4 focuses on our discretization scheme and how we propose to learn it. Section 5 presents our mathematical formulation for learning regulation functions and describes a way to assign statistical meaning to the learned functions. Section 6 presents our results on the lysine and HOG pathways. In Section 7, we discuss the advantages and limitations of our approach and outline future research directions.

A preliminary version of this study appeared in the proceedings of RECOMB 2005 (Gat-Viks *et al.*, 2005).

2. MODELING PRIOR KNOWLEDGE AND EXPERIMENTAL OBSERVATIONS

In this section, we present our probabilistic model for a biological regulatory network. We start by defining model variables and formulating prior knowledge on the relations among them. We then incorporate

experimental evidence into the model and show how to combine prior knowledge and experiments into one integrated probability distribution.

2.1. Variables, topology and logic

The biological entities in the system under study are formulated as variables representing, e.g., mRNAs, proteins, metabolites, and various stimulators. We assume that at a given condition, each of the entities attain a logical state, represented by an integer value of limited cardinality. We wish to study regulatory relations (or regulation functions) among variables. Such relations, for example, determine the level of an mRNA variable as a function of the levels of a set of transcription factor protein variables, or the level of a metabolite variable given the levels of other metabolites and of structural enzymes.

In most studied biological systems, substantial prior knowledge on regulatory relations has accumulated. Such knowledge includes direct regulatory interactions, qualitative functional roles (activator/repressor), combinatorial switches, feedback loops, and more. Typically, that information is incomplete and of variable certainty. In order to optimally exploit it, we must model both the relations and their level of certainty. We do this by introducing a distribution on the regulation functions for each variable. This distribution may assign high probability to a single regulation function if our prior knowledge is very strong. At the other extreme, lack of information is modeled by uniform distribution over all possible regulation functions.

We formalize these notions as follows (see Fig. 1A). Let $X = \{X_1, \ldots, X_n\}$ be a collection of biological variables. Let $S = \{0, 1, \ldots, k-1\}$ be the set of logical *states* that each variable may attain. A *model state* s is an assignment of states to all the variables in X. Each variable X_i is regulated by a set of its *regulator* (or *parent*) variables $Pa_i = \{Pa_{i,1}, \ldots, Pa_{i,d_i}\} \subseteq X$. When addressing a particular regulation relation, the regulated variable is also called the *regulatee*. Lower case letters will indicate state assignments of the corresponding upper case variables. For example, given a model state s, x_i^s is the state of X_i, pa_i^s is the assignment of the set Pa_i. The *regulatory dependency graph* is a digraph $G_R = (X, A)$ representing direct dependencies, i.e., $(X_u, X_v) \in A$ iff $X_u \in Pa_v$ (G_R is sometimes called the *wiring diagram* on the *topology* of the model). The graph can contain cycles. The *regulation function prior* for a variable X_i is formulated as our belief that the variable attains a certain state given an assignment to its parents Pa_i. It is represented by the conditional probabilities θ^i:

$$\theta^i(X_i, Pa_i) = Pr(X_i | Pa_i) \tag{1}$$

2.2. From measurements to logical states

In practice, biological experiments provide noisy observations on a subset of the variables in the system. The observations are continuous, and we do not know in advance how to translate them into logical states. We thus introduce a set of real-valued *sensor variables* $Y_1, \ldots, Y_n$ and *discretizer distributions* $\psi^i(X_i, Y_i)$ that specify the joint distribution of a discrete logical state of X_i and the continuous observation on Y_i. In this work, we shall use mixtures of Gaussian (Fig. 1B) to model ψ^i, but other formulations are also possible. Note that we chose to formulate the relations between the logical and sensor variables as joint rather than as conditional probabilities $P(Y_i | X_i)$.

In addition to providing partial observations on model variables, experiments are performed in a specific environment and may possibly perturb some of the regulation functions in the system (for example, by knocking out or overexpressing some genes). We model these by fixing the values of logical variables that correspond to the environment and by changing the regulation function priors (the θ factors) to reflect the perturbations.

2.3. The factor graph network model

Our model is defined by a joint distribution over a set of logical (X) and sensor (Y) variables. The distribution is constructed as the product of the *factors* θ^i, ψ^i, such that

$$Pr_M(X, Y) = \frac{1}{Z} \prod_i \theta^i(X_i, Pa_i) \psi^i(X_i, Y_i) \tag{2}$$

where Z is a normalization constant.

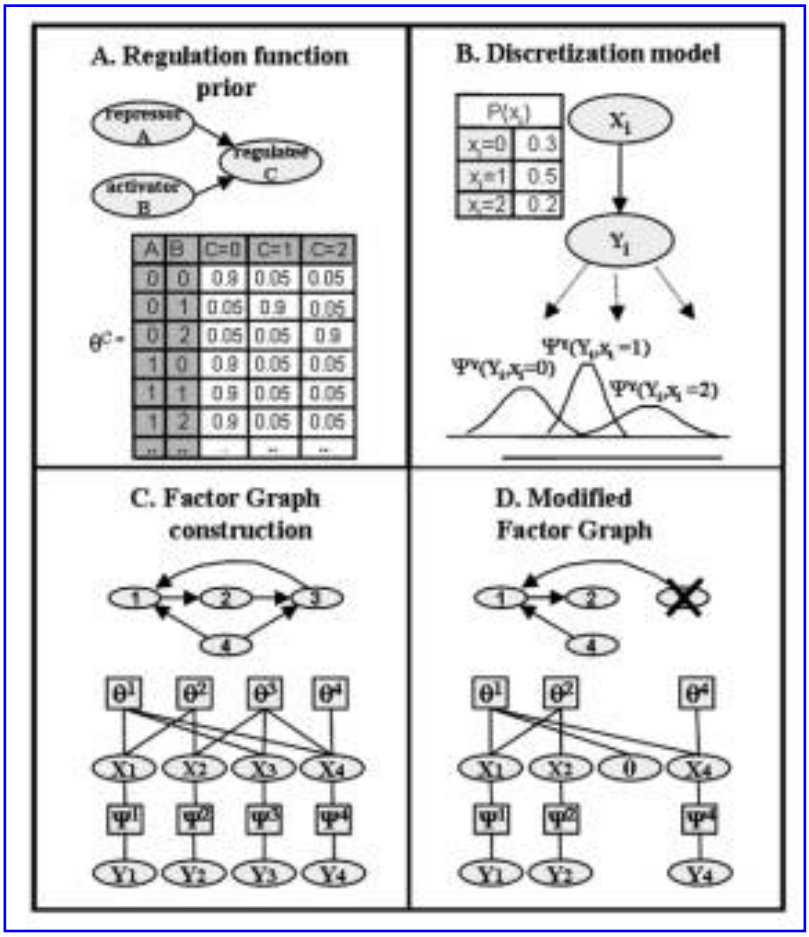

FIG. 1. An overview of the factor graph network model. (**A**) Knowledge on the logical regulation functions is formalized as conditional probabilities. (**B**) Continuous measurements and logical states are linked by joint discretizer distributions. (**C**) A possibly cyclic network structure G_R (top) is transformed into a factor graph (bottom), using the regulation function priors and the discretizers' distributions. (**D**) Given an experiment in which gene X_3 is knocked out, the model is modified accordingly by fixing the state of X_3 to zero and eliminating the corresponding factors Θ^3, Ψ^3.

