Sackler Faculty of Exact Sciences, Blavatnik School of Computer Science

Modeling and Analysis of Perturbations in Gene Regulatory Networks

THESIS SUBMITTED FOR THE DEGREE OF

"DOCTOR OF PHILOSOPHY"

by

Guy Karlebach

The work on this thesis has been carried out

under the supervision of **Prof. Ron Shamir**

Submitted to the Senate of Tel-Aviv University

July 2012

Preface

This thesis is based on the following collection of four articles that were published or submitted for publication in scientific journals throughout the PhD period.

1. Modeling and Analysis of Gene Regulatory Networks(1)

Guy Karlebach and Ron Shamir

Published in Nature Reviews Molecular Cell Biology

2. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case(2)

Guy Karlebach and Ron Shamir

Published in BMC Systems Biology

3. Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy Based Approach(3)

Guy Karlebach and Ron Shamir

Published in Journal of Computational Biology

4. **A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets**(4)

Guy Karlebach and R. Shamir

Submitted

Abstract

An important goal in contemporary biomedical research is to understand biological systems. System level understanding requires large quantities of experimental data and algorithms that can infer system behavior from them. Gene regulatory networks (GRNs) are of particular interest since they regulate core cellular processes such as the cell cycle and embryonic development. A GRN is a network formed by a set of genes and regulatory interactions between them. For example, a gene can turn on the expression of another gene, which will in turn shut off the expression of the first gene. Real GRNs are complex and contain many feedback loops.

It has been conjectured for various biological phenomena that specific dynamic behaviors of a GRN correspond to physiological states. A dynamic behavior is in essence a sequence of gene expression patterns that changeover time.

Recent years saw the development of a new brand of experimental techniques that are termed ''High Throughput Technologies''. These technologies enable the experimentalist to measure the states of genes, proteins and even binding of proteins to DNA on a genome-wide scale. They can be used to probe the state of the GRN at different times and under different conditions, and from this information a mathematical model can be created and used for predictions. However, there is a downside to high throughput technologies – the data that they produce are incomplete and noisy. This is an additional layer of complexity on the already intricate task of modeling a complex system. Therefore, many labs worldwide have commenced in an effort to develop computational methods that can correct and extrapolate high throughout data with the final goal of creating predictive models. Once such methods are available they will allow us to study this important biological mechanism and we may learn to affect processes that are to-date poorly understood.

In this thesis I present a new methodology for predicting the effect of perturbations on GRNs. I show that useful conclusions about the effect of perturbations can be drawn without knowing the network structure exactly. I characterize the difficulties in fitting gene expression data to a logical model using computational complexity theory, and give a heuristic algorithm to perform this task. I demonstrate the effectiveness of the algorithm by reconstructing the mouse embryonic stem cell network. I develop an algorithm that given an ensemble of alternative network structures finds minimal perturbations that induce a desired network behavior. I also present a randomized approach that improves the running time of the latter algorithm significantly.

Contents

1. Introduction 6
1.1 The systems approach to biology 6
1.2 Gene regulatory network models 6
1.2.1 Logical models 7
1.2.2 Continuous models 8
1.2.3 Molecular-level models 8
1.3 Experimental data and network models 8
1.3.1 Robust model construction 9
1.3.2 Robust dynamical analysis 10
1.4 Computational problems in modeling 11
1.4.1 State explosion 11
1.4.2 Complexity of asynchronous systems 11
1.4.3 The curse of dimensionality 12
1.5 Research objectives 12
2. Summary of articles included in this thesis 14
2.1 Modeling and analysis of gene regulatory networks 14
2.2 Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case 15
2.3 Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy Based Approach 16
2.4 A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets 17
3. Modeling and analysis of gene regulatory networks 18
4. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case 30
5. Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy Based Approach 45
6. A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets 58
7. Discussion 78
7.1 Logical models as the gold standard for gene network analysis 78
7.2 Open questions 78
7.3 Future research 80
Bibliography 82

Appendix 89
Glossary 89

1. Introduction

1.1 The systems approach to biology

Up to about a decade ago most biologists conducted research based on the reductionist approach. This approach assumes that in order to understand biological systems one needs to examine each component separately. The emerging discipline of Systems Biology advocates that the most important properties of biological systems emerge from the system as a whole and are not always evident in each component separately(5,6). Driving this new approach are new experimental technologies that produce genome-wide measurements and have become known as "high-throughput technologies". High throughput technologies are rapidly improving in accuracy and scale while innovative ways to probe cellular systems are developed (7-11). To exploit this abundance of data, computational methods are developed alongside "wet lab" techniques and are primed to complement them.

A distinctive property that a system can exhibit is dynamic behavior(*1*). Dynamic behavior can be viewed as a propagation of changes of the local states of single components as triggered by the local states of other components. Indeed, the amounts and the temporal pattern in which molecular species appear in cells play a major role in orchestrating the processes of life. In particular, gene regulatory networks govern the levels of gene expression and hence studying these networks is high on the agenda of Systems Biology.

1.2 Gene regulatory network models

Like any phenomenon in nature, gene networks can be described using different levels of complexity. While more detailed descriptions can be more accurate, they require finer measurements that may be hard to obtain. However, sometimes inaccurate descriptions can also contribute to our understanding of nature (12). On the other hand, when abstraction renders the network into an obscure mathematical entity, little can be gained from its analysis. Therefore, a key question that should precede any discussion about analysis of biological networks is what kind of model describes the network?

Various computational models have been developed for regulatory network analysis. These models can be roughly divided into three classes. The first class, logical models, describes regulatory networks qualitatively. They allow users to obtain a basic understanding of the different functionalities of a given network under different conditions. Their qualitative nature makes them flexible and easy to fit to biological phenomena, although they can only answer qualitative questions. To understand and manipulate behaviors that depend on finer timing and exact molecular concentrations, a

second class of models was developed — continuous models. For example, to simulate the effects of dietary restriction on yeast cells under different nutrient concentrations, users must resort to the finer resolution of continuous models (13). A third class of models was introduced following the observation that the functionality of regulatory networks is often affected by noise. As the majority of these models account for interactions between individual molecules, they are referred to here as single molecule-level models. Single molecule-level models explain the relationship between stochasticity and gene regulation. The different classes are described in more detail in the following subsections.

1.2.1 Logical models

The most basic and simplest modeling methodology is discrete and logic based, and was introduced by Kauffman and Thomas before the emergence of high throughput technologies (14,15). Logical models represent the local state of each entity in the system (for example, genes, proteins and small molecules) at any time as a discrete level, and the temporal development of the system is often assumed to occur in synchronous, discrete time steps. Entity levels are updated at each time step according to regulation functions. Discrete modeling allows researchers to rely on purely qualitative knowledge. Such models can be analyzed using a broad range of well established mathematical methods.

Boolean networks are the simplest discrete model and many other discrete models share similarities with Boolean networks. Boolean regulatory networks were first presented by Kauffman(15). In a Boolean network, an entity can attain two alternative levels: active (1) or inactive (0). For example, a gene can be described at any time as expressed or not expressed. The level of each entity is updated according to the levels of several entities, via a specific Boolean function. The 0–1 vector that describes the levels of all entities is called the system's state, or the global state. It is assumed to change synchronously, such that at every time step, the level of each entity is determined according to the levels of its regulators at the previous time step and according to the entity's regulation function. An associated computational problem is exploring the state transition graph of large networks, a problem that stems from the fact that the number of global states is exponential in the number of entities. Methods that exhaustively enumerate of all the possible trajectories are only practical for small networks(16).

1.2.2 Continuous models

Biological experiments usually produce real, rather than discrete valued, measurements. Examples include reaction rates, cell mass, cell cycle length and gene expression intensities(17,18). In most of the works that used logical models real valued measurements were discretized, a process that reduces the accuracy of the data. Continuous models, using real valued parameters over a continuous timescale, allow a straightforward comparison of the global state and experimental data and can theoretically be more accurate. In practice, however, quantitative measurements are almost always partial (that is, they cover only a fraction of the system's entities). Therefore, some of the parameters of continuous models are usually based on estimations or inference. This may potentially introduce more errors than discretization, but nevertheless the continuous approach has many followers (19-21). Large metabolic networks have also been successfully modeled in steady state using continuous approaches, which shows that a steady state assumption can be useful for inference of continuous parameters (22).

1.2.3 Molecular-level models

Even continuous models that are based on enzyme kinetics(23) are inaccurate when one delves into a finer level of description using single-cell measurements (24,25).At a finer level biological networks contain stochastic components and may manifest different behaviors starting from the same initial conditions(26,27). This is particularly true for regulatory networks, where the number of regulatory molecules is often low (28,29). How important is stochasticity to the understanding of gene networks in general? An instructive example comes from the phage lambda model of McAdams, Arkin and Ross(30). They showed that the phage's choice between the lysogenic and lytic pathways depends on a stochastic mechanism. The presence of stochasticity in gene regulatory networks needs to be further studied and kept in mind when modeling such networks. Some of the computational problems that arise at this level of detail are the need to use simulation and its high computational cost (31).

1.3 Experimental data and network models

Despite the new tantalizing capabilities that high modern experimental techniques offer, the data that they produce require careful interpretation. To elaborate on this point we shall consider gene expression measurement using microarrays, and similar arguments can be made with regard to other high-throughput technologies. A microarray is an array of typically thousands of oligonucleotide probes that hybridize with fluorescently-

tagged complementary oligonucleotides called targets. In order to measure the level of gene expression, one extracts mRNA from a population of cells and applies it as targets for the microarray(*2, 3*). The measurement of gene expression is done at a specific time point. Clearly this time point has to be chosen carefully since cells are dynamic. In addition, if one is interested in changes over time the intervals between different time points also need to be decided upon. Also, the measurement is a continuous intensity level – what does each intensity mean, and is that meaning the same for all the genes? Another problem is that we are measuring an average over a population of cells that are sometimes hard to synchronize. Does this averaging process render the mixture uninterruptable? Finally, how noisy is microarray technology? Can this noise be corrected? Practical utilization of gene expression data depends on answers to all these questions. That is also the case for other high throughput data like ChIP on chip. Therefore, computational techniques and in particular gene network models often offer methods to tackle such problems.

The fact that most available experimental data are inaccurate is widely acknowledged by the modeling community. Modeling tools are often designed to handle inaccuracies and to generate predictions that can be interpreted despite these inaccuracies. A modeling methodology that produces the same predictions from datasets that differ only by noise is called robust.

1.3.1 Robust model construction

Akutsu et al. proposed a polynomial algorithm that infers regulatory interactions from experimental data by finding for each gene a Boolean function that predicts its level with maximal accuracy (32). The inputs of that function are the levels of the gene's regulators. This algorithm requires that continuous expression data first be discretized into Boolean values, i.e. that each real value will be converted into a Boolean one, and then it selects the function and regulators that are in best agreement with the discretized data. A later extension allows each discretized sample to be associated with a continuous confidence value (33), namely, the reliability of each microarray profile (a vector of gene expression values) in the dataset. Akutsu et al. also studied the case in which only partial experimental data are available, and showed that learning the regulation functions in this setting is NP-complete (34).

Segal et al. (35) developed a methodology that uses expression data for inferring regulatory functions formulated as decision trees: each node of the tree corresponds to a regulator and some threshold value, and the level of the regulatee is determined by traversing the tree from root to leaf, selecting a child at each node by comparing the

regulator's continuous expression level to the threshold. The algorithm clusters genes into groups that have a similar expression pattern and assigns to every cluster its set of regulators.

Shamir and Tanay presented an efficient algorithm that assumes a monotone relationship between a transcription factor's (TF) continuous level, its affinity to a target gene and the strength of regulation, and uses this assumption to determine whether or not a target gene is activated. Since their algorithm requires TF-target affinities, they also suggested a method for inferring the affinity of a TF to its target genes based on analysis of the promoters of regulated genes(36).

Other approaches to model reconstruction can be found in(21,35,37,38). The quality of a static model that one can construct depends critically on the properties of high throughout input data. As a consequence, the predictions that a model generates also depend on these properties.

1.3.2 Robust dynamical analysis

Probabilistic Boolean networks (PBNs) are one approach to generating predictions based on gene expression data despite the inherent inaccuracies (39,40). PBNs generalize the Boolean network model such that an entity can have several regulation functions, and each of which is given a probability based on its compatibility with prior data. At each time step, every entity is subjected to a regulation function that is randomly selected according to the defined probabilities. Hence the model is stochastic and an initial global state can lead to many trajectories of different probabilities. The resulting model, the probabilistic Boolean network, generates a sequence of global states that constitutes a Markov chain. For example, a PBN was used to model a 15-gene subnetwork that was inferred from human glioma expression data (39). This analysis demonstrates that the stationary distributions of entities may indicate possible regulatory relationships among them: entities that have the same states in a significant proportion of the global states are likely to be related. A problem with this approach is that the Markov chain will have a number of states that is exponential in the number of entities. Another problematic point is that in a single trajectory a gene can be regulated by several different regulation functions, which seems to somewhat depart from biological reality.

Dynamics in which not all entities are subjected to regulation functions at every time step is called asynchronous. Steggles et al. relied on the asynchronous dynamics of Petri nets in order to generate more robust predictions (41). They showed that the asynchronous nature of Petri nets can be used to allow some degree of freedom in the

choice of regulation functions. As in the case of PBNs, several different functions can regulate the same gene in a single trajectory. In contrast to probabilistic Boolean networks, regulation functions are not assigned probabilities. There is no quantification of the likelihood of different network structures. They used their approach to model the regulatory network of *Bacillus subtilis* sporulation and obtained a behavior that is in good agreement with existing literature. For example, when initializing the system to a global state that corresponds to vegetative growth and activating the sporulation signal, the dynamics of the system leads to a state that corresponds to sporulation. Their model also correctly predicted the sporulation capabilities of mutants. Similar approaches to handling incomplete models can be found in (42-44).

1.4 Computational problems in modeling

Modeling raises some inherent computational problems that are independent of the sources of data from which the model is constructed. These problems are characterized by the need to search a space that grows very rapidly with the size of the model.

1.4.1 State explosion

A model that has N entities and each entity can assume 2 states or more will have at least 2^N global states. Exhaustively searching this state space for particular states, for example steady states of a gene regulatory network, is only practical for small models. While statistical methods can provide general insights about network behavior (45), searching the state space of specific models is limited by computational complexity. Even for a Boolean network, finding a steady state is NP-Complete (46).Counting the steady states of a Boolean network is #P-complete (47). Finding Boolean network inputs that will drive the network to a particular state is NP-Hard (48). One can prune the state space by considering a single initial network state and only close reachable states. If the model is synchronous then there is only one trajectory and we can follow this trajectory for a sequence of k steps stating from the initial state. It has been argued that synchronicity in gene regulatory networks is an oversimplification and yields essentially different dynamic behavior than the more realistic asynchronicity (49). If a model is asynchronous, one initial state can lead to an exponential number of states that are separated from the initial states by at most k traversals on the state transition graph. A simple example that illustrates this fact is a Boolean network in which each node regulates itself.

1.4.2 Complexity of asynchronous systems

From a purely biological perspective asynchronous models seem to have clear advantages for describing the dynamics of a gene network. There's apparently no global

clock that synchronizes the expression times of all the genes, and therefore there is no reason to impose synchronicity in the model. However, synchronous models are much simpler to analyze and on these grounds they have been often preferred(50-52). The relationship between the maximal number of edges in the state transition graph of a synchronous model and an asynchronous model is quadratic. In other words a node in the graph of the latter can have N outgoing edges, compared to at most one in the graph of the former. Results from the field of model checking (53-55) predict that analysis of asynchronous models can be computationally expensive. Notably, Thieffry and colleagues created predictive asynchronous models that contain several dozens of entities (56,57).

1.4.3 The curse of dimensionality

In the context of network modeling, the curse of dimensionality refers to the large number of parameter sets that can be assigned to a model. Even if the topology is known perfectly, different parameter sets can yield entirely different behaviors (58). It was recently shown that resolving discrepancies between a logical model and experimental data is NP-Complete (3) . Sensitivity analysis can prioritize parameters according to the impact of change in their value on the model, thereby reducing the number of parameter sets that need to be explored(59).The size of the parameter space can also be decreased by using a simple model such as a linear model(21). Another approach circumvents parameter space search by generating predictions from an ensemble of alternative parameter sets(42). The impact of a few wrong parameters on dynamic behavior can be profound and therefore searching the parameter space efficiently is a problem of primary importance (60).

1.5 Research objectives

The goals of this research are to classify and compare current modeling strategies, characterize and study computational problems associated with modeling of GRNs, develop solutions for these problems and improve existing methods for incorporating high-throughput data into a model. Chapter 3 reviews modeling methods and classifies them into different types according to various criteria. Chapter 4 presents an algorithm that predicts minimal network perturbations given an ensemble of alternative network structures. Chapter 5 analyzes the problems of network reconstruction and describes an algorithm that aims to solve some of these problems. Chapter 6 shows how to improve the running time of the minimal perturbation algorithm of Chapter 4 by adopting a randomized strategy. Chapter 7 tries to predict where the field is going and specifies my future research plans.