We can represent our joint distribution using a probabilistic factor graph (Kschischang *et al.*, 2001) which explicitly expresses the structure of the joint distribution?s factorization. Factor graphs are widely used probabilistic graphical models that were originally applied to coding/decoding problems (for a different application of factor graphs in computational biology, see Yeang *et al.* [2004]). A factor graph is a bipartite graph associating variable nodes (in one side of the graph) with factor nodes (in the other side of the graph). We add an edge between a variable x and a factor f_j if the scope of f_j contains x. In our case (Fig. 1C), the factor graph has a variable node for each logical and sensor variable (X, Y) and a factor node for each function θ^i, ψ^i. A modified factor graph representation, matching a perturbation experiment, is shown in Fig. 1D. We call this formulation a *factor graph network (FGN) model*. Note that our formulation is undirected although part of our model (the sensor variables) represent conditional probabilities. Although it is in principle possible to use hybrid models (e.g., chain graphs Buntine [1995]) and maintain the directionality information in the model, for our purpose here, the undirected formulation suffices.

When the dependency graph G_R is acyclic, our FGN model is equivalent to a Bayesian network on the variables X_i and Y_i, constructed using the edges of G_R and additional edges from each X_i to the corresponding Y_i. This can be easily seen from (2) by noting that in the acyclic case $Z = 1$ (the proof is as in Bayesian networks theory, e.g., Pearl [1988]). When the model contains loops, the situation gets more

complicated. For example, we note that according to the FGN model, $Pr_M(X_i|Pa_i)$ does not necessarily equal the original beliefs $\theta(X_i, Pa_i)$.

We note that the semantics of the prior θ^i distributions is different than that used in previous works (e.g., Friedman *et al.* [2000]), where they served as conditional probabilities on the values of the variables in a probabilistic setting. Instead, we assume that the true model *deterministically* determines X_i given its parents, but we are not sure which deterministic rule applies, and therefore what value X_i will attain. Regulation functions approximate an underlying biochemical reaction whose exact parameters are usually not known. The regulatory process is stochastic at the single cell level, but the parameters of the reaction equations governing it are deterministic. When we observe a large ensemble of cells in a high-throughput experiment, we average millions of stochastic processes and in theory should obtain an almost deterministic outcome or a superposition of several deterministic modes. Such deterministic outcome is obscured by significant experimental noise, so a practical modeling approach may assume uncertainties on deterministic logic and noisy observations. In the future, given measurements at the single cell level, the θ distributions may be applicable to describe the inherent stochasticity of some biological switches.

3. INFERENCE

In this section, we discuss the inference problem in the FGN model. Each experiment provides partial information on the value of model variables. Typically, a subset of the sensor real-valued variables are observed in each experiment (for example, mRNA variables are determined in a gene expression experiment), and the model is modified according to some perturbations at the appropriate condition (compare Fig. 1D). The inference problem seeks the computation of the distribution of hidden (unmeasured) variables given the experimental data and the model.

There are two types of inference problems we shall deal with. The first problem (*marginal inference*) is to compute *posterior distribution*s for a single hidden variable. For example, given a gene expression profile D (specifying observations on all mRNA sensor variables), we may wish to compute the marginal $P(X_i|D)$ of a protein variable or a certain metabolite. The second problem is to compute the likelihood $P(D)$ of the observed data D. Inference in graphical models is an NP-hard problem (Cooper, 1990) that was extensively studied. We explored the effects of our model's specific characteristics on the performance of three inference algorithms. Specifically, we implemented a Gibbs sampler, the loopy belief propagation algorithm, and a structure-based instantiation inference algorithm.

The *Gibbs sampler* is a naive MCMC algorithm (MacKay, 1998) that performs a random walk over the space of model states, based on sampling from local distributions. To perform Gibbs sampling, we convert the FGN model to the equivalent Bayesian network as described by Yedidia *et al.* (2004). In our model, sampling is done only for the logical variables (unobserved sensors do not affect marginal posteriors of the logical variables, since they are integrated to 1).

The loopy belief propagation (LBP) algorithm belongs to a popular class of algorithms (Yedidia *et al.*, 2004) which approximate the posterior distribution assuming certain decomposition over independent variables or clusters of variables. Algorithms from this class include LBP, mean field, and their generalizations. The LBP algorithm for the FGN model (implemented as described by Yedidia *et al.* [2004]) is a message-passing procedure that is guaranteed to reach an exact solution for acyclic models and was reported to perform well in some cases of cyclic models.

We also developed an instantiation-based inference algorithm that exploits the known dependency structure of the model and builds on ideas from the deterministic network model (Gat-Viks *et al.*, 2004). Briefly, recall that a deterministic (possibly loopy) network model is defined by a set of deterministic regulation functions (one for each variable) and that such a network may attain a limited number of steady states (or *modes*) in which the value of each variable is correctly determined by its regulation function and the values of its regulators. Also recall that in an acyclic model (or in a loopy model in which the values of the variables in a feedback set are fixed), the mode is uniquely determined (if one exists). We can therefore search for modes in a deterministic network by analyzing the underlying topology, identifying a feedback set, and enumerating over all value assignments for it. The modes instantiation (MI) algorithm first builds a deterministic network model by taking, for each variable, the maximum likelihood regulation function (using the prior θ^i and breaking ties arbitrarily). It then identifies a feedback set in G_R and computes

the appropriate set of modes. The algorithm next computes the likelihoods of each mode in the original probabilistic model, possibly optimizing it using a greedy algorithm. The results of this procedure are a set of model states with locally optimal likelihoods. For models that are close to being deterministic in many of the variables, such set of modes may represent a significant chunk of the total likelihood of the model given the data. The algorithm therefore approximates the posterior distribution as a mixture of modes, weighted by their likelihoods. Since the number of modes may be small in practice, the algorithm adds to the set of locally optimal states an additional small set of states derived using the Gibbs sampler, generating a more smooth approximation for the posterior. The MI algorithm constructs a tractable estimation of the joint posterior which can be used both for computation of marginal posteriors and for estimation of the full probability $P(D)$. There are no guarantees for the quality of this approximation, but our empirical studies suggest that the algorithm works well in practice, probably due to the nature of models we use (strong priors on many of the regulation functions).

We tested the three inference algorithms on a simulated model (see Fig. 2). We constructed simulated FGN models by starting from a deterministic model and randomizing it. We use a *prior strength* parameter α to construct θ functions that assign probability α for the anticipated deterministic function outcome and $\frac{1-\alpha}{k-1}$ to other values. For a detailed description of the simulation, see our website *www.cs.tau.ac.il/~rshamir/fgn/*.