I prove that even when the network structure is known, fitting high throughput gene expression data to a logical model is NP-Complete. This proof is based on a reduction from the vertex cover problem(61). Given an instance of the vertex cover problem a simple acyclic GRN can be constructed such that fitting a set of two microarray experiments to it will provide a solution to the vertex cover problem. Since it is easy to verify that a dataset fits a model, the fitting problem is in NP meaning that it is NP-Complete.

I develop an algorithm that interprets continuous high throughput values probabilistically in order to find a good fit of data to a model. The algorithm computes the conditional entropy of every regulatee given his regulators, based on the probabilistic interpretation of the data, and minimizes the sum of conditional entropies using gradient descent (62). In order to obtain truth tables in which every regulator can change the output value in some state, I develop a branch and bound algorithm that finds an optimal solution under this constraint (63).

As an approach to generate predictions of gene network response to perturbations I suggest to add a desired certainty level to queries about network behavior. Based on this desired certainty level the state space of alternative perturbed models can be searched such that the proportion of a model's state space that is searched depends on that model's probability. The choice of certainty level is a tradeoff between certainty on one side and speed and minimal perturbation size on the other side. A modification of McMillan's unfolding algorithm is used in order to optimize performance (55).

In order to improve the utilization of model checking techniques in predicting gene network dynamic behavior, in particular the latter method for finding minimal perturbations, I design and implement a Monte Carlo version of McMillan's unfolding algorithm. I modify the improved unfolding algorithm of Esparza et al. and show that the Monte Carlo algorithm is significantly faster than the deterministic algorithm (64).

2. Summary of articles included in this thesis

2.1 Modeling and analysis of gene regulatory networks

Guy Karlebach and Ron Shamir

Published in *Nature Reviews Molecular Cell Biology*

Various computational models have been developed for regulatory network analysis. These models can be roughly divided into three classes. The first class, logical models, describes regulatory networks qualitatively. They allow users to obtain a basic understanding of the different functionalities of a given network under different conditions. Their qualitative nature makes them flexible and easy to fit to biological phenomena, although they can only answer qualitative questions. To understand and manipulate behaviors that depend on finer timing and exact molecular concentrations, a second class of models was developed — continuous models. For example, to simulate the effects of dietary restriction on yeast cells under different nutrient concentrations, users must resort to the finer resolution of continuous models. A third class of models was introduced following the observation that the functionality of regulatory networks is often affected by noise. As the majority of these models account for interactions between individual molecules, they are referred to here as single-molecule level models. Single-molecule level models explain the relationship between stochasticity and gene regulation. Here we review the available methodologies for modeling and analyzing regulatory networks. These methodologies have already proved to be a valuable research tool, both for the development of network models and for the analysis of their functionality. We discuss their relative advantages and limitations, and outline some open questions regarding regulatory networks, including how structure, dynamics and functionality relate to each other, how organisms use regulatory networks to adapt to their environments, and the interplay between regulatory networks and other cellular processes, such as metabolism.

2.2 Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case

Guy Karlebach and Ron Shamir

Published in *BMC Systems Biology*(*4*)(*4*).

Mathematical modeling of biological networks is an essential part of Systems Biology. Developing and using such models in order to understand gene regulatory networks is a major challenge. We present an algorithm that determines the smallest perturbations required for manipulating the dynamics of a network formulated as a Petri net, in order to cause or avoid a specified phenotype. By modifying McMillan's unfolding algorithm, we handle partial knowledge and reduce computation cost. The methodology is demonstrated on a glioma network. Out of the single gene perturbations, activation of glutathione S-transferase P (GSTP1) gene was by far the most effective in blocking the cancer phenotype. Among pairs of perturbations, NFkB and TGF-ß had the largest joint effect, in accordance with their role in the EMT process. Our method allows perturbation analysis of regulatory networks and can overcome incomplete information. It can help in identifying drug targets and in prioritizing perturbation experiments.

2.3 Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy Based Approach

Guy Karlebach and Ron Shamir

Published in *Journal of Computational Biology*

Models of gene regulatory networks (GRNs) attempt to explain the complex processes that determine cells' behavior, such as differentiation, metabolism, and the cell cycle. The advent of high-throughput data generation technologies has allowed researchers to fit theoretical models to experimental data on gene-expression profiles. GRNs are often represented using logical models. These models require that real-valued measurements be converted to discrete levels, such as on/off, but the discretization often introduces inconsistencies into the data. Dimitrova et al. posed the problem of efficiently finding a parsimonious resolution of the introduced inconsistencies. We show that reconstruction of a logical GRN that minimizes the errors is NP-complete; so that an efficient exact algorithm for the problem is not likely to exist. We present a probabilistic formulation of the problem that circumvents discretization of expression data. We phrase the problem of error reduction as a minimum entropy problem, develop a heuristic algorithm for it, and evaluate its performance on mouse embryonic stem cell data. The constructed model displays high consistency with prior biological knowledge. Despite the oversimplification of a discrete model, we show that it is superior to raw experimental measurements and demonstrates a highly significant level of identical regulatory logic among co-regulated genes.

2.4 A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets

Guy Karlebach and Ron Shamir

Submitted

Petri nets are a modeling formalism for concurrent systems. Given a Petri net and its initial state, determining if a target state or a set of states is reachable is a fundamental problem. McMillan's unfolding algorithm constructs a compact representation of a Petri net's state space. However, the algorithm can solve in practice only very small reachability problems, due to the computational resources required. We developed a Monte-Carlo algorithm based on McMillan's unfolding for solving the reachability problem on Petri nets. The algorithm repeatedly constructs random prefixes of the state space representation, thereby avoiding some of the computational problems that arise when the full representation is constructed. Our tests show that the randomized algorithms can solve problems size greater than 100 within seconds, and it is faster than the deterministic algorithm by several orders of magnitude.

3. Modeling and analysis of gene regulatory networks

Modelling and analysis of gene regulatory networks

Guy Karlebach and Ron Shamir

Abstract | Gene regulatory networks have an important role in every process of life, including cell differentiation, metabolism, the cell cycle and signal transduction. By understanding the dynamics of these networks we can shed light on the mechanisms of diseases that occur when these cellular processes are dysregulated. Accurate prediction of the behaviour of regulatory networks will also speed up biotechnological projects, as such predictions are quicker and cheaper than lab experiments. Computational methods, both for supporting the development of network models and for the analysis of their functionality, have already proved to be a valuable research tool.

Stochasticity
The property of a system whose behaviour depends on probabilities. In a model with stochasticity, a single initial state can evolve into several different trajectories, each with an associated probability.

The Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. Correspondence to R.S. e-mail: rshamir@tau.ac.il
doi:10.1038/nrm2503
Published online
17 September 2008

The genome encodes thousands of genes whose products enable cell survival and numerous cellular functions. The amounts and the temporal pattern in which these products appear in the cell are crucial to the processes of life. Gene regulatory networks govern the levels of these gene products. A gene regulatory network is the collection of molecular species and their interactions, which together control gene-product abundance. Numerous cellular processes are affected by regulatory networks.

Innovations in experimental methods have enabled large-scale studies of gene regulatory networks and can reveal the mechanisms that underlie them. Consequently, biologists must come to grips with extremely complex networks and must analyse and integrate great quantities of experimental data. Essential to this challenge are computational tools, which can answer various questions: what is the full range of behaviours that this system exhibits under different conditions? What changes are expected in the dynamics of the system if certain parts stop functioning? How robust is the system under extreme conditions?

Various computational models have been developed for regulatory network analysis. These models can be roughly divided into three classes. The first class, logical models, describes regulatory networks qualitatively. They allow users to obtain a basic understanding of the different functionalities of a given network under different conditions. Their qualitative nature makes them flexible and easy to fit to biological phenomena, although they can only answer qualitative questions. To understand and manipulate behaviours that depend on finer timing and exact molecular concentrations, a second class of models was developed — continuous models. For example, to simulate the effects of dietary restriction on yeast cells under different nutrient concentrations[1], users must resort to the finer resolution of continuous models. A third class of models was introduced following the observation that the functionality of regulatory networks is often affected by noise. As the majority of these models account for interactions between individual molecules, they are referred to here as single-molecule level models. Single-molecule level models explain the relationship between stochasticity and gene regulation.

Predictive computational models of regulatory networks are expected to benefit several fields. In medicine, mechanisms of diseases that are characterized by dysfunction of regulatory processes can be elucidated. Biotechnological projects can benefit from predictive models that will replace some tedious and costly lab experiments. And, computational analysis may contribute to basic biological research, for example, by explaining developmental mechanisms or new aspects of the evolutionary process.

Here we review the available methodologies for modelling and analysing regulatory networks. These methodologies have already proved to be a valuable research tool, both for the development of network models and for the analysis of their functionality. We discuss their relative advantages and limitations, and outline some open questions regarding regulatory networks, including how structure, dynamics and functionality relate to each other, how organisms use regulatory networks to adapt to their environments, and the interplay between regulatory networks and other cellular processes, such as metabolism.

© 2008 Macmillan Publishers Limited. All rights reserved

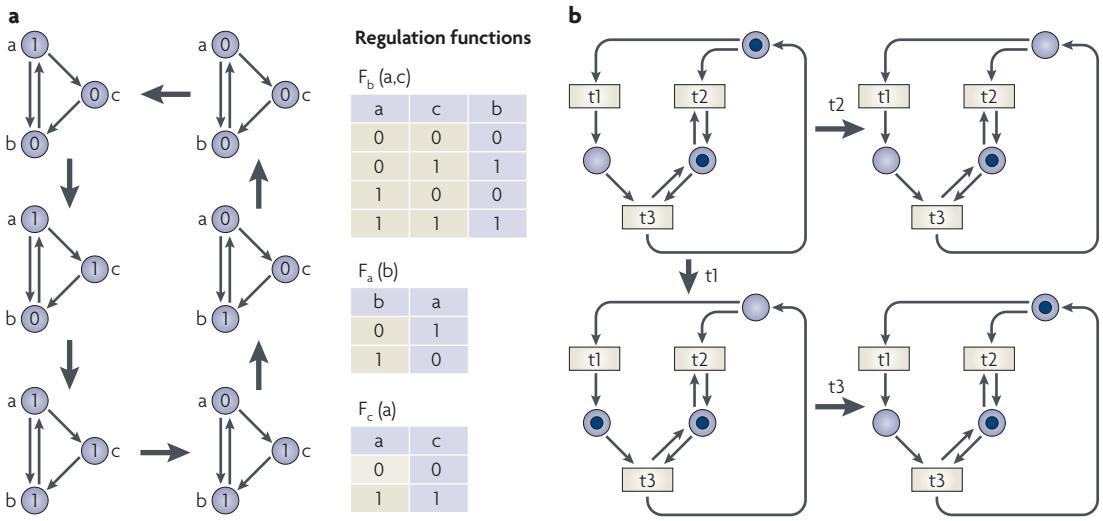


Figure 1 | **Logical models. a** | A Boolean network. Each of the entities a, b and c in the network can be in state 0 or 1. State transitions obey the regulation functions shown on the right, which describe the rules of the model. For example, if a is in state 1 and c is in state 0, at the next time step the state of b will be 0. Thin arrows indicate the regulators of each node. Time steps are represented by thick arrows. The global state of the model is the combination of the three entity states. The system cycles through the six global states. A sequence of consecutive global states is called a trajectory. **b** | A Petri net. The net contains 'places' (light blue circles) that are the model's entities, and 'transitions' (rectangles) that constitute the regulation functions and define the model's dynamics. Arcs connect input places to transitions, and transitions to their output places. Places that receive discrete values are called tokens (dark blue dots). A transition that is activated, or 'fired', reduces the tokens in its input places and increases the number of tokens in each of its output places. At any time step, every transition that has enough tokens in its input places may be fired. In the example, every transition consumes one token from every input place, and produces one token at every output place. Labels at thick arrows indicate which transition fired. Transitions t1 and t3 can be fired in alternation indefinitely, whereas no other transition can be fired after t2 has fired.

Local state
At any time point, the value representing the status of an entity in a model is its (local) state. For example, the state of a protein may indicate whether it is phosphorylated or not (a Boolean value), or the time since its last phosphorylation (a continuous value).

Synchronous model
A model wherein the time steps at which the global state changes are discrete and (usually) equally spaced. On each step, all the states are updated simultaneously, depending on the model's regulation functions and on the global state at the previous step. In asynchronous models, system changes are not confined to specific times and global states do not progress according to 'a common clock'. Time is often continuous, and entities may change their states at different times.

Regulation function
A rule that determines the state of a specific entity in the model as a function of the states of some (other) entities. For example, several transcription factors may together regulate the expression of a gene. The set of entities whose states determine the state of entity X are entity X's regulators.

Global state
The combination of all the local states of a model at one time point.

Steady state
A global state that, once reached, always repeats itself in a trajectory. Another important dynamic behaviour in biological systems is a cycle of global states. For example, the oscillations observed in the cell cycle.

Robustness
A measure of a model's ability to withstand changes without changing its essential properties. For example, in network models, robustness can be quantified as the fraction of edge additions and/or removals that change the trajectory that emanates from some initial state.

Logical models

The most basic and simplest modelling methodology is discrete and logic-based, and was introduced by Kauffman and Thomas[2,3]. The reconstruction of the regulatory network that controls the development of sea urchin embryos[4,5] is a seminal example of the profound insights that qualitative examination of regulatory network models can provide. This work demonstrates how maternal cues initiate the activity of the regulatory network and how this network orchestrates the developmental process. Logical models represent the local state of each entity in the system (for example, genes, proteins and small molecules) at any time as a discrete level, and the temporal development of the system is often assumed to occur in synchronous, discrete time steps. Entity levels are updated at each time step according to regulation functions (FIG. 1a). Discrete modelling allows researchers to rely on purely qualitative knowledge. Such models can be analysed using a broad range of well established mathematical methods.

Boolean networks. Boolean regulatory networks were first presented by Kauffman[2,6]. In a Boolean network, an entity can attain two alternative levels: active (1) or inactive (0). For example, a gene can be described as expressed or not expressed at any time. The level of each entity is updated according to the levels of several entities, via a specific Boolean function. The 0–1 vector that describes the levels of all entities is called the system's state, or the global state. It is assumed to change synchronously, such that at every time step, the level of each entity is determined according to the levels of its regulators at the previous time step and according to the regulation function (FIG. 1a).

Boolean networks were recently used to analyse the relationship between regulation functions and network stability in the yeast transcriptional network, using only the network's structure[7]. According to this study, the network is stable when random regulation functions are used, and solution stability increases when the regulation functions are biologically meaningful. It also showed that Boolean networks do not correctly model the dynamics of a transcription factor that downregulates its own expression, due to the model's limited level of detail. Another problem is that it is computationally expensive to analyse the dynamics of large networks, as the number of global states is exponential in the number of entities. However, when the number of entities is small and only qualitative knowledge is available, Boolean networks can provide important insights, such as the existence and nature of steady states or network robustness.

To study the dynamics of cell-cycle regulation in yeast, Li *et al.*[8] constructed a literature-based Boolean network in which all the regulation functions are threshold functions. This model generated trajectories with a high degree of overlap, most of which led into a path that corresponded to the cell-cycle phases of yeast. In addition, most small changes in the model did not significantly change its dynamic behaviour, indicating that it is robust. As the analysis relied on an exhaustive enumeration of all the possible trajectories, this method is only practical for small networks.

© 2008 Macmillan Publishers Limited. All rights reserved

In many cases, the regulatory relationships between network components have not been established, and therefore need to be derived from experimental data. For any entity under a Boolean network model, both its regulators and a regulatory function that is consistent with a set of gene-expression profiles can be found efficiently, provided that the number of regulators of each entity does not exceed a set limit[9]. Such an algorithm is faster than a previous one proposed by Akutsu and colleagues[10]. Lahdesmaki *et al.*[9] also presented an algorithm for selecting a set of candidate regulation functions in the presence of contradictory evidence, whereby each expression profile is associated with a certainty level (that is, a numerical value that expresses one's confidence in the profile). This algorithm was tested by deriving regulation functions for 5 yeast cell-cycle regulated genes using expression profiles of 733 candidate regulators[11]; the maximum number of regulators that together regulate a single gene was first set to 1, then 2 and finally 3. The analysis yielded a large number of regulation functions that were equally consistent with experimental data. Some of the suggested functions matched previous findings.

Probabilistic Boolean networks. Often, due to insufficient experimental evidence or incomplete understanding of a system, several candidate regulatory functions may be possible for an entity. This raises the need to express uncertainty in the regulatory logic. Shmulevich *et al.*[12,13] addressed this idea by modifying the Boolean network model such that an entity can have several regulation functions, each of which is given a probability based on its compatibility with prior data. At each time step, every entity is subjected to a regulation function that is randomly selected according to the defined probabilities[12]. Hence the model is stochastic and an initial global state can lead to many trajectories of different probabilities. The new model, the probabilistic Boolean network (PBN), generates a sequence of global states that constitutes a Markov chain[14]. For example, a PBN was used to model a 15 gene sub-network that was inferred from human glioma expression data[13]. This analysis demonstrates that the stationary distributions of entities may indicate possible regulatory relationships among them: entities that have the same states in a significant proportion of the global states are likely to be related. As the number of global states in the gene sub-network was prohibitively large, one study[13] estimated the stationary distribution by sampling the global states[15].

MetaReg. An exponential number of global states makes it difficult to analyse the dynamics of all but tiny models. In some cases, analysis under steady state conditions turns out to be a practical goal. Gat-Viks *et al.*[16] developed the MetaReg model, in which entities can have several levels (typically 3–5) and regulation functions are discrete. Two efficient heuristics were developed: the first detects a network's steady states and the second selects regulation functions that are most consistent with these steady states. The former heuristic can be used to analyse the dynamics of the network, whereas the latter can complete or correct a literature-based network. MetaReg was used to analyse the regulation of lysine biosynthesis in yeast and indicated previously unknown transcriptional controls of several metabolic enzymes.

To express uncertainty in regulation functions, Gat-Viks *et al.*[17] created a probabilistic version of the MetaReg model. In this model, an entity can have one of several possible regulation functions (with the same regulators), and probabilities that each one is correct. Technically, the model is represented as a factor graph (an expansion of Bayesian networks)[18]. Analogously to the model in REF. 16, it can be subjected to steady state identification and optimization of regulation function[18,19]. It can also discover new regulatory relationships. The method has been improved[20] to facilitate changes in the network structure (refinement) and inclusion of additional entities (expansion). Analysis of a network of 4 interconnected osmotic stress-related yeast signalling pathways, which consists of 43 entities, along with 106 expression profiles, identified novel regulatory modules and crosstalks between pathways. Thus, the model can correct and expand a known regulatory network.

Petri nets. The dynamics of a regulatory network can also be analysed using Petri nets[21], non-deterministic models (FIG. 1b). An example of a question that users can ask with a Petri net is: how many transition sequences lead from global state A to global state B? The qualitative description of biochemical reactions using a Petri net is straightforward, and Petri net models are useful analysis tools for large metabolic networks[22–24]. Chaouiya *et al.* showed that Petri nets can also model regulatory networks using Boolean regulatory functions[25], and that the metabolic and regulatory layers can be connected[26]. Steggles *et al.* proved that the synchronous dynamics of a Boolean network can be captured by a Petri net[27] and demonstrated that uncertainty in the regulation functions can also be expressed by the model. Heuristics for analysing the dynamics of Petri nets have been studied extensively in the past 3 decades, and include detection of active pathways, testing if a given system state is reachable and detecting state cycles[28]. Steggles *et al.* modelled the regulatory network of *Bacillus subtilis* sporulation using Petri nets and produced a behaviour that is in good agreement with existing literature[27]. For example, when initializing the system to a global state that corresponds to vegetative growth and activating the sporulation signal, the dynamics of the system lead to a state that corresponds to sporulation. This model also correctly predicted the sporulation capabilities of mutants.

Inference of particular network properties. In certain cases, incomplete information about a regulatory network can be used to infer topological features and regulatory interactions of the network. Due to the noisy nature of biological experiments, inference is usually based on a probabilistic framework that integrates experimental data in a network context. Here we briefly describe some static probabilistic models that infer properties of regulatory networks. These models do not describe in full the regulation of each entity under every possible condition, and do not describe dynamic processes (the concept of

Threshold function
A regulation function is a threshold function if it determines the state of the output entity by summing the states of its inputs and comparing the sum to some fixed value. For example, a gene upregulated if any two out of three transcription factors are active can be modelled by such a function.

Trajectory
In logical models, a trajectory is a sequence of global states that occur consecutively. In continuous models, a trajectory is the change of the level of an entity over time.

Markov chain
A stochastic process in which the next state depends only on the present state, regardless of the trajectory that led to the present state.

Heuristic
An algorithm for solving a problem that does not always provide an optimal solution to it. Heuristics are often used when it is impractical to obtain an exact optimal solution, and in many cases they provide satisfactory solutions.

Bayesian network
A probabilistic model that represents (in)dependencies between variables, taking the form of a directed acyclic graph. Often, both inference and learning can be carried out efficiently in such models. Dynamic Bayesian networks are an extension that describes dynamic behaviour.

Module
A set of genes that have identical regulation functions (and regulators). In other contexts, a module can also be a set of genes with a common function.

Inference
The selection of regulatory functions (or regulators) that best agrees with a dataset.

© 2008 Macmillan Publishers Limited. All rights reserved

trajectory is not defined for them), but provide higher level, lower resolution modelling and analysis (REF. 29 is an excellent source on probabilistic inference).

Module networks, introduced by Segal and colleagues, is a model that infers the regulation logic of gene modules given gene-expression data[30]. A regulation logic is represented by a decision tree, in which a path from the root to a leaf is determined by the up- or downregulation of regulatory modules, and a leaf determines the expression level of the corresponding genes. Module networks were tested with experimental data and correctly predicted some regulatory modules. Friedman *et al.* introduced Bayesian networks as a probabilistic tool for the identification of regulatory genes using high-throughput experimental data[29] and showed that they can reproduce certain known regulatory relationships[31,32]. Physical network models combine protein–DNA interactions, protein–protein interactions and knockout experiments for the discovery of regulatory interactions. The network structure of these models predicted knockout effects correctly[33]. Yeang and Vingron integrated perturbation data with knowledge from the literature into a joint model of regulation and metabolism and created a framework for the prediction of regulatory interactions and pathways[34]. They verified the predictive power of their model on the regulatory networks that govern the metabolism of glucose in *Escherichia coli* and found that the use of a joint model explains more perturbations than a regulatory network would explain alone. In all the probabilistic inference models above, predicted properties are assigned a certainty level. A cut-off for deciding which features will be selected for further analysis can then be determined. Examples for using cut-off criteria for network-feature selection can be found in REFS 7,32,35.

Continuous models

Biological experiments usually produce real, rather than discrete-valued, measurements. Examples include reaction rates, cell mass, cell-cycle length and gene-expression intensities[36–39]. Logical models require discretization of the real-valued data, which reduces the accuracy of the data. Continuous models, using real-valued parameters over a continuous timescale, allow a straightforward comparison of the global state and experimental data and can theoretically be more accurate. In practice, however, quantitative measurements are almost always partial (that is, they cover only a fraction of the system's entities). Therefore, some of the parameters of continuous models are usually based on estimations or inference. Below we describe some types of continuous models and the predictions that they can generate.

Continuous linear models. The defining property of linear models is that each regulator contributes to the input of the regulation function independently of the other regulators, in an additive manner. In other words, the change in the level of each entity depends on a weighted linear sum of the levels of its regulators. This assumption allows a high level of abstraction and efficient inference of network structure and regulation functions.

Discretization
A process that transforms continuous numerical values into discrete ones. For example, real-valued measurements can be discretized to 0, 1 or 2, corresponding to low, medium and high levels.

Time-series data usually contain many more genes than time points. This presents a difficulty in reverse engineering a network's structure and regulation functions. Yeung *et al.*[40] used a linear model and singular value decomposition[41] to generate a family of candidate networks that are consistent with a given dataset, thus compensating for this deficiency in time points. The network that is most consistent with prior knowledge is selected. The authors demonstrated in simulations that this approach is effective in dealing with shortages of data. Weaver *et al.*[42] described a model in which the expression of each gene is regulated by a 'squashing' function that takes as input a weighted linear sum of regulator levels, and presented an algorithm for reverse engineering real networks under these assumptions. One recent study adopted the linear framework to create a model of a regulatory network that is subjected to an arbitrary number of perturbations and studied multiple perturbation scenarios using simulated data and a single-perturbation scenario using experimental data[43]. Another study added time delays to regulatory interactions[44], which can be used to infer the duration of protein synthesis.

Linear models do not require extensive knowledge about regulatory mechanisms and can be used to obtain qualitative insights about regulatory networks, the simplest example being detection of novel regulations. However, when higher sensitivity to detail is desired, more complex models are preferable.

Models of transcription factor activity. The linear model is a crude description of the process of gene expression, and as such it cannot provide answers to subtle questions such as: how does the affinity of a transcription factor to a target promoter affect the network? Nachman *et al.*[45] created a fine-level model of gene regulation. In their model, entities correspond to either genes or transcription factors, and levels represent mRNA abundance or transcription factor activity, respectively. All the regulators are transcription factors. The levels of genes are determined by real-valued, non-linear regulation functions that take the Michaelis–Menten form[46]. The level of a gene is thus determined by that function together with the mRNA-decay rates. The time-dependent transcription factor activities are inferred from microarray time-series data using dynamic Bayesian networks[47,48]. An efficient heuristic aims to discover new regulators and regulatory relationships. Given an established regulatory network of 141 yeast cell-cycle genes, the heuristic successfully predicted the activity levels of the 7 regulators that controlled this network. In addition, it proposed novel regulatory relationships that improved the explanatory power of the model. Moreover, when given the entities, but not the network structure, as input, this method identified the seven regulators.

Shamir and Tanay developed a different model for identifying transcription factor–gene regulations[49]. The method relies on an efficient algorithm that infers transcription factor activity under the assumption that it is a monotone increasing function of both the transcription factor–promoter affinity and the transcription factor dosage. Transcription factor–promoter affinities are

© 2008 Macmillan Publishers Limited. All rights reserved

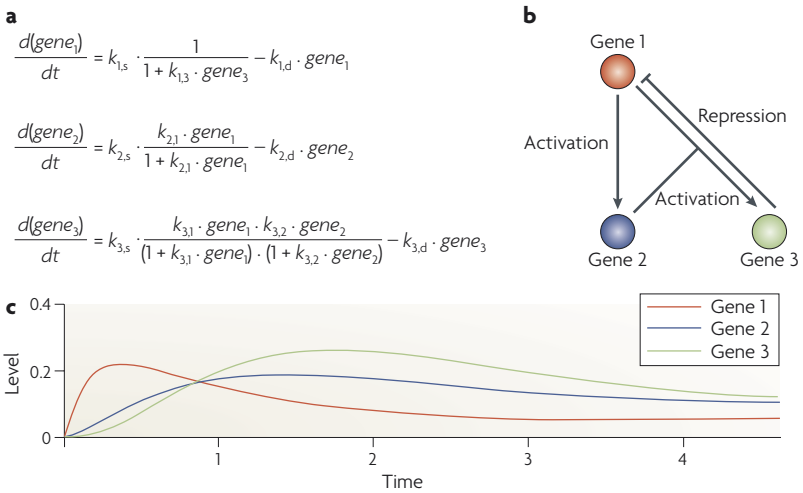


Figure 2 | **Ordinary differential equation model. a** | A network of three genes is modelled using ordinary differential equations (ODEs). Reaction rate constants are denoted by 'k'. **b** | The regulatory relations are depicted graphically. **c** | The trajectories of the model. Each equation shows the change in the level of a gene as a difference of its synthesis and degradation. Gene 1 is constitutively expressed, and is repressed by gene 3. Therefore, its level may reach a maximal rate of increase ($k_{1,s}$; 's' stands for synthesis) when the level of gene 3 is 0, in which case $k_{1,s}$ will be multiplied by 1. When the level of gene 3 is non-zero, the level of gene 1 rises slower than $k_{1,s}$. Transcription of gene 2 is activated by gene 1. This is expressed in the second equation of panel **a**, in which gene 2 level rises as a Michaelis–Menten function of the level of gene 1. Similarly, transcription of gene 3 is activated when both gene 1 and gene 2 levels are non-zero, and this relationship is given in the third equation of panel **a**. Degradation is modelled as a first-order reaction with rate constants $k_{i,d}$ (in which 'i' can be 1, 2 or 3). This formulation assumes that every transcript is immediately translated, and therefore the synthesis constants $k_{i,s}$ refer to both transcription and translation. According to simulation (bottom), the system stabilizes in a steady state at about 4.5 time units. The values of the rates in the simulations were: $k_{1,s}=k_{2,s}=2$; $k_{3,s}=15$; $k_{1,d}=k_{2,d}=k_{3,d}=1$; $k_{2,1}=k_{3,1}=k_{3,2}=1$; and $k_{1,3}=100$. The initial levels were all zero. Equations were solved using DESSolver v1.7 and the fourth order Runge–Kutta method.

inferred based on analysis of the promoters of the regulated genes. The model was applied to 140 genes of the galactose system in yeast, and inferred transcription factor activities that were in accordance with the literature. Two putative novel transcription factors, along with their genomic binding sites, were suggested. This demonstrates that the integration of multiple datasets can yield additional predictions that would be difficult to obtain from either dataset alone. The increased prediction power is obtained by the algorithm's ability to link different, but related, biological phenomena, in this case *cis*-regulatory elements and mRNA abundance.

Recently, Pan *et al.* extended the model developed by Nachman and colleagues by integrating genome sequence data[50]. Although these models offer a detailed description of regulation and provide inference algorithms, they do not directly incorporate interactions between regulatory entities. A similar methodology that uses discrete global states was suggested for inferring transcription factor activities based on the complete regulatory structure[51]. The model of Segal *et al.*[52] reproduced expression patterns that are generated by maternal and zygotic factors in the early *Drosophila melanogaster* embryo and provided interesting insights about the regulatory interactions of this system.

Michaelis–Menten functions
Equations that describe the kinetics of an enzymatic reaction. They can be derived from ordinary differential equations that describe the concentration changes of the involved molecular species under some simplifying assumptions.

Ordinary differential equations. A more general, detailed model of regulation can be described by ordinary differential equations (ODEs) (FIG. 2). These equations describe the (instantaneous) change in each entity as a function of the levels of some network entities. For simple ODE systems, an analytical solution can be formulated and the resulting set of algebraic equations then describes the change in entity levels over time. (REF. 46 provides a good overview for the use of ODEs in biological context and gives some illustrative examples). The Hill and Michaelis–Menten functions are examples of such analytical solutions of small systems. Larger networks, which often use these functions in addition to linear and bilinear functions, practically always require a numerical solution.

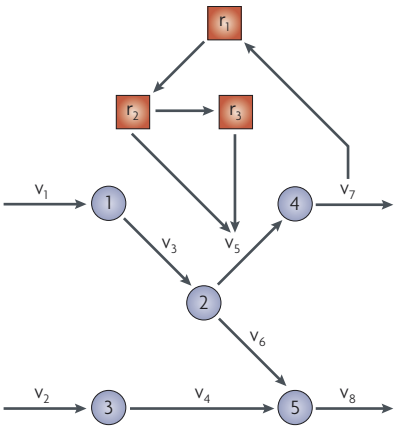
The ODE approach provides detailed information about the network's dynamics, but requires high-quality data on kinetic parameters and it is therefore currently applicable to only a few systems. The idea of using ODEs for modelling regulatory networks was suggested several decades ago[53]. Here we give some recent examples for modelling the dynamics of regulatory networks using ODEs.

Li *et al.* used ODEs to evaluate their model for the cell-cycle regulation in *Caulobacter crescentus*[54]. This bacterium divides asymmetrically into two morphologically distinctive cells, one of which, the stalked cell, is identical in form to the parent[55–58]. Their implementation follows the network dynamics from the parent cell to the stalked daughter cell. Entities correspond to protein concentrations, to the constriction ring at the mid-cell plane, to the process of DNA synthesis and to gene promoters. The system contains 16 equations (one for each variable), and these make use of 44 constants that were initially retrieved from the literature and then adjusted by trial and error. In tests in wild-type and 16 mutant strains, the model's simulations agreed with experimental measurements.

Chen *et al.* used the same approach to model the cell-cycle regulatory network in yeast[59]. In their model, entity levels corresponded to protein concentrations, cell mass, DNA mass, the state of the mitotic spindle and the state of the emerging bud from which the daughter cell was formed. The change in cell mass is assumed to depend only on the current cell mass. Therefore, the mass at division time is determined by the duration of the cell cycle. In total, 36 equations and 148 constants were used. After manual fitting, the model generated trajectories that reasonably matched the parent and daughter cell-cycle durations, the lengths of the G_1, G_2, S and M phases, and some of the experimentally determined ratios between groups of regulatory proteins. Moreover, 120 out of 131 simulated mutant strains had properties that were consistent with experimentally observed properties, including viability, growth rate, size at birth and size at budding.

Thus, ODE models can generate predictions that may subsequently be compared to cellular phenotypes. Additional examples for modelling with ODEs include the *Arabidopsis thaliana* circadian system[60] and osmoregulation in yeast[61]. More restricted types of ODE have also been proposed for modelling regulatory networks[62,63]. These are usually more abstract, require less detail during the modelling process and can be subjected to more powerful analysis.