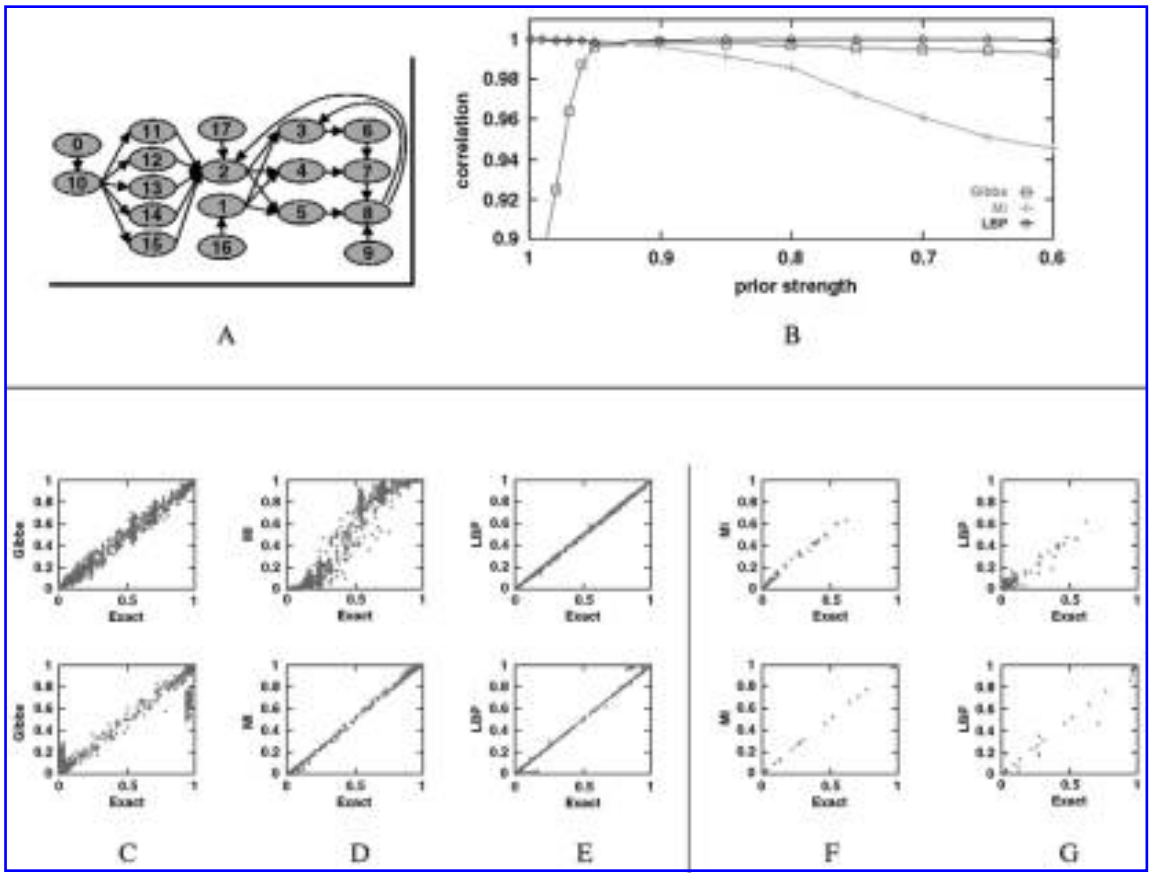

FIG. 2. Performance of different inference algorithms on a simulated model. Performance is measured by the correlation of the inferred and the exact posterior distribution. (**A**) The dependency graph G_R of the simulated model. (**B**) Effect of prior strength on inference accuracy. Y axis: the correlation of the inferred and exact marginal posteriors. X axis: prior strength (α). For strong priors, LBP and MI give a good approximation for the posterior, while the accuracy of the Gibbs sampler is low. As priors get weaker, the performance of MI deteriorates, indicating that the mixture of deterministic states is a poor approximation for the posterior in these cases. (**C**, **D**, **E**) Detailed correlation of inferred and exact marginal posteriors for $\alpha = 0.7$ (top) and $\alpha = 0.97$ (bottom). (**F**, **G**) Detailed correlation of inferred and exact joint posteriors for $\alpha = 0.7$ (top) and $\alpha = 0.97$ (bottom). We see that MI outperforms LBP when comparing the joint posteriors.

We explored the behavior of the different algorithms as a function of the prior strength α using the correct posterior as the reference. Models with α near 1 represent very good knowledge on the system under study. Models with α near $\frac{1}{k}$ represent complete lack of knowledge. We first tested the accuracy of inferring marginal posteriors. Figures 2B,C,D,E indicate that for estimating marginal posteriors, LBP outperforms the other two algorithms (and also the mean field algorithm and a simple clustered variational algorithm [Jaakkola, 2001], data not shown). When the prior is strong, MI provides comparable accuracy. We also wished to test the quality of inferring joint posterior distributions. Joint posteriors cannot be computed directly from LBP, and thus are estimated by multiplying marginal posteriors (assuming independence among the variables). For the MI algorithm, we applied the approximation of the posterior distribution using a mixture of locally optimal states. The results (Figs. 2F,G) confirm that for models with loops and strong prior knowledge, the approximation using the MI algorithm performs better, exemplifying the limitations of the posterior independence assumptions. Overall, we prefer using LBP to infer marginal posteriors and MI to approximate the joint posterior distribution.

4. LEARNING DISCRETIZERS

Adequate transformation of continuous measurements into logical states (i.e., discretization) is essential for the combined analysis of experimental data and a model representing extant biological knowledge. There are several alternative approaches to discretization. In most previous works on discrete models (e.g., Friedman *et al.* [2000] and Gat-Viks *et al.* [2004]), discretization was done as a preprocess, using some heuristic rule to map real-valued measurements into discrete states. In this approach, the rule must be determined and tuned rather arbitrarily, and typically all variables are discretized using the same rule. Here we propose a different approach to discretization. As in the FGN model the discretization is an integral part of the model, the dependencies between the discretization schemes and regulation function priors are fully accounted for. It is thus possible to (a) define different discretization scheme for different variables and (b) apply standard learning algorithms to optimize the discretization functions used. Given a logical function prior and experimental evidence D, we learn the discretization functions ψ^i using an EM algorithm. We initialize all ψ^i using any heuristic discretization scheme. In each EM iteration, we infer the posterior distributions for each of the variables X_i in each of the conditions and then reestimate the ψ^i mixtures using these posteriors, by computing the Gaussians sufficient statistics $E(Y_i|X_i = j, D)$, $V(Y_i|X_i = j, D)$. The new ψ^i distributions are used in the next iteration, and the algorithm continues until convergence.

The FGN model thus provides a very flexible discretization scheme. In practice, this flexibility may lead to overfitting and may decrease learnability. One can control such undesired effects by using the same or few discretization schemes on all variables. As we shall see below, on real biological data, variable specific discretization outperforms global discretization using a single scheme and is clearly more accurate than the standard preprocessing approach.

5. LEARNING REGULATION FUNCTIONS

Given an FGN model and experimental evidence, we wish to determine the optimal regulation function for each variable and provide statistical quantification of its robustness. We assume the parameters of the logical factors in the FGN model represent our prior beliefs on the logical relations between variables and attempt to learn by confirming beliefs, deciding whether a certain regulator assignment gives rise to a certain deterministic regulatee assignment.

5.1. Formulating the learning problem

We focus on the regulation of some variable X_i and attempt to learn a single deterministic feature in the model: the value of X_i given a fixed parents value assignment pa_i^s. Define h_j as the FGN model derived from M by setting $\theta(j, pa_i^s) = 1$ and $\theta(j', pa_i^s) = 0$ for $j' \neq j$ and keeping all other model parameters at their original values. We define the learning problem in our model as selecting the maximum likelihood h_j.

To that end, we shall have to compute the likelihood of the data given each of the h_js, a difficult problem when we have hidden variables even on an acyclic model.

The likelihood of the data, given a model, is approximated by the inference algorithms described above. Recall that we can approximate the full probability using a small number of high-probability modes (using, e.g., the MI algorithm). While this may be a crude approximation, our empirical analysis shows that it is still adequate (see below). Importantly, the likelihood of each h_j takes into account all our prior knowledge on regulation functions and experimental observations.

We note that our approach can be viewed as standard Bayesian learning, using a prior that assumes that the only possible regulation functions are the deterministic ones. We have chosen to represent the learning process as selection of the maximum likelihood discrete hypothesis for two reasons: First, this sharp prior helps us define the semantic of the features that we learn (e.g., activation/repression). Second, it allows standard statistical tools (e.g., likelihood ratio testing) to be applied, so that p-values of learned regulation rules can be derived. In the future, when single cell measurements are available, and when models that explicitly express the stochasticity of regulatory switches are developed, other types of priors may be more appropriate.

5.2. Statistical evaluation

To assign statistical meaning to the learning procedure, we use two methods: bootstrap and likelihood ratio testing. In the bootstrap method, we repeatedly select random subsets of conditions from the original data D and for each one perform the learning procedure. We count the number of times each h_j was selected as the maximum likelihood model and define the feature *robustness* as the fraction of times it was selected. Bootstrap is in widespread use in cases where sampling from the background distribution is impossible or very difficult. In our case, approximated sampling from $Pr(D|h_j)$ is possible given our representation of the posterior landscape as a mixture of modes. We can thus try to directly perform a likelihood ratio test and derive p-values for the learned features.

In a likelihood ratio test, we test the null hypothesis H_0 against the alternative hypothesis H_1. The test statistic is the ratio $\lambda = \frac{max_{h_i \in H_0 \cup H_1} Pr(D|h_i)}{max_{h_i \in H_0} Pr(D|h_i)}$. We decide to reject the null hypothesis (and accept H_1) if an observed ratio λ' is too high and assign this decision a significance level by computing a p-value $pr(\lambda \geq \lambda'|H_0)$. Therefore, the distribution of λ given H_0 must be estimated.

In our case, we fix j and define $H_1 : h_j$, $H_0 : \cup_{k \neq j} h_k$. To estimate the distribution $p(\lambda|H_0)$, we generate samples from the distribution $Pr(D|H_0)$, compute the corresponding λs, and reconstruct the λ distribution. The main problem is therefore the sampling of datasets D. When sampling, we take into account the

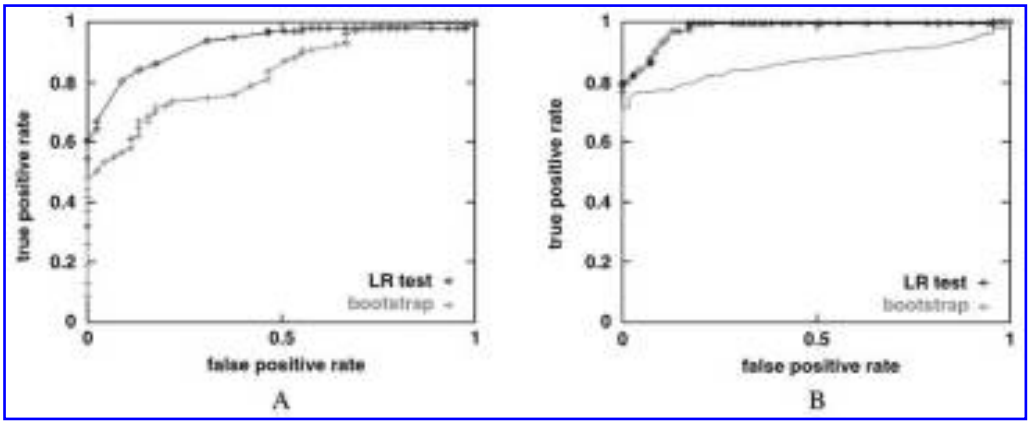

FIG. 3. Accuracy of learning regulation functions. Each figure is a ROC curve (X axis: false positives rate, Y axis: true positives rate) for learning the functions in a simulated model using bootstrap and likelihood ratio test for determining the significance of learned features. LR-test (bootstrap) curves were obtained by varying the p-value (robustness) between 0 and 1 and for each value, averaging over the true positive rates for all variables in the model. Results are shown for learning from 15 (**A**) and 80 (**B**) conditions and represent the average across all model variables. The accuracy of the likelihood ratio test method is consistently higher.

model H_0 that was modified according to the perturbations in each of the experiments (Fig. 1D). We do this as follows: for each of the conditions in the original dataset, we form the modified model according to the experiment. We then apply the MI algorithm to the modified model, without any observation on Y variables, and compute the set of posterior modes for the X variables. These modes represent logical model states that are probable given the experimental conditions. We then generate a sample by (a) selecting a mode from the set of posterior modes, weighted by their likelihood, and (b) generating observations on Y variables using the model discretizer distributions ψ. Our procedure therefore generates a random sample of conditions for a true H_0 model in which the corresponding experimental perturbations were performed.

We analyzed the performance of the bootstrap and likelihood ratio test methods by learning features in our simulated model (see Figure 2A and our website *www.cs.tau.ac.il/∼rshamir/fgn/* for details). Figure 3 shows ROC curves for learning in the simulated model using 15 and 80 conditions. We see consistently better accuracy when using the likelihood ratio tests, probably due to better resolution of features that are nearly ambiguous given the data. While bootstrap has the advantage of not assuming an approximation to the full probability of the data, the likelihood ratio test is more accurate when the posterior can be reasonably approximated.

6. RESULTS ON BIOLOGICAL DATA

In order to test the applicability of our methods to real biological systems, we constructed models of two important yeast pathways, the Hog1 MAPK pathway, which mediates the yeast response to hyperosmotic stress, and the lysine intake and biosynthesis pathway. For each of the models, we performed an extensive literature survey in order to construct the initial model of the system (for the lysine system, our previously developed deterministic model [Gat-Viks *et al.* [2004] was the main source). We collected published experimental data on each of the models.

The HOG model is an acyclic model with 50 variables (outlined in Fig. 4). The lysine biosynthesis model contains 140 variables, 28 of which are involved directly in feedback loops or in the biosynthesis regulation. Figure 5 illustrates only this part of the model. The full topology of the model appears in

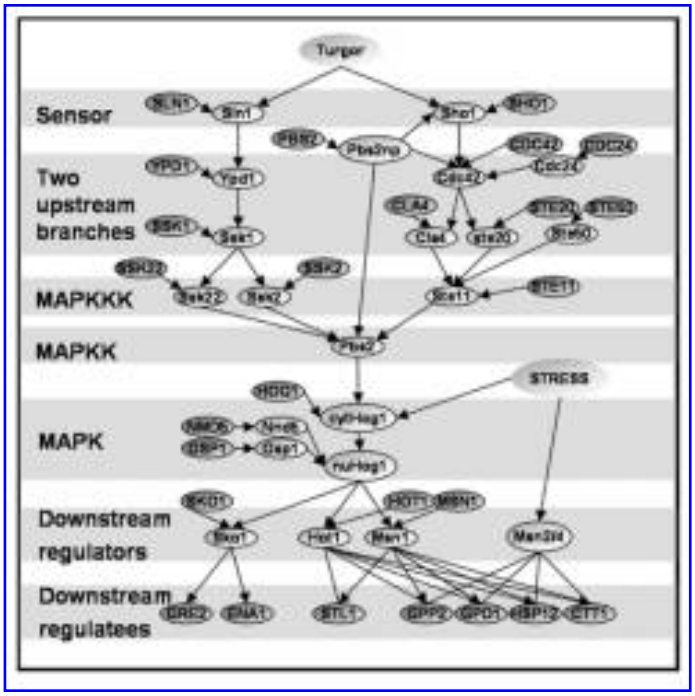

FIG. 4. Topology of the HOG model. The mRNA variable names are capitalized; protein variable names appear with initial capital letters. Turgor and stress are stimulator variables; cytHog1: cytoplasmic Hog1; nuHog1: nuclear hog1.

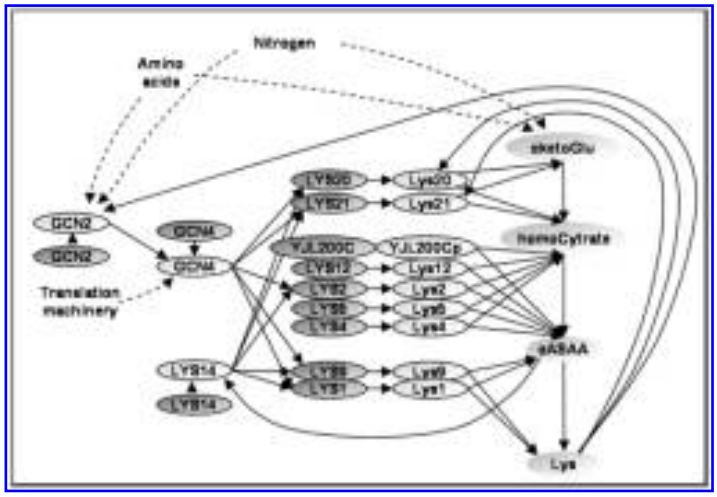

FIG. 5. Partial topology of the lysine biosynthesis model. The mRNA variable names are capitalized; protein variable names appear with initial capital letters. Metabolites are shown as unoutlined ovals. Amino acids and nitrogen transport modeling (110 variables) and translation machinery (10 variables) are not shown.