© 2008 Macmillan Publishers Limited. All rights reserved



Trajectory

r_1	r_2	r_3	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
1	0	1	0.1	0.2	0.1	0.2	0	0.1	0	0.3
0	0	1	0.1	0.2	0.1	0.2	0	0.1	0	0.3
0	1	1	0.2	0.2	0.2	0.2	0.2	0	0.2	0.2
1	1	0	0.2	0.2	0.2	0.2	0.2	0	0.2	0.2
1	0	0	0.2	0.2	0.2	0.2	0.2	0	0.2	0.2
1	0	1	0.1	0.2	0.1	0.2	0	0.1	0	0.3

Regulation functions

$f_{r1}(V_7)$

v_7	r_1
= 0	1
≥ 1	0

$f_{r2}(r_1)$

r_1	r_2
0	1
1	0

$f_{r3}(r_2)$

r_2	r_3
0	1
1	0

$f_{v5}(r_2,r_3)$

r_2	r_3	v_5
0	0	≥ 0
0	1	= 0
1	0	≥ 0
1	1	≥ 0

Stoichiometric matix

1	0	–1	0	0	0	0	0
0	0	1	0	–1	–1	0	0
0	1	0	–1	0	0	0	0
0	0	0	0	1	0	–1	0
0	0	0	1	0	1	0	–1

Constraints

$0 \le v_1, v_2 \le 0.2$
$0 \le v_3, v_4, v_5, v_6 \le 0.4$
$0 \le v_7, v_8 \le 0.3$

Objective function

$v_7 + v_8$

Figure 3 | **Regulated flux balance analysis model.** The model shown contains three regulatory genes (squares) that regulate a metabolic layer. Metabolites are represented by circles, and metabolic fluxes by arrows that connect metabolites. Fluxes are denoted as v_1–v_8. The objective function that must be maximized is v_7+v_8. The metabolic flux v_7 regulates r_1. If it is non-zero, r_1 becomes active. Otherwise, r_1 becomes inactive. The regulators r_2 and r_3 regulate the flux v_5. When r_2 is not active and r_3 is active, v_5 is set to zero. Otherwise v_5 is not constrained. The regulation functions are shown. When v_5 is not constrained, a maximal value of v_7+v_8 is obtained by fluxes of magnitude 0.2 in all reactions, except v_6, the value of which remains 0. This is one of several possible solutions for the linear programming problem (they are referred to together as the solution space). When v_5 is constrained to 0 by the regulatory layer, v_7 must also become 0, and, hence, v_8 becomes the only outgoing flux. The trajectory cycles through five global states. The stoichiometric matrix describes the metabolites that each reaction consumes and produces. The columns correspond to reactions, and the rows to metabolites. For example, the third column means that the third reaction consumes one molecule of metabolite 1 for each molecule of metabolite 2 that is produced.

Regulated flux balance analysis. The cell-cycle ODE models incorporate cell growth and division by considering the progression of regulatory processes. However, in reality, changes in cell mass depend on metabolic activity. A complete picture of cellular regulation must take into account metabolic reactions and their interplay with the regulatory layer. For example, in the *lac* operon, a regulatory protein, the lac repressor, is regulated by a metabolite, lactose[64]. Regulated flux balance analysis (rFBA)[65,66] is a modelling approach that aims to integrate regulation and metabolism. rFBA is an extension of FBA[67] (see below; for more information on FBA, see REFS 67,68).

A major problem in using ODEs for describing biochemical reactions is the scarcity of experimental data on rate constants. FBA addresses this problem by assuming that the network is in a steady state and therefore that the total concentration of each substance does not change. Under this assumption, a system of ODEs is transformed into a system of linear equations, and its rates can be obtained by solving a linear programming problem that optimizes a certain objective function, for example, cellular growth. Such optimization problems can be solved efficiently. Further constraints are added to narrow the solution space. For example, the rate constants are restricted according to the catalytic capacities of metabolic enzymes[69]. The method has been successfully used to model large metabolic networks covering the near-complete metabolism of several species[70–72].

rFBA extends FBA by adding a layer of Boolean regulatory entities. For example, transcription factors that can be active or inactive and that can regulate enzymes that catalyse metabolic reactions (FIG. 3). Hence, it models interactions of both logical and continuous entities. The reactions of FBA are subjected to Boolean regulation functions that can set the reaction rates to zero if the regulatory logic dictates inactivation. For example, the production rate of a metabolite can drop to zero if the enzyme that produces it is not transcribed. The entities of the regulatory layer may also regulate each other via Boolean functions, and can also depend on discretized levels of metabolic entities. This regulation can be associated with a time delay. For instance, a Boolean entity that corresponds to a transcription factor can switch from 0 to 1 after a delay due to transcription and translation times.

Covert and Palsson[74] used rFBA to model the regulation of the central metabolic network of *E. coli*, which includes 149 genes, 16 regulatory proteins, 73 enzymes, 45 transcriptional regulations and 113 biochemical reactions. Growth predictions agreed well with experimental measurements in 106 out of 116 combinations of mutant strain–growth medium (measurements included viability, metabolite concentrations, cell mass and gene-expression values). A more comprehensive model that accounts for 1,010 genes was later introduced by Covert and colleagues[75].

Solution space
The set of possible solutions to an optimization problem. In the context of flux balance analysis, the solution space corresponds to different combinations of fluxes that optimize the objective function and that satisfy the constraints.

© 2008 Macmillan Publishers Limited. All rights reserved

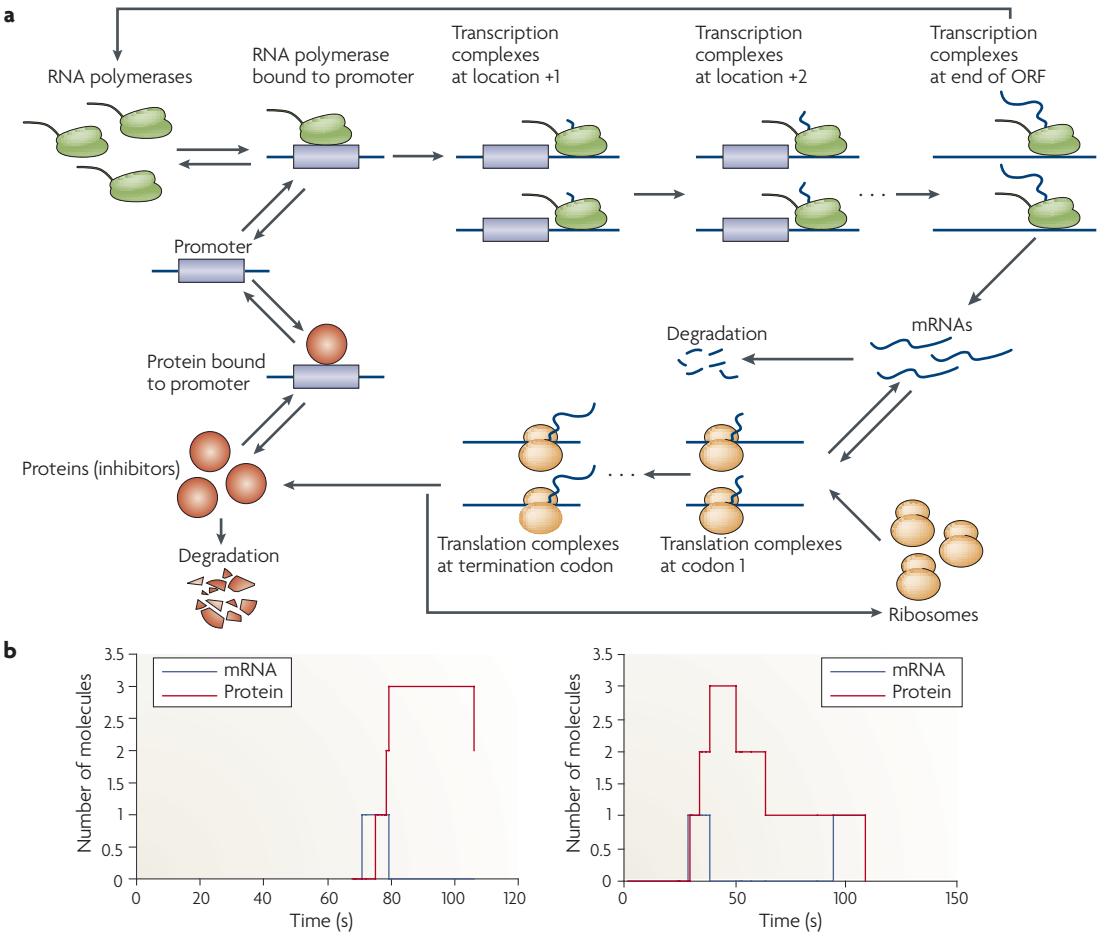


Figure 4 | **Single-molecule level model. a** | Stochastic model for a negative-feedback loop. The system contains a single gene, the product of which represses its own promoter. The diagram shows the different interactions between molecules, each represented by a different entity. For example, the transcription complex is represented by a distinct entity for every location of the transcription complex on the open reading frame (ORF). Arrows represent transformations of molecular species that occur during a reaction. The tails of the arrows point to the substrates and the arrowheads point to the products. For example, the dissociation of the complex RNA polymerase + promoter is represented by the two arrows pointing from the complex to RNA polymerase and to the naked promoter (top left). **b** | Two possible trajectories for the mRNA and protein entities of the model. In the first trajectory, a transcription event occurs, followed by a translation event. Next, several ribosomes initiate translation consecutively and produce two additional proteins (the model allows this as initiations of translation do not consume an mRNA molecule, as is depicted in panel **a**). At the same time, the only transcript degrades. The last event is protein degradation. In the second trajectory, a transcript is produced at an earlier time, and also degrades earlier. Three proteins are generated and then gradually degrade. At about 90 seconds, RNA polymerase manages to bind the promoter and produces a second transcript. Simulations performed using STOCKS 2.0 (REF. 138). The values of the rates in seconds^{-1} were: 100 for elongation of transcript; 30 for elongation of the polypeptide chain; 1 for termination of transcription and/or translation; 0.04 for transcript degradation; 0.025 for protein degradation; and 0.1 for all other reactions. Transcript size was 100, and polypeptide chain size was 30. Initial levels were 1 promoter. The initial number of RNA polymerase molecules is selected from the normal distribution N(35,3.5), and the initial number of ribosome molecules is selected from the normal distribution N(15,3.5), and 0 for all other entities.

Barrett and Palsson[76] created an algorithm that uses rFBA to design a series of experiments for reverse engineering a regulatory network. Before every lab experiment, the algorithm chooses a set of transcription factors that will be knocked out and two growth environments between which the cells will be shifted. The goal is to minimize the total number of experiments. Given probabilistic knowledge about regulatory interactions, the algorithm simulates cell growth for every possible combination of environments and knockout sets, and selects one under which the largest number of novel regulatory interactions are most likely to occur. The lab experiment that follows applies the suggested perturbations and environmental shift, generates an expression profile and verifies all the indicated regulatory interactions using chromatin immunoprecipitation (ChIP). Experimentally verified interactions are added to the model, and the process can be repeated. The algorithm's selections showed good agreement with the decisions of scientists in the reconstruction of an *E. coli* network. A similar methodology was proposed[77] and tested experimentally[33,78] for selecting experiments in Boolean network reconstruction.

© 2008 Macmillan Publishers Limited. All rights reserved

Box 1 | **Stochastic simulation of phage λ development**

Phage λ is a bacteriophage that infects *Escherichia coli* cells. A network of regulatory interactions between phage molecules determines if the phage selects the lysogenic pathway or the lytic one[137]. When a phage chooses the lytic pathway, the concentration of the Cro protein in the host is relatively high and the concentration of the CI protein is relatively low. If the lysogenic pathway is chosen, the opposite is true. McAdams and Arkin simulated the pathway-decision process by using an stochastic simulation algorithm (SSA) under several simplifying assumptions (for example, that the host's housekeeping molecules are present in constant concentrations)[104]. Their model defined 26 reaction types, 40 parameters and 18 molecular species (not including complexes). For example, elongation of a polypeptide chain is a single reaction with the same rate for all amino acids. They view the DNA as one species, although the position of RNA polymerase affects transcription rate, and consider the translation of any mRNA transcript by the ribosome as a single reaction type. The simulations showed that the trajectories of CI and Cro concentrations may vary substantially as a result of the intrinsic stochasticity of the system. Furthermore, the fraction of lysogens as a function of the average number of phages per host was in good accordance with experimental data.

This work demonstrated, for the first time, that a real regulatory network can generate profoundly different trajectories due to stochasticity. Subsequently, Weinberger *et al.*[103], on the basis of experiments and simulations, proposed that a positive-feedback loop created by the Tat protein and affected by stochasticity generates fluctuations in latency time. Schultz *et al.*[102] used SSA to explain the transition between vegetation and competence in *Bacillus subtilis*. Gonze and Golbeter[100] investigated the effects of noise on circadian clocks and the conditions that promote their robustness. More efficient methods are needed to carry out simulations of larger networks.

The rFBA approach offers a detailed description of the metabolic layer and also accounts for the interplay between regulation and metabolism. Although the modelling of the regulatory layer is qualitative and less detailed than in other continuous approaches, this is compensated for by the model's capability to infer metabolic fluxes. (For another example of rFBA, see REF. 79 for an analysis of the regulation of metabolism in yeast.) Shlomi *et al.* extended rFBA to study the regulation of metabolism in the steady state[80]. In their model a steady state is obtained by solving a mixed integer linear programming problem rather than by following a trajectory. A different constraint-based approach that allows analysis of the regulatory network in various environments was introduced by Gianchandani and colleagues[81].

Single-molecule level models

Every biological network is composed of stochastic components, and therefore it may manifest different behaviours, even starting from the same initial conditions[82,83]. When the number of involved molecules of each species is large, the law of mass action[46] can be used to accurately calculate the change in concentrations, and little or no stochastic effect is observable. However, when the number of molecules is small, significant stochastic effects may be seen (FIG. 4). This is particularly true for regulatory networks, in which the number of regulatory molecules is often low[84–87]. Recently, single-cell experimental assays demonstrated the stochastic behaviour of the processes of transcription[88–90] and translation[89,91,92]. Here we present models that incorporate the stochastic nature of regulation by accounting for the fluctuations that occur on the molecular level (reviewed in REF. 93).

Gillespie's stochastic simulation algorithm. McAdams and Arkin[94] showed that fluctuations in time intervals between biochemical reactions, and consequently in the occurrence times of regulatory events, can be expressed by a model that follows biochemical reactions at single-molecule resolution. The model is based on Gillespie's stochastic simulation algorithm (SSA)[95,96]. SSA takes as input the initial number of molecules of several species (for example, mRNAs and proteins) and reaction-probability constants, and simulates the dynamics of the system, reaction by reaction. A reaction probability is the probability that the necessary combination of specific molecules will participate in that reaction in an infinitesimal time interval. For example, consider the phosphorylation reaction:

$$\text{Kinase–phosphate} + \text{target} \xrightarrow{c_1} \text{kinase} + \text{target–phosphate}$$

The reaction probability $c_1 \cdot dt$ is the probability that a specific kinase molecule will phosphorylate a specific protein molecule in the infinitesimal time interval dt. Gillespie has shown how reaction probability constants can be derived from deterministic reaction rates.

The basic assumption of the algorithm is that the system is 'well stirred' — that is, that each molecule always has an equal chance of being anywhere in the system's volume. This assumption applies, for example, if most of the collisions between molecules are non-reactive. Although it overlooks some biological processes that affect regulation, such as diffusion[97] and transportation[98,99], the algorithm proved useful in describing the time evolution of several small regulatory networks and mechanisms[100–104]. BOX 1 provides an example of how SSA can be used to analyse a biological system.

Approximations to SSA. Although Gisbon and Bruck introduced a way to speed up SSA[105], SSA still requires extensive computational resources because it simulates every individual reaction. Consequently, SSA is not ideal for modelling large-scale networks. Therefore, researchers further modified SSA, sacrificing a certain level of detail for the sake of faster simulation.

τ-leaping is a variation of SSA that trades accuracy for efficiency[106]. Instead of generating every single reaction, τ-leaping 'leaps' over a time interval of size τ and randomly selects the number of reactions of each type that occurred in this interval. Gillespie suggested[106] a procedure for selecting τ that was later improved and implemented as part of a stochastic simulation toolkit[107] (REF. 97 describes in detail different SSA approximation methods). When some of the reactions can be described using ODEs, a more efficient strategy is to separate reactions into two regimes: discrete and continuous (see, for example, REFS 108, 109). The integration algorithm of E-Cell version 3 (see Supplementary information S1 (table)) combines multiple stand-alone algorithms (for example SSA and a numerical ODE solver[110]). The use of effective reactions, which amalgamate several simple reaction steps into a single complex step, is a method for abstraction and increasing simulation speed[111,112]. Reaction steps can also be eliminated by applying a steady state assumption[113]. Additional approximation methods are described in REF. 93.