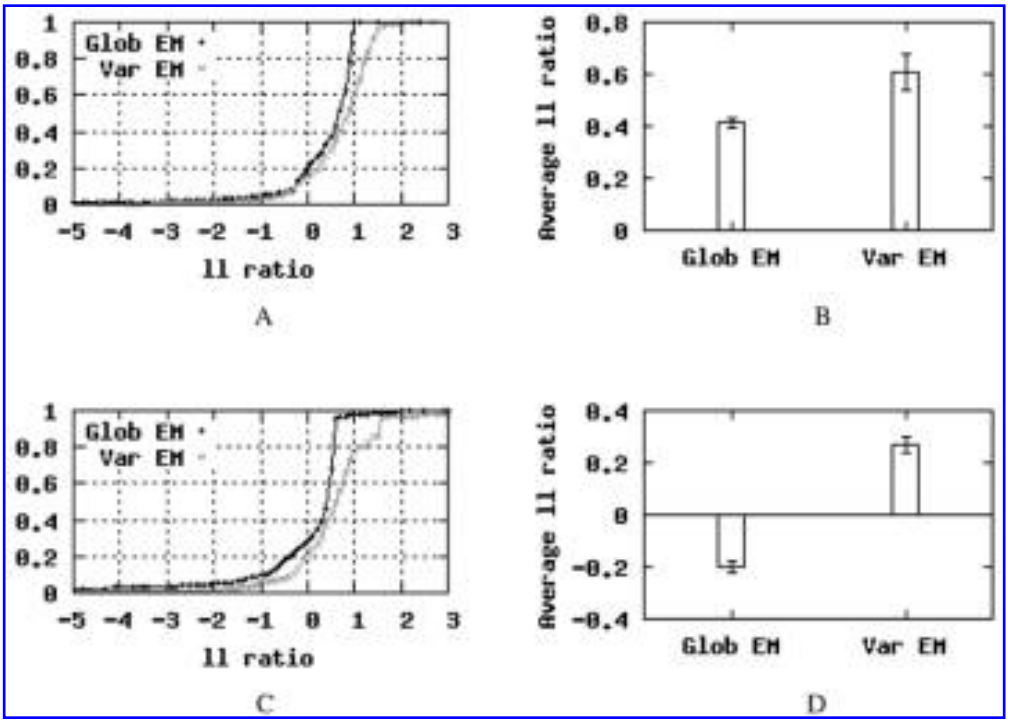

FIG. 6. Learning discretization distributions. Cross validation results for alternative methods for estimating the discretization functions ψ^i in the HOG (**A**, **B**) and lysine model (**C**, **D**). Glob EM: optimized single common discretization function. Var EM: optimized variable specific discretization. (**A**, **C**) Cumulative distribution of log likelihood (ll) ratios comparing each of the two discretization methods to the global predefined discretization scheme. (**B**, **D**) Average ll ratios for the two methods. Bars indicate the predicted standard deviation of the averages.

Gat-Viks *et al.* (2004). The complete description of the models, including the regulation functions, can be found at our website.

We collected published experimental data on each of the models. The data consisted of 129 conditions for the HOG model (O'Rourke and Herskowitz, 2004), and 23 conditions (cf. Gat-Viks *et al.* [2004]) for the lysine model. Differential measurements from cDNA microarrays were transformed into absolute values as described by Gat-Viks *et al.* (2004). In both models, we used three-valued logical variables, with values 0,1,2 corresponding to low, intermediate, and high levels. We used prior strength $\alpha = 0.9$ for all regulation functions in both models.

6.1. Learning discretization

The FGN model couples continuous measurements and discrete states via the discretizer distributions ψ^i. We tested our ability to learn the functions ψ^i by performing cross validation using gene expression data for the HOG and lysine models.

We used cross validation to compare three alternatives: (A) a single common predefined mixture of Gaussians, (B) using the EM algorithm described in Section 4 to learn a single common maximum likelihood ψ distribution, and (C) applying an unconstrained EM to learn variable specific ψ^i-s.

Cross validation was done as follows. For each condition, we used one of the above methods to learn the ψ distributions, using all data excluding that condition. We then iterated over all the model's variables. For each variable v, we hid its observation in the omitted condition and inferred its posterior distribution using the trained ψ's. Finally, we computed the likelihood of v's observation given the posterior.

Figure 6 shows the results of the cross validation on the HOG and lysine models. We present the distribution and the average log likelihood ratio of each of the methods B and C to the predefined discretization (method A). This comparison allows us to view the results in terms of the generalization capabilities of the optimized discretizers: negative log likelihood ratios represent cases where the refined discretization resulted in overfitting. Positive log likelihood ratios represent successful generalizations. We conclude that in both models, incorporating the variable specific discretization into the model improves performance for about 80% of the cases and also improves the average log likelihood ratio. In both cases, variable specific discretization outperforms the optimized single common discretization scheme. Interestingly, in the case of the lysine model, the common discretization scheme performs worse than the predefined discretization, as indicated by its negative average log likelihood ratio (Figure 6D).

6.2. Biological analysis of the HOG model

The response of yeast to hyperosmotic stress is mediated through two parallel MAPK upstream signaling branches, the multitarget MAP kinase Hog1 and an array of transcription factors that coordinate a complex process of adaptation by transient growth repression and modifications to glycerol metabolism, membrane structure, and more (Hohmann, 2002). Two key regulators in this response are regulated also by the general stress response pathway. We have constructed an FGN model that represents known regulatory relations in the HOG system (Fig. 4) and used it to study the transcriptional program following treatment by variable levels of KCl (O'Rourke and Herskowitz, 2004). The data we used contained observations of all the mRNA variables in the model and assignments of fixed values for the logical variables describing experimental conditions (general stress and turgor pressure). To test the prediction accuracy of the prior model, we applied the LBP inference algorithm to estimate the marginal posteriors of all logical variables. We summarize the model predictions in the *discrepancy matrix* shown in Fig. 7. The discrepancy matrix shows the correspondence between model predictions and experimental observations for each single variable under each condition. Essentially, the discrepancy matrix is the result of a leave-one-out cross validation procedure. To generate it, we examine each sensor variable Y_i in each condition. We infer the marginal posterior distribution of Y_i given the observations on all other variables (except Y_i) and compute the expected value and the probability of Y_i observation. We present the difference between the expected values and the observations in a color-coded matrix.

The discrepancy matrix reveals several important discrepancies between the current model for osmoregulation and the microarray experiments we analyzed. We discuss here briefly two major trends. The first trend affects a group of genes coding for proteins participating in the MAPK signaling cascade (*SSK1, SHO1, STE20, PBS2, CDC42, HOG1,* and more). These genes are repressed during the peak of

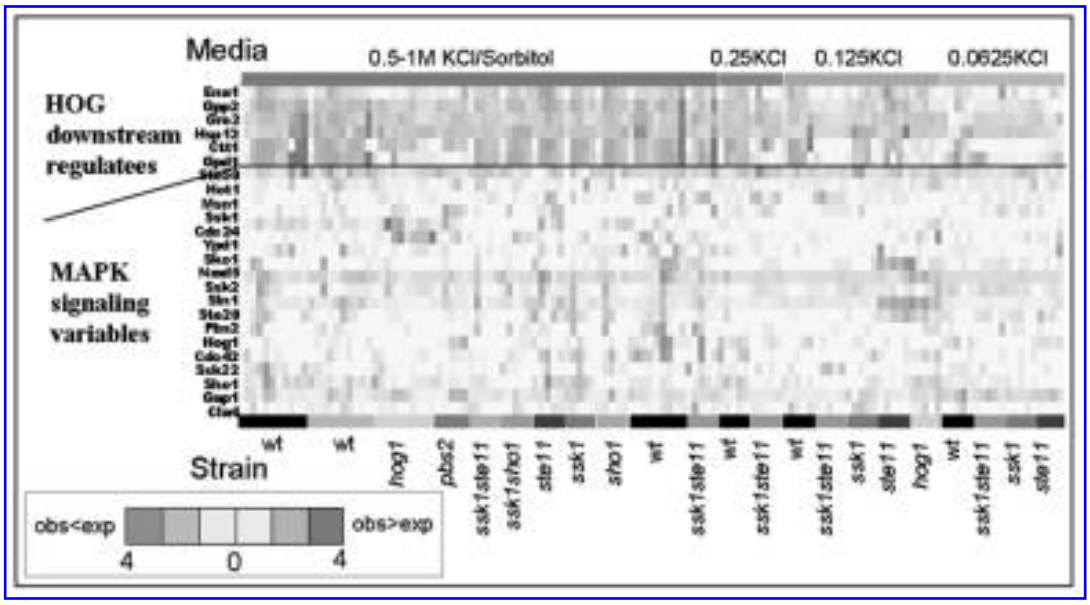

FIG. 7. Discrepancy matrix for the HOG model on data from O'Rourke *et al.* Rows correspond to mRNA variables and columns to experimental conditions from O'Rourke and Herskowitz (2004). Top bars indicate conditions in variable levels of osmotic shock. Bottom colored bars indicate groups of experiments with the same wild type or knockouts at different time points. In treatments of 0.0625 M, 0.125 M and 0.25 M KCL, each group is spanning five time points over 15, 30, and 40 minutes, respectively. In 0.5 M and 1 M, short (long) bars indicate 5 (10) experiments over 40–180 minutes. Dark cells indicate observations that are lower or higher than the expected prediction. Color intensity is proportional to minus the log likelihood of the observation.

the osmo-regulation program (10–30 minutes after treatment with 0.5 M KCl, around 60 minutes in the 1 M KCl treatment). This repression is not reported in the current literature. We hypothesize that as part of the adaptation to high levels of osmotic pressure, yeasts may reduce the sensitivity of the Hog1 signaling cascade, by slowing down the production of some central components in it.

A second group of discrepancies involves genes that are targets of the Hog1 downstream transcription factors. These genes include Sko1, Hot1, and Msn1 (Proft and Serrano, 1999; Rep *et al.*, 1999; Rep *et al.*, 2000) (Fig. 4). In many cases, the literature does not specify the logical relations among the regulators and each of their regulatees, and this lack of knowledge is manifested as discrepancies.

We used our model learning machinery to refine the regulatory logic for several model variables that are known to be affected by Hog1-downstream regulators. Figure 8 shows examples of logical relations

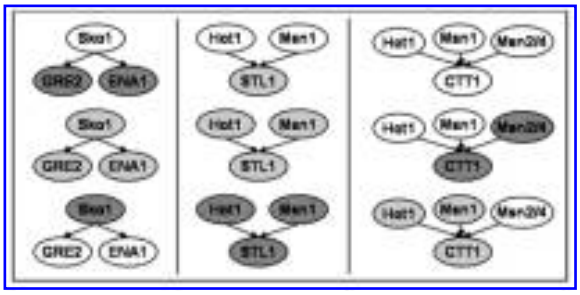

FIG. 8. Learning in the HOG model. Examples of model features learned by the FGN methodology. We show logical relations that were learned with significant p-values. Each graph depicts the regulation of one regulatee given the particular states of its regulators. Variable states are indicated by node colors: white—0, light gray—1, dark gray—2.

learned. First, we were able to learn the known repressive role of Sko1 in the regulation of *GRE2* and *ENA1* (Proft and Serrano, 1999). We learned three model features that associated high levels of the mRNA variables of these two genes with low state of the inferred Sko1 regulator state, and vice versa. The expression of the *SKO1* gene during osmotic stress is static, and the correct regulation function could only be learned given the inferred Sko1 protein activities. These inferred activities take into account, in addition to the mRNA measurements, the entire model and its regulatory functions. We also learned the regulation of *STL1*. That regulation is reported to be completely dependent on Hot1 and Msn1 (Rep *et al.*, 2000), but the literature does not clarify the logical relations among them. Our results show that although these two regulators have a positive effect on *STL1* expression, the gene can be induced even when both regulators lack any activity. We can thus hypothesize that a third factor is involved in *STL1* regulation. A third, more complex regulation function associates the Hog1 specific regulators Hot1, Msn1 and the general stress factor Msn2/4 into a single program controlling several genes. Our model contains only four representatives of a larger regulon: *GPP2*, *GPD1*, *HSP12*, and *CTT1* (Rep *et al.*, 1999). Similar results as for CTT1 (Fig. 8) were obtained also for the other three regulatees (data not shown). Our results indicate that the two signaling pathways (the HOG cascade and the general stress pathway) act in parallel, and each of the pathways can induce the regulon in the absence of activity from the other.

6.3. Biological analysis of the lysine biosynthesis model

Figure 5 shows the core of the lysine biosynthesis system in the yeast *S. cerevisiae*. It includes a linear metabolic pathway from α-ketoglutarate through αAASA to lysine, the catalyzing enzymes of the metabolic reactions (Lys1,2,9,12,20,21, and YJL200C) and their transcription control via the transcription factors Gcn4 and Lys14. Gcn4 activity is regulated during transcription, and Lys14 is influenced by the αASSA positive feedback loop, sensing the lysine biosynthesis flux. Additional feedback loops are the general nitrogen control regulation mediated by Gcn2 and the lysin negative regulation on Lys20 and Lys21. The full model includes also amino acids and ammonium (NH3), which represent the environmental conditions enforced on the system, and their transport into the cell by specific permeases (see Fig. 5 and *www.cs.tau.ac.il/∼rshamir/fgn/* for a full topology and logic).

We now wish to demonstrate the power of the feedback modeling and test the advantage of our method over former methods. The performance of the method is measured here by the capability to learn real regulation functions from real data. We thus apply cross validation in the lysine model and compare the performance of our approach to the deterministic model approach and to a naive Bayesian approach. The deterministic model approach (Gat-Viks *et al.*, 2004) learns a deterministic regulation function by optimizing a least squares score. It assumes a prior model that is 100% certain and solves the deterministic analog of the inference problem to enable the learning of a regulation function from partial observations. To allow comparison of the deterministic model with the current one, we transformed its discrete predictions into continuous distributions using predefined Gaussians. The same discretizers were used in the other two models, in order to ensure that differences in model performance were not due to the discretization. In the naive Bayesian approach, we assume that the topology of a Bayesian network over the observed variables (the mRNAs in our case) is given, and we learn the conditional probabilities of each variable separately given its regulators using complete data. The learning problem in this case is trivially solved by building a frequency table. Learning in the FGN model was done given the probabilistic function priors θ^i. We used the hypothesis testing procedure described above to repeatedly attempt the learning of regulation function features. For a variable with m regulators, we have k^m such features corresponding to each assignment of states to the regulators. For each feature, and given a p-value threshold (we used 0.01), our learning algorithm may or may not be able to decide on the correct regulatee outcome. We update the regulation function to reflect a strong θ for the feature ($\alpha = 0.99$) where a decision was made and a uniform distribution for θ where no decision could be made. We iterate the learning process until no further improvement is possible and report a regulation function in which only a fraction of the features are determined.