© 2008 Macmillan Publishers Limited. All rights reserved

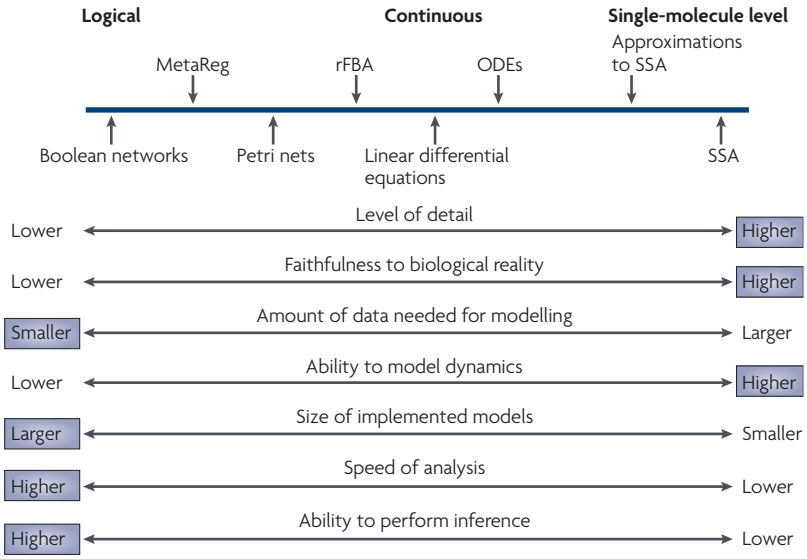


Figure 5 | **A schematic comparison of regulatory network models.** Models are listed along an imaginary scale, in which the level of detail of the models decreases, and the amount of detail increases, from left to right. Several pertinent criteria are indicated below the scale. Boolean networks are the purest form of logical models. They are highly abstract and hence require the least amount of data, but at the same time can display only qualitative dynamic behaviour. MetaReg is closer to biological reality because it can express intermediate regulator concentrations and accommodate probabilities, but requires more knowledge about the network and is limited to analysis of steady states. Petri nets can reveal finer detail to metabolic and signalling networks, and can therefore be used to describe integrated regulatory and metabolic/signalling networks and handle some dynamics. The analysis Petri nets offer is still qualitative. Regulated flux balance analysis (rFBA) produces metabolic predictions that can be compared to experimental measurements, but requires biochemical knowledge and is more challenging to analyse. Linear differential equations can model and predict experimentally observed concentrations of regulatory entities, and possess more detailed dynamics than the former models. General ordinary differential equations (ODEs) are more consistent with biochemical mechanisms than linear ODEs, but are harder to analyse. Single-molecule level models, the most detailed, can capture stochasticity, but are computationally expensive. To deal with this computational burden, approximations to stochastic simulation algorithms (SSAs) were developed, which sacrifice some detail for better performance. Methods that infer particular properties (not shown) can fall anywhere on the left half of the scale, depending on the properties of the chosen model.

Summary

The introduction of novel and powerful experimental methods for studying gene regulation has created an upsurge of interest in modelling regulatory networks. In this article, three approaches to modelling were highlighted and some representative examples were discussed. We also discussed key differences among these approaches and rules of thumb for selecting an appropriate model (FIG. 5). Available modelling tools from each approach, as well as relevant databases, are listed in Supplementary information S1 (table).

A model's quality can be assessed by how similar its predictions are to experimental data. If two models generate predictions that match the same data equally well, then the simpler model is preferable, because it can be better understood and is less prone to over-fitting. When available observations are qualitative in nature, logical models can be accurate and have the advantage of having a modest number of global states. This enables more intuitive and efficient analysis methods. When data include real-valued measurements, such as time[74] or space[55,114], real-valued predictions can be more accurate. In addition, the simplified dynamics of logical models are often less appropriate for the complex behaviours that generate such measurements, and this motivates the use of continuous models or models that combine logical and continuous approaches.

The stochastic nature of gene expression influences the dynamics of regulatory networks, and this aspect is usually not modelled by continuous approaches[115]. Single-molecule level models are the most detailed and can explain stochastic behaviour in several scenarios. While accounting for the full complexity of gene regulation, single-molecule level models are also the hardest to study analytically, and stochastic experimental data are currently very scarce.

Limited availability of reaction rate constants and incomplete understanding of gene regulation are major impediments for building accurate models. In this respect, lower model resolution is an advantage, as it requires fewer parameters and less detailed understanding of the regulatory mechanisms[116–123]. Analytical methods that cope with these problems were developed for logical and continuous models, and some of these were presented above. As a brute force alternative, the space of potential parameters can be scanned for certain dynamic behaviours, provided that the model is both computationally simple and has a sufficiently small number of global states. For example, one study[103] searches the parameter space of a continuous model and derives molecular level parameters from results. Another problem associated with building accurate models is that experimental data are usually derived from a population of cells that needs to be synchronized[124]. The mean behaviour of a population (for example, as measured by gene expression) does not always exhibit fluctuations that can be observed at a single cell level. In such cases, the accuracy of deterministic and stochastic approaches is equally limited.

Despite substantial progress in modelling regulatory networks over the past decade, nature's design of regulatory networks confronts us with many open questions. Although it is clear that structure alone does not determine network dynamics[125,126], the role of different network architectures in generating dynamic behaviours[127,128], and the evolutionary processes that produced them, are far from understood. And, what is the effect of noise on regulatory networks? General strategies for overcoming stochastic effects are known[82], but a large-scale quantitative study has not yet been performed. Stochastic effects can also give rise to evolutionary advantages in a population by creating diversity[129,130]. Characterization of the beneficial role of stochasticity remains a future challenge. Notably, stochastic effects have been extensively studied in other types of dynamic biological systems, including population genetics and theoretical ecology[131–134].

Our current picture of how regulation is carried out is probably still missing several significant pieces. More experimental work is needed, and we must incorporate results into improved network models. Experimental design approaches[76–78] will help us to select the most efficient

© 2008 Macmillan Publishers Limited. All rights reserved

set of experiments. In addition to understanding regulation as a stand-alone process, models for the interplay of regulation with other processes, for example metabolism and cell–cell signalling, need to be created[135,136]. The benefits of accurate, large-scale regulatory network models for medicine and biotechnology provide a strong incentive for cooperation between experimentalists and computational scientists.

1. Weindruch, R. & Walford, R. L. *The Retardation Of Aging And Disease By Dietary Restriction* (Thomas, Springfield, 1988).
2. Glass, L. & Kauffman, S. A. The logical analysis of continuous, non-linear biochemical control networks. *J. Theor. Biol.* **39**, 103–129 (1973).
3. Thomas, R. Boolean formalization of genetic control circuits. *J. Theor. Biol.* **42**, 563–585 (1973).
4. Davidson, E. H. *et al.* A genomic regulatory network for development. *Science* **295**, 1669–1678 (2002).
A highly detailed reconstruction of the regulatory network that controls the first 24 hours of sea urchin embryo development.
5. Smith, J., Theodoris, C. & Davidson, E. H. A gene regulatory network subcircuit drives a dynamic pattern of gene expression. *Science* **318**, 794–797 (2007).
6. Kauffman, S. A. *The Origins Of Order: Self-Organization And Selection In Evolution* (Oxford University Press, Oxford, 1993).
An accessible description of Kauffman's work, including the Boolean network model and its analysis.
7. Kauffman, S., Peterson, C., Samuelsson, B. & Troein, C. Random Boolean network models and the yeast transcriptional network. *Proc. Natl Acad. Sci. USA* **100**, 14796–14799 (2003).
8. Li, F., Long, T., Lu, Y., Ouyang, Q. & Tang, C. The yeast cell-cycle network is robustly designed. *Proc. Natl Acad. Sci. USA* **101**, 4781–4786 (2004).
9. Lähdesmäki, H., Shmulevich, I. & Yli-Harja, O. On learning gene regulatory networks under the Boolean network model. *Machine Learning* **52**, 147–167 (2003).
10. Akutsu, T., Miyano, S. & Kuhara, S. Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. *Pac Symp. Biocomput.*, 17–28 (1999).
11. Spellman, P. T. *et al.* Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell* **9**, 3273–3297 (1998).
12. Shmulevich, I., Dougherty, E. R., Kim, S. & Zhang, W. Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics* **18**, 261–274 (2002).
13. Shmulevich, I., Gluhovsky, I., Hashimoto, R. F., Dougherty, E. R. & Zhan, W. Steady-state analysis of genetic regulatory networks modelled by probabilistic Boolean networks. *Comp. Funct. Genomics* **4**, 601–608 (2003).
14. Bhattacharya, R. N. & Majumdar, M. *Random Dynamical Systems: Theory And Applications* (Cambridge University Press, Cambridge, 2007).
15. Gilks, W. R., Richardson, S. & Spiegelhalter, D. J. *Markov Chain Monte Carlo In Practice* (Chapman & Hall, Boca Raton, 1998).
16. Gat-Viks, I., Tanay, A. & Shamir, R. Modeling and analysis of heterogeneous regulation in biological networks. *J. Comput. Biol.* **11**, 1034–1049 (2004).
17. Gat-Viks, I., Tanay, A., Raijman, D. & Shamir, R. A probabilistic methodology for integrating knowledge and experiments on biological networks. *J. Comput. Biol.* **13**, 165–181 (2006).
An exposition of the MetaReg methodology, including the probabilistic layer.
18. Kschischang, F. R., Frey, B. J. & Loeliger, H. A. Factor graphs and the sum-product algorithm. *IEEE Trans. Info. Theory* **47**, 498–519 (2001).
19. MacKay, D. J. C. *Introduction To Monte Carlo Methods In Learning In Graphical Models* (ed. Jordan, M. I.) (Kluwer Academic Press, New York, 1998).
20. Gat-Viks, I. & Shamir, R. Refinement and expansion of signaling pathways: the osmotic response network in yeast. *Genome Res.* **17**, 358–367 (2007).
21. Petri, C. A. Kommunikation mit Automaten. *Schriften des Instituts für Instrumentelle Mathematik* (1962).
22. Koch, I., Schueler, M. & Heiner, M. STEPP — search tool for exploration of Petri net paths: a new tool for Petri net-based path analysis in biochemical networks. *In Silico Biol.* **5**, 129–137 (2005).
23. Reddy, V. N., Liebman, M. N. & Mavrovouniotis, M. L. Qualitative analysis of biochemical reaction systems. *Comput. Biol. Med.* **26**, 9–24 (1996).
24. Kuffner, R., Zimmer, R. & Lengauer, T. Pathway analysis in metabolic databases via differential metabolic display (DMD). *Bioinformatics* **16**, 825–836 (2000).
25. Chaouiya, C., Remy, E., Ruet, P. & Thieffry, D. in *Proceedings of the 25th International Conference on Applications and Theory of Petri Nets* (eds Cortadella, J. & Reisig, W.) (Springer, Berlin, 2004).
26. Simao, E., Remy, E., Thieffry, D. & Chaouiya, C. Qualitative modelling of regulated metabolic pathways: application to the tryptophan biosynthesis in *E. coli*. *Bioinformatics* **21**, 190–196 (2005).
27. Steggles, L. J., Banks, R., Shaw, O. & Wipat, A. Qualitatively modelling and analysing genetic regulatory networks: a Petri net approach. *Bioinformatics* **23**, 336–343 (2007).
28. Peterson, J. *Petri Net Theory and the Modeling of Systems* (Prentice Hall PTR, New Jersey, 1981).
29. Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Francisco, 1988).
An excellent source on the foundation of probabilistic inference.
30. Segal, E. *et al.* Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genet.* **34**, 166–176 (2003).
31. Friedman, N. Inferring cellular networks using probabilistic graphical models. *Science* **303**, 799–805 (2004).
32. Friedman, N., Linial, M., Nachman, I. & Pe'er, D. Using Bayesian networks to analyze expression data. *J. Comput. Biol.* **7**, 601–620 (2000).
A pioneering work that describes the use of Bayesian networks for recovering regulatory interactions from experimental data.
33. Yeang, C. H., Ideker, T. & Jaakkola, T. Physical network models. *J. Comput. Biol.* **11**, 243–262 (2004).
34. Yeang, C. H. & Vingron, M. A joint model of regulatory and metabolic networks. *BMC Bioinformatics* **7**, 332 (2006).
35. Lee, T. I. *et al.* Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science* **298**, 799–804 (2002).
36. Sauer, U. *et al.* Physiology and metabolic fluxes of wild-type and riboflavin-producing *Bacillus subtilis*. *Appl. Environ. Microbiol.* **62**, 3687–3696 (1996).
37. Ness, S. A. Basic microarray analysis: strategies for successful experiments. *Methods Mol. Biol.* **316**, 13–33 (2006).
38. Kingsmore, S. F. Multiplexed protein measurement: technologies and applications of protein and antibody arrays. *Nature Rev. Drug Discov.* **5**, 310–320 (2006).
39. Hellerstein, M. K. *In vivo* measurement of fluxes through metabolic pathways: the missing link in functional genomics and pharmaceutical research. *Annu. Rev. Nutr.* **23**, 379–402 (2003).
40. Yeung, M. K., Tegner, J. & Collins, J. J. Reverse engineering gene networks using singular value decomposition and robust regression. *Proc. Natl Acad. Sci. USA* **99**, 6163–6168 (2002).
41. Golub, G. H. & Van Loan, C. F. *Matrix computations* (Johns Hopkins University Press, Maryland, 1996).
42. Weaver, D. C., Workman, C. T. & Stormo, G. D. Modeling regulatory networks with weight matrices. *Pac Symp. Biocomput.*, 112–123 (1999).
43. Bansal, M., Gatta, G. D. & di Bernardo, D. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics* **22**, 815–822 (2006).
44. Dasika, M. S., Gupta, A. & Maranas, C. D. A mixed integer linear programming (MILP) framework for inferring time delay in gene regulatory networks. *Pac Symp. Biocomput*, 474–485 (2004).
45. Nachman, I., Regev, A. & Friedman, N. Inferring quantitative models of regulatory networks from expression data. *Bioinformatics* **20**, 248–256 (2004).
46. Klipp, E. *Systems Biology In Practice: Concepts, Implementation And Application* (Wiley-VCH, Weinheim, 2005).
Explains basic concepts in modelling biological networks and provides an excellent introduction for the use of ODEs.
47. Friedman, N., Murphy, K. & Russell, S. in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (eds Cooper, G. F. & Moral, S.) 129–138 (Morgan Kaufmann, San Francisco, 1998).
48. Kim, S. Y., Imoto, S. & Miyano, S. Inferring gene networks from time series microarray data using dynamic Bayesian networks. *Brief Bioinform.* **4**, 228–235 (2003).
49. Shamir, R. & Tanay, A. in *RECOMB'03* (ACM, Berlin, 2003).
50. Pan, Y., Durfee, T., Bockhorst, J. & Craven, M. Connecting quantitative regulatory-network models to the genome. *Bioinformatics* **23**, 367–376 (2007).
51. Li, Z., Shaw, S. M., Yedwabnick, M. J. & Chan, C. Using a state-space model with hidden variables to infer transcription factor activities. *Bioinformatics* **22**, 747–754 (2006).
52. Segal, E., Raveh-Sadka, T., Schroeder, M., Unnerstall, U. & Gaul, U. Predicting expression patterns from regulatory sequence in *Drosophila* segmentation. *Nature* **451**, 535–540 (2008).
53. Goodwin, B. C. *Temporal Organization In Cells; A Dynamic Theory Of Cellular Control Processes* (Academic Press, New York, 1963).
54. Li, S., Brazhnik, P., Sobral, B. & Tyson, J. J. A quantitative study of the division cycle of *Caulobacter crescentus* stalked cells. *PLoS Comput. Biol.* **4**, e9 (2008).
55. McAdams, H. H. & Shapiro, L. A bacterial cell-cycle regulatory network operating in time and space. *Science* **301**, 1874–1877 (2003).
56. Laub, M. T., McAdams, H. H., Feldblyum, T., Fraser, C. M. & Shapiro, L. Global analysis of the genetic network controlling a bacterial cell cycle. *Science* **290**, 2144–2148 (2000).
57. Holtzendorff, J. *et al.* Oscillating global regulators control the genetic circuit driving a bacterial cell cycle. *Science* **304**, 983–987 (2004).
58. Biondi, E. G. *et al.* Regulation of the bacterial cell cycle by an integrated genetic circuit. *Nature* **444**, 899–904 (2006).
59. Chen, K. C. *et al.* Integrative analysis of cell cycle control in budding yeast. *Mol. Biol. Cell* **15**, 3841–3862 (2004).
An ODE model for the yeast cell-cycle regulatory network that is supported by phenotypes of more than 100 mutant strains.
60. Locke, J. C. *et al.* Extension of a genetic network model by iterative experimentation and mathematical analysis. *Mol. Syst. Biol.* **1**, 0013 (2005).
61. Klipp, E., Nordlander, B., Kruger, R., Gennemark, P. & Hohmann, S. Integrative model of the response of yeast to osmotic shock. *Nature Biotechnol.* **23**, 975–982 (2005).
62. Gebert, J., Radde, N. & Weber, G. Modeling gene regulatory networks with piecewise linear differential equations. *Eur. J. Oper. Res.* **181**, 1148 (2007).
63. Ropers, D., de Jong, H., Page, M., Schneider, D. & Geiselmann, J. Qualitative simulation of the carbon starvation response in *Escherichia coli*. *Biosystems* **84**, 124–152 (2006).
64. Mèuller-Hill, B. *The lac Operon: A Short History Of A Genetic Paradigm* (Walter de Gruyter, Berlin, 1996).
65. Covert, M. W., Schilling, C. H. & Palsson, B. Regulation of gene expression in flux balance models of metabolism. *J. Theor. Biol.* **213**, 73–88 (2001).
66. Kauffman, K. J., Prakash, P. & Edwards, J. S. Advances in flux balance analysis. *Curr. Opin. Biotechnol.* **14**, 491–496 (2003).
67. Palsson, B. Ø. *Systems Biology: Properties Of Reconstructed Networks* (Cambridge University Press, New York, 2006).
An excellent book about the study of biological networks by one of the leading researchers in this field.
68. Edwards, J. S., Covert, M. & Palsson, B. Metabolic modelling of microbes: the flux-balance approach. *Environ. Microbiol* **4**, 133–140 (2002).