To perform the cross validation we repeatedly selected a variable and set its prior θ^i to the uniform distribution. We removed one condition from the dataset, learned the variable's regulation function, and used it to compute the posterior of the variable, given the omitted condition without the observation for the test variable. Figure 9 depicts the log likelihood ratio distribution for the three methods (compared to a uniform

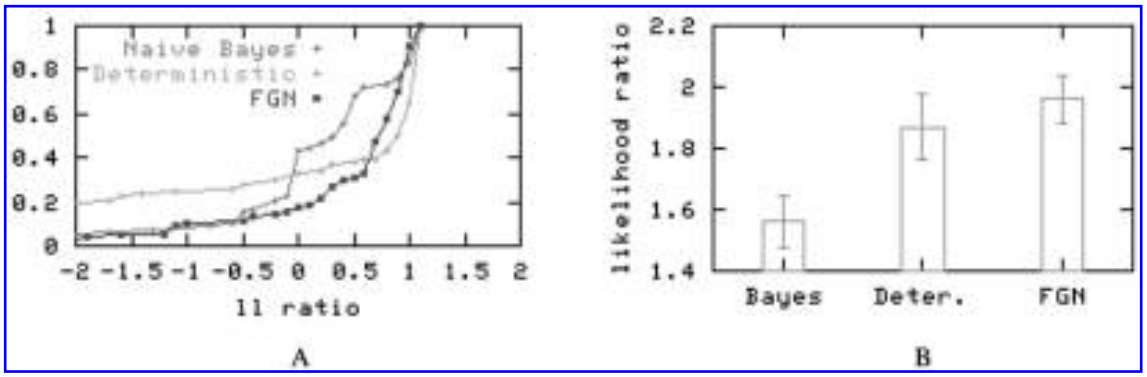

FIG. 9. Performance of different methods for learning regulation functions on the lysine model. Cumulative distributions (**A**) and averages (**B**) of the log likelihood (ll) ratio for cross validation in the lysine model using three methods for learning regulation functions: A naive Bayesian method, assuming the network topology, a deterministic learning scheme as in Gat-Viks *et al.* (2004), and learning using the FGN model. Bars indicate the predicted standard deviation of the averages.

prior model). We see that the FGN model improves over the other two methods. Detailed examination of the distribution reveals that the probabilistic model makes half as many erroneous predictions (negative log likelihood ratios) as its deterministic counterpart, probably due to its ability to evaluate statistically the learning predictions and thus avoid false positive predictions. Both the deterministic and probabilistic methods make good use of the additional knowledge, formalized into the model logic, to obtain better results than the naive Bayesian approach.

Figure 10 shows an example of how the formalized biological knowledge might improve the learning performance. In order to learn the regulation of the biosynthesys enzymes (e.g., LYS1,9,20) by their regulators Gcn4 and Lys14, our model infers the protein levels of the regulators and uses it as the basis for the learning process. Lys14 and Gcn4 are subject to a major posttranscriptional control, and thus using

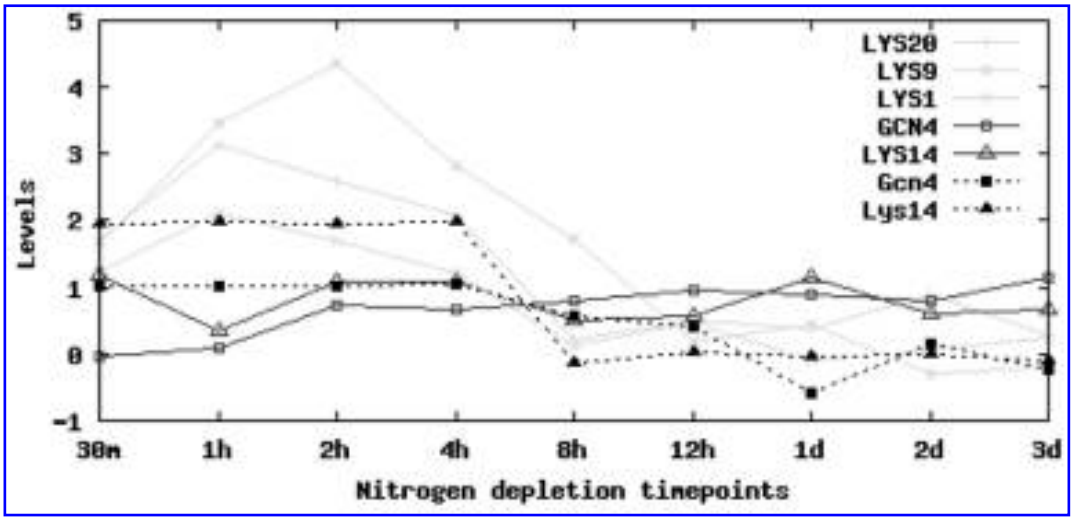

FIG. 10. States of lysine model variables in nitrogen depletion experiments. X axis: time points of the nitrogen depletion experiments of Gasch *et al.* (2000). Y axis: solid lines are measured mRNA levels; broken lines are inferred protein levels. (The mRNA levels are as explained in Gat-Viks *et al.* [2004]; protein levels are computed using a predefined discretization scheme with the arbitrary averages −1, 1, and 3). We plot the observed levels of the mRNAs of the TFs GCN4 and LYS14 in gray empty shapes, and regulatees LYS1, LYS20, and LYS9 in light gray. We also plot expected levels of the proteins Gcn4 and Lys14 as inferred by the model. Note that the mRNA levels of the TFs are roughly constant throughout the experiment, while the model-based inference highlights possible changes in the protein levels, by exploiting the connection between the protein levels and their regulators levels.

the mRNA levels to approximate the protein levels might lead to learning mistakes. We used our learning method to refine the model for the lysine biosynthetic enzymes and were able to learn the known inductive role of each of their regulators. In addition, Lys14 can activate transcription in the absence of Gcn4 activity (see *www.cs.tau.ac.il* for details). The features obtained are similar to the results of Gat-Viks *et al.* (2004), but now we can use the p-values to pinpoint the highly significant features learned.

7. DISCUSSION

In this study, we have introduced a computational framework for the study of biological systems using a combination of prior knowledge on the regulation of system's components with data from diverse high-throughput experiments. We developed a practical approach for exploiting as much of the available information on the system as possible in an integrative fashion. The goals were to systematically test the correctness of prior assumptions on the regulation of the system, by comparing predicted and observed experimental behavior, and to refine our regulation models so that possible model discrepancies are alleviated. Our mathematical formulation offers flexibility that can be used to express knowledge at all levels: In terms of the model, extant knowledge can range from confirmed and quantified regulatory relations to hypotheses and beliefs on poorly characterized parts of the system. In terms of experimental data, these can range from controlled high-throughput experiments, testing the behavior of the system from many possible angles, to high- and low-throughput experiments indicating the activity of only a small fraction of the system's factors. We believe that such a flexible and data-absorbing approach to the learning of models for biological systems is pertinent to making computational tools helpful when addressing concrete biological problems.

In developing the current framework, we have used many simplifications and limiting assumptions, trying to strike the right balance between our wish to construct a faithful description of the biological system and the scarcity of accurate experimental information at very high resolution. In the future, with the anticipated advent of refined understanding of regulatory switches, truly quantitative experiments on more aspects of biological regulation (e.g., protein abundance and states) and measurements at the single cell level, our framework could be extended in several major directions.

In its current form, our model describes the steady state behavior of the system. Biological processes are inherently temporal, but when the sampling rate (the number and time resolution of experiments) is slow relative to the rate of the regulatory mechanisms, the steady state assumption is more practical than other assumptions. We note that different regulatory processes operate on different time scales: In the typical high-throughput experimental sampling rate, the steady state assumption is highly adequate for metabolic pathways and posttranslational regulation and reasonable for transcriptional programs. The models considered in this work included variables of many types, and we validated empirically (using, e.g., cross validation) that the steady state assumption still enables biologically meaningful results with each of them. The model is already capable of handling steady state (or fast) feedback loops, and it will be natural to extend it to handle slower temporal processes in a way analogous to the construction of dynamic Bayesian networks (DBN) (Friedman *et al.*, 1998; Smith *et al.*, 2002) from steady state Bayesian networks. As in DBNs, the algorithms for inference and learning can be naturally generalized from the steady state model to the dynamic model.