© 2008 Macmillan Publishers Limited. All rights reserved

69. Beg, Q. K. *et al.* Intracellular crowding defines the mode and sequence of substrate uptake by *Escherichia coli* and constrains its metabolic activity. *Proc. Natl Acad. Sci. USA* **104**, 12663–12668 (2007).
70. Duarte, N. C., Herrgard, M. J. & Palsson, B. O. Reconstruction and validation of *Saccharomyces cerevisiae* iND750, a fully compartmentalized genome-scale metabolic model. *Genome Res.* **14**, 1298–1309 (2004).
71. Resendis-Antonio, O., Reed, J. L., Encarnacion, S., Collado-Vides, J. & Palsson, B. O. Metabolic reconstruction and modeling of nitrogen fixation in *Rhizobium etli*. *PLoS Comput. Biol.* **3**, 1887–1895 (2007).
72. Feist, A. M. *et al.* A genome-scale metabolic reconstruction for *Escherichia coli* K-12 MG1655 that accounts for 1260 ORFs and thermodynamic information. *Mol. Syst. Biol.* **3**, 121 (2007).
73. Papin, J. A., Price, N. D., Edwards, J. S. & Palsson, B. B. The genome-scale metabolic extreme pathway structure in *Haemophilus influenzae* shows significant network redundancy. *J. Theor. Biol.* **215**, 67–82 (2002).
74. Covert, M. W. & Palsson, B. O. Transcriptional regulation in constraints-based metabolic models of *Escherichia coli*. *J. Biol. Chem.* **277**, 28058–28064 (2002).
75. Covert, M. W., Knight, E. M., Reed, J. L., Herrgard, M. J. & Palsson, B. O. Integrating high-throughput and computational data elucidates bacterial networks. *Nature* **429**, 92–96 (2004).
A large-scale model of *E. coli* metabolic regulation that uses quantitative metabolic flux values.
76. Barrett, C. L. & Palsson, B. O. Iterative reconstruction of transcriptional regulatory networks: an algorithmic approach. *PLoS Comput. Biol.* **2**, e52 (2006).
77. Ideker, T. E., Thorsson, V. & Karp, R. M. Discovery of regulatory interactions through perturbation: inference and experimental design. *Pac Symp. Biocomput*, 305–316 (2000).
78. Yeang, C. H. *et al.* Validation and refinement of gene-regulatory pathways on a network of physical interactions. *Genome Biol.* **6**, R62 (2005).
79. Herrgard, M. J., Lee, B. S., Portnoy, V. & Palsson, B. O. Integrated analysis of regulatory and metabolic networks reveals novel regulatory mechanisms in *Saccharomyces cerevisiae*. *Genome Res.* **16**, 627–635 (2006).
80. Shlomi, T., Eisenberg, Y., Sharan, R. & Ruppin, E. A genome-scale computational study of the interplay between transcriptional regulation and metabolism. *Mol. Syst. Biol.* **3**, 101 (2007).
81. Gianchandani, E. P., Papin, J. A., Price, N. D., Joyce, A. R. & Palsson, B. O. Matrix formalism to describe functional states of transcriptional regulatory systems. *PLoS Comput. Biol.* **2**, e101 (2006).
82. McAdams, H. H. & Arkin, A. It's a noisy business! Genetic regulation at the nanomolar scale. *Trends Genet.* **15**, 65–69 (1999).
83. Ross, I. L., Browne, C. M. & Hume, D. A. Transcription of individual genes in eukaryotic cells occurs randomly and infrequently. *Immunol. Cell Biol.* **72**, 177–185 (1994).
84. Bae, K., Lee, C., Hardin, P. E. & Edery, I. dCLOCK is present in limiting amounts and likely mediates daily interactions between the dCLOCK–CYC transcription factor and the PER–TIM complex. *J. Neurosci.* **20**, 1746–1753 (2000).
85. Guptasarma, P. Does replication-induced transcription regulate synthesis of the myriad low copy number proteins of *Escherichia coli*? *Bioessays* **17**, 987–997 (1995).
86. Bailone, A., Levine, A. & Devoret, R. Inactivation of prophage λ repressor *in vivo*. *J. Mol. Biol.* **131**, 553–572 (1979).
87. Shea, M. A. & Ackers, G. K. The OR control system of bacteriophage λ. A physical-chemical model for gene regulation. *J. Mol. Biol.* **181**, 211–230 (1985).
88. Golding, I., Paulsson, J., Zawilski, S. M. & Cox, E. C. Real-time kinetics of gene activity in individual bacteria. *Cell* **123**, 1025–1036 (2005).
89. Ozbudak, E. M., Thattai, M., Kurtser, I., Grossman, A. D. & van Oudenaarden, A. Regulation of noise in the expression of a single gene. *Nature Genet.* **31**, 69–73 (2002).
90. Raj, A., Peskin, C. S., Tranchina, D., Vargas, D. Y. & Tyagi, S. Stochastic mRNA synthesis in mammalian cells. *PLoS Biol.* **4**, e309 (2006).
91. Cai, L., Friedman, N. & Xie, X. S. Stochastic protein expression in individual cells at the single molecule level. *Nature* **440**, 358–362 (2006).
92. Yu, J., Xiao, J., Ren, X., Lao, K. & Xie, X. S. Probing gene expression in live cells, one protein molecule at a time. *Science* **311**, 1600–1603 (2006).
93. Gillespie, D. T. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.* **58**, 35–55 (2007).
94. McAdams, H. H. & Arkin, A. Stochastic mechanisms in gene expression. *Proc. Natl Acad. Sci. USA* **94**, 814–819 (1997).
McAdams and Arkin describe their modelling methodology, which uses Gillespie's SSA, and present simulation results of a simple system that exhibits protein bursts.
95. Gillespie, D. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comp. Phys.* **22**, 403–434 (1976).
Gillespie presents his SSA, the understanding of which is of major importance for computational modelling.
96. Gillespie, D. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.* **81**, 2340–2361 (1977).
97. Halford, S. E. & Marko, J. F. How do site-specific DNA-binding proteins find their targets? *Nucleic Acids Res.* **32**, 3040–3052 (2004).
98. Terry, L. J., Shows, E. B. & Wente, S. R. Crossing the nuclear envelope: hierarchical regulation of nucleocytoplasmic transport. *Science* **318**, 1412–1416 (2007).
99. Rapoport, T. A., Jungnickel, B. & Kutay, U. Protein transport across the eukaryotic endoplasmic reticulum and bacterial inner membranes. *Annu. Rev. Biochem.* **65**, 271–303 (1996).
100. Gonze, D. & Goldbeter, A. Circadian rhythms and molecular noise. *Chaos* **16**, 026110 (2006).
101. Niemitalo, O. *et al.* Modelling of translation of human protein disulfide isomerase in *Escherichia coli* — a case study of gene optimisation. *J. Biotechnol.* **120**, 11–24 (2005).
102. Schultz, D., Ben Jacob, E., Onuchic, J. N. & Wolynes, P. G. Molecular level stochastic model for competence cycles in *Bacillus subtilis*. *Proc. Natl Acad. Sci. USA* **104**, 17582–17587 (2007).
103. Weinberger, L. S., Burnett, J. C., Toettcher, J. E., Arkin, A. P. & Schaffer, D. V. Stochastic gene expression in a lentiviral positive-feedback loop: HIV-1 Tat fluctuations drive phenotypic diversity. *Cell* **122**, 169–182 (2005).
104. Arkin, A., Ross, J. & McAdams, H. H. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected *Escherichia coli* cells. *Genetics* **149**, 1633–1648 (1998).
A seminal work that quantitatively explains how phage λ makes a stochastic decision between lysis and lysogeny using regulatory network simulation.
105. Gibson, M. & Bruck, J. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem.* **104**, 1876–1889 (1999).
106. Gillespie, D. Approximate accelerated stochastic simulation of chemically reacting systems. *J. Chem. Phys.* **115**, 1716–1733 (2001).
107. Li, H., Cao, Y., Petzold, L. R. & Gillespie, D. T. Algorithms and software for stochastic simulation of biochemical reacting systems. *Biotechnol. Prog.* **24**, 56–61 (2008).
108. Kiehl, T. R., Mattheyses, R. M. & Simmons, M. K. Hybrid simulation of cellular behavior. *Bioinformatics* **20**, 316–322 (2004).
109. Griffith, M., Courtney, T., Peccoud, J. & Sanders, W. H. Dynamic partitioning for hybrid simulation of the bistable HIV-1 transactivation network. *Bioinformatics* **22**, 2782–2789 (2006).
110. Takahashi, K., Kaizu, K., Hu, B. & Tomita, M. A multi-algorithm, multi-timescale method for cell simulation. *Bioinformatics* **20**, 538–546 (2004).
111. Ribeiro, A., Zhu, R. & Kauffman, S. A. A general modeling strategy for gene regulatory networks with stochastic dynamics. *J. Comput. Biol.* **13**, 1630–1639 (2006).
112. Zhu, R., Ribeiro, A. S., Salahub, D. & Kauffman, S. A. Studying genetic regulatory networks at the molecular level: delayed reaction stochastic models. *J. Theor. Biol.* **246**, 725–745 (2007).
113. Rao, C. & Arkin, A. Stochastic chemical kinetics and the quasi-steady-state assumption: application to the Gillespie algorithm. *J. Chem. Phys.* **118**, 4999–5010 (2003).
114. Furlong, E. E. A topographical map of spatiotemporal patterns of gene expression. *Dev. Cell* **14**, 639–640 (2008).
115. Gillespie, D. The chemical Langevin equation. *J. Chem. Phys.* **113**, 297–306 (2000).
116. Bernstein, J. A., Khodursky, A. B., Lin, P. H., Lin-Chao, S. & Cohen, S. N. Global analysis of mRNA decay and abundance in *Escherichia coli* at single-gene resolution using two-color fluorescent DNA microarrays. *Proc. Natl Acad. Sci. USA* **99**, 9697–9702 (2002).
117. Kim, V. N. MicroRNA biogenesis: coordinated cropping and dicing. *Nature Rev. Mol. Cell Biol.* **6**, 376–385 (2005).
118. van Driel, R. & Fransz, P. Nuclear architecture and genome functioning in plants and animals: what can we learn from both? *Exp. Cell Res.* **296**, 86–90 (2004).
119. Ogden, S. K. *et al.* p53 targets chromatin structure alteration to repress α-fetoprotein gene expression. *J. Biol. Chem.* **276**, 42057–42062 (2001).
120. Wong, W. W., Tsai, T. Y. & Liao, J. C. Single-cell zeroth-order protein degradation enhances the robustness of synthetic oscillator. *Mol. Syst. Biol.* **3**, 130 (2007).
121. Tetko, I. V. *et al.* Spatiotemporal expression control correlates with intragenic scaffold matrix attachment regions (S/MARs) in *Arabidopsis thaliana*. *PLoS Comput. Biol.* **2**, e21 (2006).
122. Chapman, R. D. *et al.* Transcribing RNA polymerase II is phosphorylated at CTD residue serine-7. *Science* **318**, 1780–1782 (2007).
123. Brodersen, P. *et al.* Widespread translational inhibition by plant miRNAs and siRNAs. *Science* **320**, 1185–1190 (2008).
124. Zhou, T. *et al.* Identification of primary transcriptional regulation of cell cycle-regulated genes upon DNA damage. *Cell Cycle* **6**, 972–981 (2007).
125. Mangan, S. & Alon, U. Structure and function of the feed-forward loop network motif. *Proc. Natl Acad. Sci. USA* **100**, 11980–11985 (2003).
126. Ingram, P. J., Stumpf, M. P. & Stark, J. Network motifs: structure does not determine function. *BMC Genomics* **7**, 108 (2006).
127. Remy, É., Ruet, P. & Thieffry, D. Graphic requirements for multistability and attractive cycles in a Boolean dynamical framework. *Adv. Appl. Math.* (2008).
128. Richard, A. & Comet, J. Necessary conditions for multistationarity in discrete dynamical systems. *Dis. Appl. Math.* **155**, 2403–2413 (2007).
129. Stern, S., Dror, T., Stolovicki, E., Brenner, N. & Braun, E. Genome-wide transcriptional plasticity underlies cellular adaptation to novel challenge. *Mol. Syst. Biol.* **3**, 106 (2007).
130. Bar-Even, A. *et al.* Noise in protein expression scales with natural protein abundance. *Nature Genet.* **38**, 636–643 (2006).
131. Fisher, R. A. On the dominance ratio. Reprint in *Bull. Math. Biol.* **52**, 297–318 (1990).
132. May, R. M. *Theoretical Ecology: Principles And Applications* (Blackwell Scientific Publications, Oxford, 1981).
133. Nei, M. *Molecular Evolutionary Genetics* (Columbia University Press, New York, 1987).
134. Wright, S. Evolution in mendelian populations. *Genetics* **16**, 97–159 (1931).
135. *Min.* Lee, J., Gianchandani, E. P., Eddy, J. A. & Papin, J. A. Dynamic analysis of integrated signaling, metabolic, and regulatory networks. *PLoS Comput. Biol.* **4**, e1000086 (2008).
136. Tomita, M. *et al.* E-CELL: software environment for whole-cell simulation. *Bioinformatics* **15**, 72–84 (1999).
137. McAdams, H. H. & Shapiro, L. Circuit simulation of genetic networks. *Science* **269**, 650–656 (1995).
138. Kierzek, A. M. STOCKS: Stochastic kinetic simulations of biochemical systems with Gillespie algorithm. *Bioinformatics* **18**, 470–481 (2002).

Acknowledgements
We thank I. Ulitsky and I. Gat-Viks for their comments on the manuscript. This study was supported in part by a grant to the APO-SYS consortium from the European Community's Seventh Framework Programme, and by a grant from the Ministry of Science, Culture and Sport, Israel, and the Ministry of Research, France. G.K. was supported in part by a fellowship from the Edmond J. Safra Bioinformatics program at Tel Aviv University, Israel.

FURTHER INFORMATION

Ron Shamir's homepage:
www.cs.tau.ac.il/~rshamir

SUPPLEMENTARY INFORMATION

See online article: S1 (table)

ALL LINKS ARE ACTIVE IN THE ONLINE PDF

© 2008 Macmillan Publishers Limited. All rights reserved

4. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case

BMC Systems Biology

METHODOLOGY ARTICLE Open Access

Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case

Guy Karlebach*, Ron Shamir

Abstract

Background: Mathematical modeling of biological networks is an essential part of Systems Biology. Developing and using such models in order to understand gene regulatory networks is a major challenge.

Results: We present an algorithm that determines the smallest perturbations required for manipulating the dynamics of a network formulated as a Petri net, in order to cause or avoid a specified phenotype. By modifying McMillan's unfolding algorithm, we handle partial knowledge and reduce computation cost. The methodology is demonstrated on a glioma network. Out of the single gene perturbations, activation of glutathione S-transferase P (GSTP1) gene was by far the most effective in blocking the cancer phenotype. Among pairs of perturbations, NFkB and TGF-β had the largest joint effect, in accordance with their role in the EMT process.

Conclusion: Our method allows perturbation analysis of regulatory networks and can overcome incomplete information. It can help in identifying drug targets and in prioritizing perturbation experiments.

Background

In contrast to the gene-centric approach, systems biology [1] emphasizes the importance of the interactions between different genes in determining the phenotype. Instead of asking "what is the role of gene A", the question becomes "what is the role of gene A in system B". The activity (or inactivity) of a gene is therefore not viewed as an isolated event, but assigned a meaning in the context in which it is active. An analogy from the sphere of computer science equates the genome to a database, and the system's dynamic behavior to the execution of a computer program that uses the database [2-4]. This paradigm shift has two major implications for the biomedical community. First, it complicates understanding cellular processes as each component must be considered with respect to its environment. Second, the fact that alternative phenotypes correspond to alternative dynamic behaviors of the system offers considerable advantages, because it is technically easier to influence the dynamics of a cellular network than to modify the information coded in the genome. Combining computational tools, which can help overcome the complexity of biological networks, with wet lab testing can spearhead system-oriented research. In this paper we present a method that was developed with this principle in mind. Focusing on gene regulatory networks, we develop a method to find minimal perturbations that change the network dynamics. By modifying established network analysis algorithms from the field of computer science, we are able to cope with some of the difficulties commonly associated with this objective.

An important tool for network analysis that will be used in this work is network perturbation. A common procedure in model analysis, it refers to applying a modification of the network and observing its resulting dynamic behavior. Knockout, knock-down or overexpression of a gene in the network are examples of possible perturbations. The exact type of perturbation varies with the model and the goals of the modeler. In some cases the motivation is to observe how single entities respond [5,6], while in others it is to determine network robustness [7] or change in the global state [8,9]. For example, Sridhar et al. [10] find enzymes whose inactivation eliminates compounds from a metabolic network.

* Correspondence: guykarle@post.tau.ac.il
Tel-Aviv University, Haim Levanon St., 69978, Tel-Aviv, Israel

BioMed Central

© 2010 Karlebach and Shamir; licensee BioMed Central Ltd. This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/2.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The implementation of a perturbation for our purposes is described in the Methods section.

A related concept in theoretical computer science is *Minimal Cut Sets* [11]. In reliability theory, network elements (e.g. edges) have a failure probability (e.g. an electronic component that has a chance for malfunction). A network is called *reliable* if a set of paths within it connect a given subset of vertices, and the joint probability of the paths is above a given threshold. A minimal cut set is the smallest set of elements whose removal from the network makes the network unreliable. Network reliability shares some important similarities with the concepts proposed in this work, as we also associate the existence of non-existence of network elements with probabilities. A main difference between the two approaches is that identification of minimal cuts sets is a method for analyzing a network via its structural properties. In contrast, our analysis will address the network dynamics and hence will be based on the concept of trajectories, as explained below.

Our first modeling choice will be to model the network's regulators as discrete entities, an approach that proved effective in previous genetic regulatory network (GRN) analyses [7,12-16]. This level of abstraction reduces the need of the modeler to provide fine details [17], while being detailed enough to capture the main features of the GRN dynamics and render them easier to analyze. In addition, the abstraction lends itself to the development of effective methods for incorporating uncertainty in the regulatory functions [18-22]. The *global state* of a network is defined as a vector whose entries are the local states of all the network's components. The network traverses from a certain global state to another in discrete time steps as a result of the activity of regulation functions. We assume that regulation functions act in an asynchronous manner: that is, that at each time step any regulation function can occur, provided its output changes the global state. A *trajectory* is a sequence of global states that the network can traverse in sequence.

Given a set of trajectories T and a set of global states S, S is called a ***phenotype*** of T if every trajectory in T visits only states of S. Similarly, S is called a ***prohibited phenotype*** of T if no trajectory in T reaches any state in S. We say that a network N has a ***phenotype*** S (***avoids a phenotype*** S) with respect to a global state g if the set S is a phenotype (prohibited phenotype) of the group of trajectories that the network generates starting from the initial state g. The following question can now be formulated: "How can the network dynamics be manipulated in order to generate or avoid a specific phenotype?" Answering this question has important practical implications, such as promoting the discovery of novel drug targets [23-25] or the design of synthetic biological systems [26,27]. Therefore, it is desirable to have a systematic way to answer the question for different networks under different conditions.

This is quite difficult, even under the simplified discrete model of GRNs: first, model dynamics can be highly complex, and second, experimental methods give only indirect clues about the network design. The second problem makes it difficult to construct models for networks that have not been extensively studied, especially when the number of participating entities is large. As for the complexity of network dynamics, consider the simple example of a network of n genes where each gene is regulated by some of the others. Assuming that a gene can be in one of two states, ON or OFF, the network can assume 2^n different global states. For ten genes, this results in over one thousand states. For twenty genes, there will be over a million states. Hence, it is possible that from some initial states the network will traverse an exponential number of states. Even this scenario is a simplification, because it assumes the network is known with perfect accuracy, which is seldom true. We will address these problems in the following sections.

In this study, given some initial states of the system and a desired phenotype, we will determine how a network should be perturbed in order to generate that phenotype, where a perturbation sets the level of one or more entities and thus changes the network's traversals between global states. In order to apply our algorithm efficiently to the Boolean model, we translate the network into a Petri net [28], and utilize McMillan's unfolding algorithm [29] to search the state space of a perturbed network. When the structure of the network is not fully known, we assign probabilities to alternative structures, redefine a phenotype probabilistically, and generalize our method to handle this case. To the best of our knowledge, this is the first method that integrates the trajectories of multiple alternative network structures, an important objective given the quality of current knowledge about biological networks. We demonstrate this methodology on the human glioma GRN.

Results

Algorithm

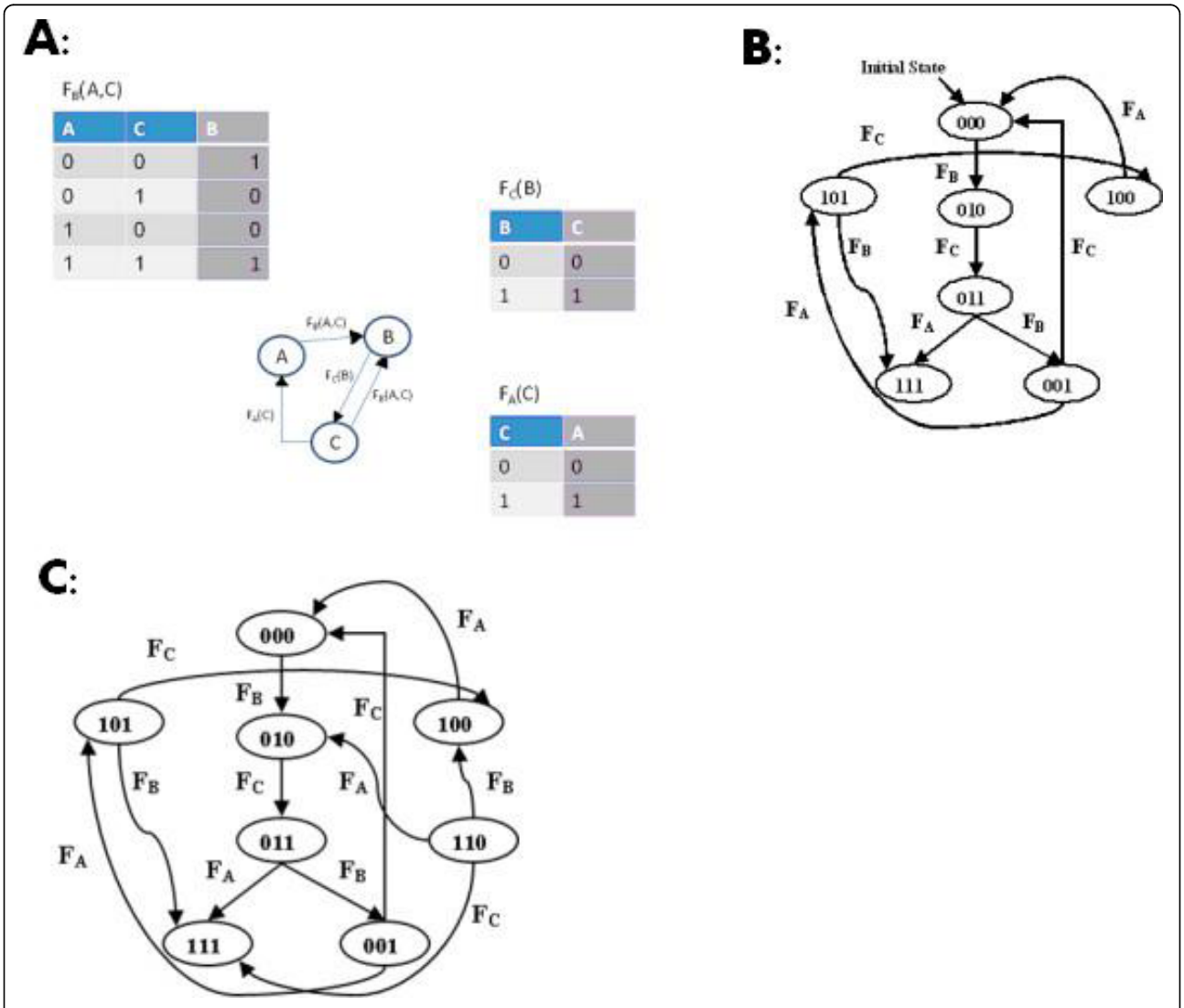
Our model represents each gene by a distinct entity that can take one of two levels: level 1 means a gene is expressed and level 0 means it is not. The levels of genes are controlled by Boolean regulation functions, which can have any of the other genes (and even the gene itself) as inputs. The initial global state of the model is a vector that assigns an initial level to each gene. Starting from the initial state, the global state of the network can change in discrete time steps, where

one regulation function is activated at each step. In other words, regulation functions can act in any order, and not all at the same time. This means that there can be more than one trajectory per initial global state. Figure 1A illustrates the model with a simple GRN.

We first give a description of our algorithm, followed by its implementation using Petri nets. The algorithm takes as input a network model, the network's state graph, a set of initial states A, and a group of states B. It outputs all the minimal perturbations that cause the network to have phenotype B with respect to every state in A.

Given a GRN N, its *state graph* is a directed graph G (V,E) whose nodes are global states of N. In G there is an edge (**a** ,**b**) if and only if there is a regulation function f that can act in state **a** and lead directly to state **b**. The label of (**a**,**b**) is the function f. Note that several labels are possible on the same edge if it is a self loop. Figure 1 illustrates a simple GRN and its state graph.

We define two operations on a network: An ***activation*** of a gene causes the gene to stay fixed on level 1. For example, if we activate gene B in Figure 1C, the network dynamics will lead to the endpoint state 111. Similarly, a ***repression*** of a gene causes the gene to stay



A	C	B
0	0	1
0	1	0
1	0	0
1	1	1

B	C
0	0
1	1

C	A
0	0
1	1

Figure 1 A simple GRN and its state graph. A: A simple GRN. The network contains three entities, A, B and C. Entities A and C regulate entity B, B regulates C, and C regulates A. Each table shows the level of the regulated gene when its regulation function acts, depending on the regulators' levels. B: The state graph of the GRN. Nodes correspond to global states (with coordinates A, B, C from left to right), and edges to transitions between these states. The labels on the edges show the regulation functions that cause this transition. C: The restricted state graph starting from the initial state 000. Only states that are reachable by transitions from 000 are shown. For simplicity, self loops are not shown. Sequences of state traversals that follow from the initial state can be cyclic (return to the same state) or can lead to the endpoint state 111.

fixed on level 0. In Figure 1C, repressing gene A will result in cyclic behavior that will lead back to the initial state. Self loops in the state graph are meaningless under these definitions, and therefore are omitted.

The biological means of activation and repression vary depending on the mechanisms of regulation [30-32]. Common examples are knock-down, overexpression, and addition of inhibitors and activators, but less standard examples can be thought of, such as insertion of artificial entities [33] or de-novo network design [26].

A ***network perturbation*** is a set of operations (activations and/or repressions) on genes. The maximal allowed size k of a perturbation P is assumed to be a small constant. An edge in the state graph ***contradicts*** perturbation P if it leads to a state in which an activated gene is at level 0 or a repressed gene is at level 1.

Let A and B be two groups of states in the state graph G, such that $A \subseteq B$. We want to find a minimal perturbation such that the network has phenotype B with respect to every state of A. Assuming that k is constant, the following algorithm runs in time polynomial in the size of G

1. For i = 0,...,k do
   For every possible perturbation set P of size i, do
   i. Modify the group A according to P; i.e. set the level of activated genes to 1 and the level of repressed genes to 0.
   ii. Add a node s and connect it by outgoing unlabeled edges to all the nodes of group A.
   iii. Add a node t and connect each node of group $\overline{B}$ to it using outgoing unlabeled edges.
   iv. Create a modified graph G' from G by removing all edges that contradict operations in P.
   v. If there is no path from s to t, output the set P and stop.
2. If this step is reached, then there is no solution of size ≤k.

If one is interested in all the minimum solutions, then instead of halting after finding the first good perturbation of size i, halt only after enumerating all perturbations of size i. If B is a prohibited phenotype then step 1a(iii) should be changed: the node t should be connected to B instead of $\overline{B}$.

The running time on a state graph G = (V,E) is $O(2^k \cdot n^k \cdot (|V|+|E|))$, where n is the number of entities in the GRN: the creation of G' and searching for paths in it can be accomplished by a BFS, and the loop occurs $O(2^k n^k)$ times. Hence this algorithm is practical if we assume that G is not too large. However, since $|V| = 2^n$, only very modest sized GRNs can be directly solved this way in practice.

To address this complexity problem, we will formulate our problem using Petri nets and present a methodology that copes better with the state explosion problem.

Petri nets are a modeling formalism that has been used to model different types of biological networks [34-40]. A Petri net is a bipartite graph composed of two sets of nodes: *places* and *transitions* (see Figure 2A). The transitions set contains nodes that represent discrete events that can occur concurrently. The places set represents network entities. Transitions and places are connected by directed edges that represent interactions between network entities. The places having an edge into (from) a transition are called its preset (postset) places. The global state of the network is given by a discrete assignment of tokens to different places (the level of each entity), and is referred to as ***marking***. For example, the network in Figure 2A has three places, and the marking in I assigns one token to each of the place p_1 and p_2 and zero tokens to p_3. Tokens can be consumed and produced by transitions. The rule that determines token consumption and production is called the ***firing rule***, and it allows a transition to fire (consume and produce tokens) if every one of its preset places contains a specified amount of tokens. When fired, a transition consumes these tokens and produces a set number of tokens to every one of its postset places. See Figure 2A for an example.

Reddy et al [41] introduced the use of Petri nets in the context of systems biology. Later, Chaouiya et al. [42] suggested a methodology for translating Boolean regulatory networks into Petri nets, which we adopt. Additional examples of modeling GRNs with Petri nets are refs. [43-45]. Translating the network to this framework has the advantage of a rich literature on techniques for analyzing the dynamics of Petri nets. In addition, Petri nets are suitable for describing other types of biological networks, such as GRN models with additional metabolic and signaling layers.