Another major simplification we have applied in this work is with the modeling of logical relations using discrete functions (or distributions over discrete functions). We have used this assumption primarily since most of the current prior knowledge on transcriptional switches and other regulatory relations is essentially qualitative. It is clear however that using more biologically justifiable classes of regulation functions (e.g., Tanay and Shamir [2004], Ronen *et al.* [2002], Imoto *et al.* [2004], and Nachman *et al.* [2004]) can help to constrain the learning process toward more significant results.

A final word of caution should be added with respect to topology learning. In the current work, we assumed a fixed topology for the regulatory network. The learning of regulation functions could be performed based on that topology, with reasonable statistical power. In order to enable true topology learning in our framework, much more data or other types of restrictions (e.g., a small repertoire of model variables) would be required. The tools we developed here could be readily applied, however, in settings where structure learning is reasonable (e.g., as in Sachs *et al.* [2005]).

ACKNOWLEDGMENTS

We thank Nir Friedman, Dana Pe'er, and the anonymous referees for helpful comments. I.G.V. was supported by a Colton fellowship. A.T. was supported in part by a scholarship in complexity science from the Yeshaia Horvitz Association. D.R. was supported by a summer student fellowship from the Weizmann Institute of Science. R.S. holds the Raymond and Beverly Sackler Chair for Bioinformatics at Tel Aviv University and was supported in 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

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

Beer, M.A., and Tavazoie, S. 2004. Predicting gene expression from sequence. *Cell* 117, 185–198.

Buntine, W.L. 1995. Chain graphs for learning. *Proc. 11th Ann. Conf. on Uncertainty in Artificial Intelligence (UAI '95)*, 46–65.

Chen, K.C. *et al.* 2000. Kinetic analysis of a molecular model of the budding yeast cell cycle. *Mol. Biol. Cell* 11, 369–391.

Cooper, G. 1990. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intell.* 42, 393–405.

Covert, M.W., Knight, E.M., Reed, J.L., Herrgard, M.J., and Palsson, B.O. 2004. Integrating high-throughput and computational data elucidates bacterial networks. *Nature* 429, 92–96.

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

Friedman, N., Murphy, K., and Russell, S. 1998. Learning the structure of dynamic probabilistic networks. *Proc. 14th Conf. on Uncertainty in Artificial Intelligence*, 139–147.

Gasch, A.P., Spellman, P.T., Kao, C.M., Carmel-Harel, O., Eisen, M.B., Storz, G., Botstein, D., and Brown, P.O. 2000. Genomic expression programs in the response of yeast to environmental changes. *Mol. Biol. Cell* 11, 4241–4257.

Gat-Viks, I., Tanay, A., Raijman, D., and Shamir, R. 2005. The factor graph network model for biological systems. *Proc. RECOMB 2005*, 31–47.

Gat-Viks, I., Tanay, A., and Shamir, R. 2004. Modeling and analysis of heterogeneous regulation in biological networks. *J. Comp. Biol.* 11, 1034-1049.

Hartemink, A., Gifford, D., Jaakkola, T., and Young, R. 2002. Combining location and expression data for principled discovery of genetic regulatory networks. *Proc. 2002 Pacific Symposium in Biocomputing (PSB '02)*, 437–449.

Hohmann, S. 2002. Osmotic stress signaling and osmoadaptation in yeasts. *Microbiol. Mol. Biol. Rev.* 66(2), 300–372.

Imoto, S., Higuchi, T., Goto, T., Tashiro, K., Kuhara, S., and Miyano, S. 2004. Combining microarrays and biological knowledge for estimating gene networks via Bayesian networks. *J. Bioinform. Comp. Biol.* 2, 77–98.

Imoto, S., Kim, S., Goto, T., Aburatani, S., Tashiro, K., Kuhara, S., and Miyano, S. 2004. Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network. *J. Bioinform. Comp. Biol.* 1, 231–252.

Jaakkola, T.S. 2001. Tutorial on variational approximation methods, *in* Saad, D., and Opper, M., eds., *Advanced Mean Field Methods—Theory and Practice*, 129–160, MIT Press.

Kschischang, F.R., Frey, B.J., and Loeliger, H. 2001. Factor graphs and the sum-product algorithm. *IEEE Trans. Information Theory* 47, 498–519.

MacKay, D.J.C. 1998. Introduction to Monte Carlo methods, *in* Jordan, M.I., ed., *Learning in Graphical Models*, 175–204, Kluwer Academic Press, New York.

Nachman, I., Regev, A., and Friedman, N. 2004. Inferring quantitative models of regulatory networks from expression data. *Bioinformatics* 20, 248–256.

O'Rourke, S.M., and Herskowitz, I. 2004. Unique and redundant roles for hog mapk pathway components as revealed by whole-genome expression analysis. *Mol. Biol. Cell* 15(2), 532–542.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufmann, New York.

Proft, M., and Serrano, R. 1999. Repressors and upstream repressing sequences of the stress-regulated ena1 gene in *Saccharomyces cerevisiae*: bzip protein sko1p confers hog-dependent osmotic regulation. *Mol. Biol. Cell* 19, 537–546.

Rep, M., Krantz, M., Thevelein, J.M., and Hohmann, S. 2000. The transcriptional response of *Saccharomyces cerevisiae* to osmotic shock. hot1p and msn2p/msn4p are required for the induction of subsets of high osmolarity glycerol pathway-dependent genes. *J. Biol. Chem.* 275, 8290–8300.

Rep, M., Reiser, V., Holzmüller, U., Thevelein, J.M., Hohmann, S., Ammerer, G., and Ruis, H. 1999. Osmotic stress-induced gene expression in *Saccharomyces cerevisiae* requires msn1p and the novel nuclear factor hot1p. *Mol. Cell. Biol.* 19, 5474–5485.

Ronen, M., Rosenberg, R., Shraiman, B., and Alon, U. 2002. Assigning numbers to the arrows: Parameterizing a gene regulation network by using accurate expression kinetics. *Proc. Natl. Acad. Sci. USA* 99, 10555–10560.

Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D.A., and Nolan, G.P. 2005. Causal protein-signaling networks derived from multiparameter single-cell data. *Science* 308(5721), 523–529.

Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D., and Friedman, N. 2003. Module networks: Identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genet.* 34(2), 166–176.

Smith, V.A., Jarvis, E.D., and Hartemink, A.J. 2002. Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics* 18, 216–224.

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

Tanay, A., and Shamir, R. 2004. Modeling transcription programs: Inferring binding site activity and dose-response model optimization. *J. Comp. Biol.* 11, 357–375.

Yeang, C.H., Ideker, T., and Jaakkola, T. 2004. Physical network models. *J. Comp. Biol.* 11(2–3), 243–262.

Yedidia, J.S., Freeman, W.T., and Weiss, Y. 2004. Understanding belief propagation and its generalizations, *in* Lakemeyer, G., and Nebel, B., eds., *Exploring Artificial Intelligence in the New Millennium*, 239–269, Morgan Kaufmann, New York.

Yedidia, J.S., Freeman, W.T., and Weiss, Y. 2004. Constructing free energy approximations and generalized belief propagation algorithms. Technical report TR-2004-040, Mitsubishi Electric Research Laboratories.

Address correspondence to:
Irit Gat-Viks
School of Computer Science
Tel-Aviv University
Tel-Aviv 69978, Israel

E-mail: iritg@post.tau.ac.il