McMillan's unfolding algorithm [46] is a method for dealing with the state explosion problem for Petri nets. A full description of the unfolding algorithm can be found in ref. [47]. Briefly, given an initial state, McMillan's algorithm gradually and implicitly records the states reachable from it by constructing a directed graph called a branching process. A ***branching process graph*** begins with places that correspond to the initial marking of the Petri net, and transitions that are added to it can consume from these places and produce new places, thereby representing consumption and production of tokens. A transition can consume only from places that do not belong to conflicting firing sequences, i.e. firing

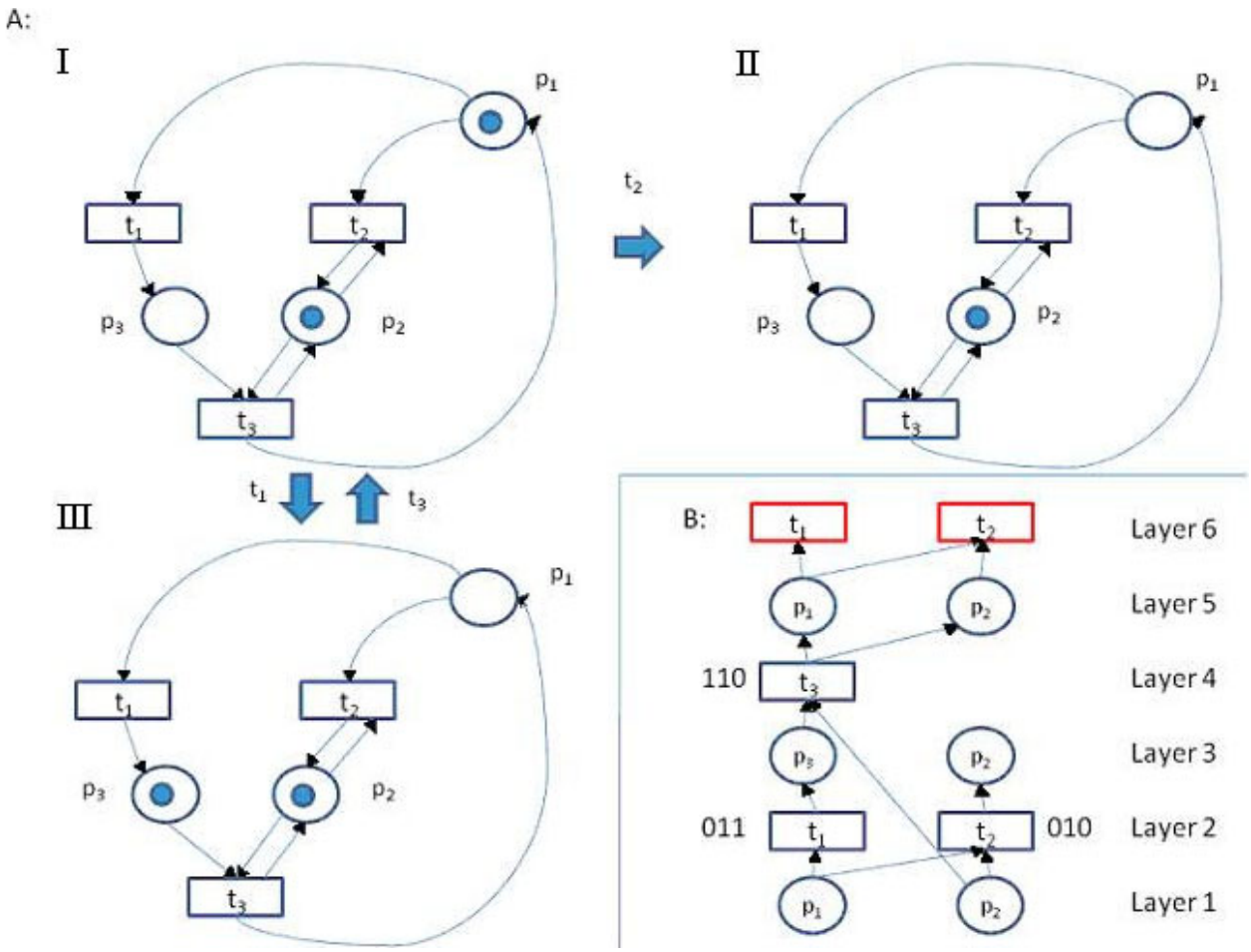


Figure 2 A Petri net and its unfolding. A: A Petri net and its unfolding. The net contains 'places' (light blue circles), the model's entities, and 'transitions' (rectangles), which constitute the regulation functions and define the model's dynamics. Arcs connect input places to transitions, and transitions to their output places. Places that receive discrete values are called tokens (blue dots). A transition that is activated, or 'fired', reduces the tokens in its input places and increases the number of tokens in each of its output places. At any time step, every transition that has enough tokens in its input places may be fired. In the example, every transition consumes one token from every input place, and produces one token at every output place. Labels next to thick arrows indicate which transition fired. Transitions t1 and t3 can be fired in alternation indefinitely, whereas no other transition can be fired after t2 has fired. B: Unfolding of the Petri net. Transitions are represented by rectangles, places by circles. The two places p_1 and p_2 that have tokens in the initial marking in state I are the input-less places of the unfolding. The local configuration of t_2 at layer 2 corresponds to the marking 010, i.e. the marking in which only p_2 contains a token, corresponding to II in **Figure 2A**. The local configuration of t_3 corresponds to the firing of t_1 followed by t_3, and to the marking 110, i.e. the initial marking. The instances of t_1 and of t_2 at layer 6 are cutoff points, since their local configurations' markings are already represented by other local configurations. The graph constitutes a branching process.

sequences that cannot occur concurrently. Thus, additions of new transitions preserve the acyclic property of the branching process graph, and ensure that it represents only feasible firing sequences (Figure 2B). Refs. [46,47] provide excellent illustrations of the algorithm's capacity to reduce the search space on larger network instances.

Every reachable marking has a subset of transitions in the branching process graph that correspond to the firing sequence that generates it. These subsets are called ***configurations***. For a transition t, the set of transitions from which there is a directed path to t is referred to as t's ***local configuration*** (denoted [t]), and is associated with a marking. The marking of [t] is the marking obtained by firing all the transitions that belong to [t].

In the GRN representation that we adopted, every entity e corresponds to two places: one represents its active level and the other represents its inactive level. The firing rules are set so that exactly one of the places is marked at any time, i.e. each pair is place invariant [37]. These places will be called the active and inactive places of the entity e. Figure 3 illustrates this concept.

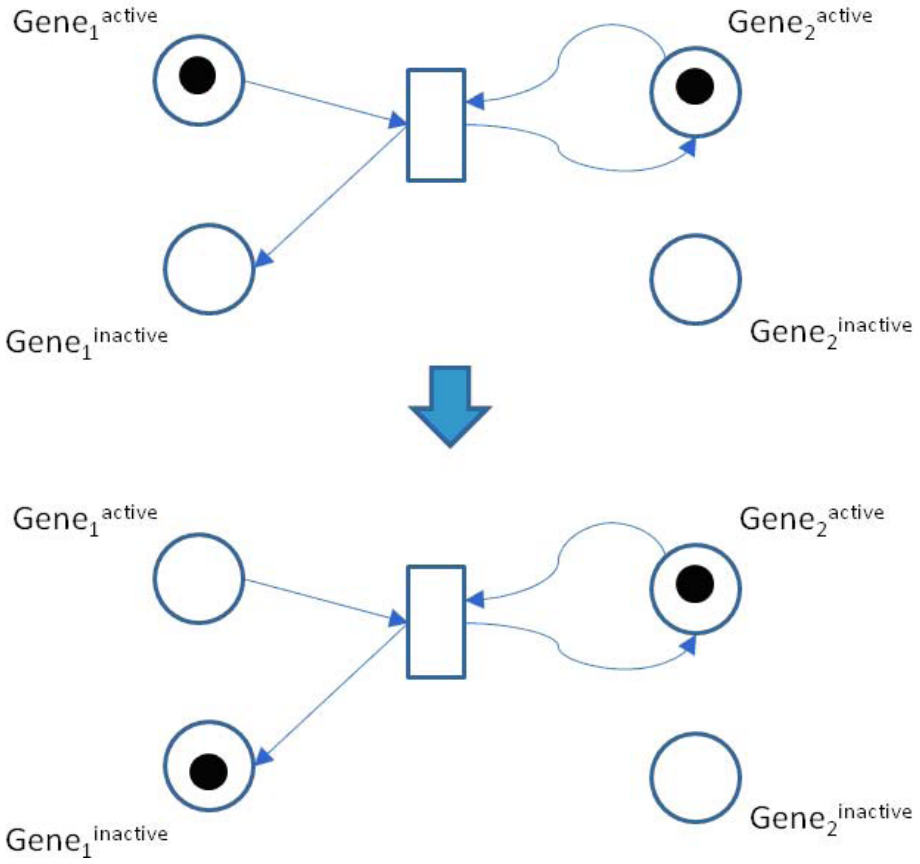


Figure 3 Petri net representation of a Boolean entity. In this example gene 2 inactivates gene 1. Each of them is represented by two places in the Petri net. The upper (lower) part of the figure shows the Petri net before (after) the transition fires. When gene 2 is active, it inactivates gene 1. Therefore, the transition consumes a token from the active place of gene 1, and produces a token to its inactive place. The transition also consumes a token from the active place of gene 2, and produces a token to the same place. The latter consumption and production express the fact that gene 2 needs to be active in order to inactivate gene 1, but the inactivation itself does not change the level of gene 2.

The unfolding algorithm can produce a much smaller graph than the complete state graph. The following preprocessing to the algorithm spans all the states that are reachable from a given initial state under a perturbation P:

1. For every activated entity e in P, set a token in the active place of e.
2. For every repressed entity e in P, set a token in the inactive place of e.
3. Remove all transitions that have edges outgoing to places contradicting P.

When there are several initial states, a branching process graph is generated for each initial state.

The above algorithm requires full information about the GRN model. Since this is usually not the case, we now address the handling of ambiguities in the GRN logic. Consider a network in which every gene can have several alternative regulation functions, each associated with a probability that it is the true regulation. The events corresponding to the true regulations of different genes are assumed to be independent. Hence, the probability of a trajectory is the product of the probabilities of the regulation functions involved in it. Given a parameter α, $0 \leq \alpha \leq 1$, the definition of a phenotype can now be extended as follows: A network has a phenotype P with respect to a set of initial states if every subset S of regulation functions that has probability $\geq\alpha$ generates only trajectories that remain in P. Note that if the condition holds for S it will hold also for every $S' \subseteq S$, which can have higher probability. This definition induces a distribution of all alternative networks into layers. The top layer contains networks with probability $\geq\alpha$. Sets of networks with lower probabilities belong to lower layers, each layer corresponding to a different probability. The lowest layer has probability α^N, where N is the number of entities. Higher layers have lower capacity because there can be less networks with high probability than networks with

low probability (as all probabilities must sum to 1). For networks in the top layer we examine every possible trajectory - this follows from the definition of probabilistic phenotype, since the full set of regulation functions of these networks has probability ≥ α. As we descend in the hierarchy, layers have greater capacities and contain networks of lower probabilities. For every such network we examine only trajectories that are generated by strict subsets of their regulation functions, because the full sets of regulation functions of these networks have probability <α. In other words, in lower layers we still follow the dynamics of every network, but to a lesser extent than in higher layers, and so each structure has an influence on the phenotype in proportion to its probability.

Similarly, a network has a prohibited phenotype P with respect to a set of initial states if every subset S of regulation functions that has probability ≥α does not generate any trajectory that leads to P.

A naïve way to test for a probabilistic phenotype would be to repeat the non-probabilistic algorithm for every set of regulation functions with probability >α. However, the number of such sets grows exponentially with the number of entities that have more than one regulation function. More specifically, assume that there are n genes and every gene has k alternative regulation functions. For each gene, a set can specify one of the k regulation functions or leave that gene unregulated, i.e. not commit to a specific function. This gives rise to $(k+1)^n$ alternative sets of regulation functions. If k is constant, the expression is exponential in n. Next we discuss how to modify the unfolding algorithm to test for a probabilistic phenotype.

Since we translate a regulatory network into a Petri net, every transition of a configuration C in the branching process graph corresponds to a regulation function (recall that in the probabilistic setting, one gene may have several regulation functions). Denote by ϕ(C) the set of regulation functions that are represented by the transitions of C. Note that if C contains a single regulation function for each entity, the size of ϕ(C) is at most the number of entities in the model. Denote by ϕ'(C) the subset of ϕ(C) that contains only regulation functions with probability <1.0. We say that ϕ'(C) is ***unambiguous*** if it does not contain two regulation functions that regulate the same entity.

A key concept in the original unfolding algorithm is a cutoff point; it is a transition t whose local configuration [t] is associated with a marking that is also associated with some other local configuration [t'] that contains fewer transitions. At cutoff points one can prune redundant branches in the constructed branching process graph. Given such a pair of transitions t and t', we modify McMillan's cutoff criterion to handle probabilities by adding another condition that must hold for t to become a cutoff point:

Cutoff criterion 1: $\phi'([t']) \subseteq \phi'([t])$

In addition, we make sure that each local configuration is unambiguous by keeping track of the functions that have been utilized in it, and allowing a transition t to fire from C only if $\phi'(C \cup \{t\})$ is unambiguous.

Finally, in order to save time and space, we add another cutoff criterion to the algorithm

Cutoff criterion 2:

A transition is a cutoff point if the product of the probabilities of regulation functions that are used in its local configuration is <α.

Note that since we tightened the cutoff criterion, the size of the branching process graph can become larger than in McMillan's algorithm.

Theorem 1: The modified version of McMillan's algorithm maintains:

1. For a phenotype P: If there is a set of regulation functions F with probability ≥α that generates a trajectory that does not remain in P, then such a trajectory will be represented by a configuration C in the branching process graph and $\phi'(C) \subseteq F$.
2. For an avoided phenotype P: If there is a set of regulation functions F with probability ≥α that generates a trajectory that leads to P, then such a trajectory will be represented by a configuration C in the branching process graph and $\phi'(C) \subseteq F$.

The proof is provided in the Appendix. Given that the theorem holds, we simply need to construct the branching process graph and test for such a configuration C in order to verify that a phenotype is maintained or avoided.

A Test Case

Shmulevich et al [48] constructed a probabilistic model of a small autonomous subnetwork of genes based on human glioma gene expression data [49] obtained for 588 known genes, in tissue samples with differing levels of glioma severity. The inferred network was used for a Probabilistic Boolean Network (PBN) simulation [50] by Akutsu et al. (The probability of a regulation function is the sum of coefficients of determination (CODs) between expression levels of each of its input genes and the output gene divided by the sum of CODs of the expression level of the output gene and all its potential regulators [48].) In view of its intriguing dynamic behavior and biomedical relevance, we used that network

model to test our minimum perturbation set algorithm. After removing a gene that had no regulators, 14 entities remained, each associated with 1-3 regulation functions. When there is more than one function for an entity, the functions are assigned probabilities that add up to one. Six genes have a single regulation function, seven genes have two alternative regulation functions, and one gene has three possible regulation functions. A description of the logic functions appears in ref. [50]

We transformed this network into a Petri net (Figure 4), and applied our algorithm to find minimum perturbations from 1000 random initial states. The initial states were tested in this way because the "biologically correct" initial states cannot be derived from current knowledge. Moreover, since the glioma network is manifested in dividing cells that constantly redistribute their molecular contents, it is not unrealistic to assume a variety of initial states.

We defined the prohibited phenotype S of the network as where as the set of global states in which the gene Tie-2 for the receptor Tie-2[51] and the gene GNB1 for the human G-protein beta subunit [52] are both expressed (see Figure 4). The set S was selected following reports that vasculogenesis, an important phase in tumor progression, is initiated by a signal to the receptor Tie-2 that is propagated through a G protein [53,54]. Since repression of either Tie-2 or GNB1 is a trivial solution, these genes were excluded from the perturbations tested. Similarly, initial states in which both Tie-2 and GNB1 are active were excluded from the set of possible initial states, because there is trivially no solution from these states. The parameter α, which determines the least probability of a trajectory that will be explored - and hence the running time of the algorithm, was set to 0.05.

Figure 5 shows the distribution of solution sizes found. In about 0.5% of the initial states the phenotype is avoided without any perturbation. Perturbations of size 1 cause the network to avoid the phenotype in about 65% of the initial states, and perturbations of size 2 and 3 are needed in the remaining cases.

Figure 6 shows the frequency of perturbations of different sizes. It should be pointed out that when there are several perturbations of the minimal size, all of them are found. As can be seen in the figure, the number of perturbations that provide minimal solutions is much smaller than the total number of possible perturbations. The activation of the gene GSTP is by far the most abundant operation in size 1 perturbations. The probability that all the operations that appear at least once in size 1 perturbations are equally likely is 0.0001 (X^2 test, 14 degrees of freedom). In addition, in contrast to other genes, GSTP is only activated and never repressed. Reassuringly, these facts are consistent with experimental observations [55]

- Mice deficient in GSTP are viable, fertile, with life spans essentially similar to animals not deficient in the gene. However, they show an enhanced susceptibility to carcinogen-induced skin papillomas.
- The absence of GSTP increases the activity of stress kinases, which results in changes in gene expression that enhance cell proliferation pathways.
- Hypermethylation of the GSTP regulatory region is a common somatic alteration identified in human prostate cancer. This alteration results in the loss of GSTP expression and is proposed to occur during pathogenesis of the disease.
- In the latter case it was suggested that there could be therapeutic value in restoring GSTP activity, although it has not been tried.

Our results are consistent with these observations. They single out the activation of GSTP as an operation that blocks tumor progression.

In initial states where no size 1 perturbation suffices, GSTP does not participate in a perturbation. This is consistent with the observation that GSTP is often highly expressed in cells that have already turned malignant.

There are four common perturbations of size 2. All of them include repression of natural killer enhancing factor B, accompanied by activation of one of BCL2A, TGF-β, NFκB, or Beta-Actin. The first two operations are associated with repression tumor cell death, while the latter three are associated with constant induction of cell migration.

The most common perturbations of size 3 are activation of both TGF-β and NFκB or repression of these entities in addition to activation of the entity BCL2A1.

These results can be understood in the context of the stages of glioma progression Zagzag et al. distinguish three stages that precede the formation of new blood vessels.

a. In the first stage tumor cells migrate and adhere to existing blood vessels. Huber et al. concluded that NFκB, at least in part, substitutes for TGF-β in the process of EMT, which is essential for tumor migration.
b. In the second stage of tumor progression, blood vessel cells undergo apoptosis, and the nearby tumor cells undergo necrosis. Breaking cell-to-cell adhesion is thought to be a trigger for the apoptotic process. Disrupting cell migration or preventing apoptosis may halt the regulatory program at the second stage.
c. In the final stage, new blood vessels are formed. The initial states that correspond to this stage are included in the prohibited phenotype.

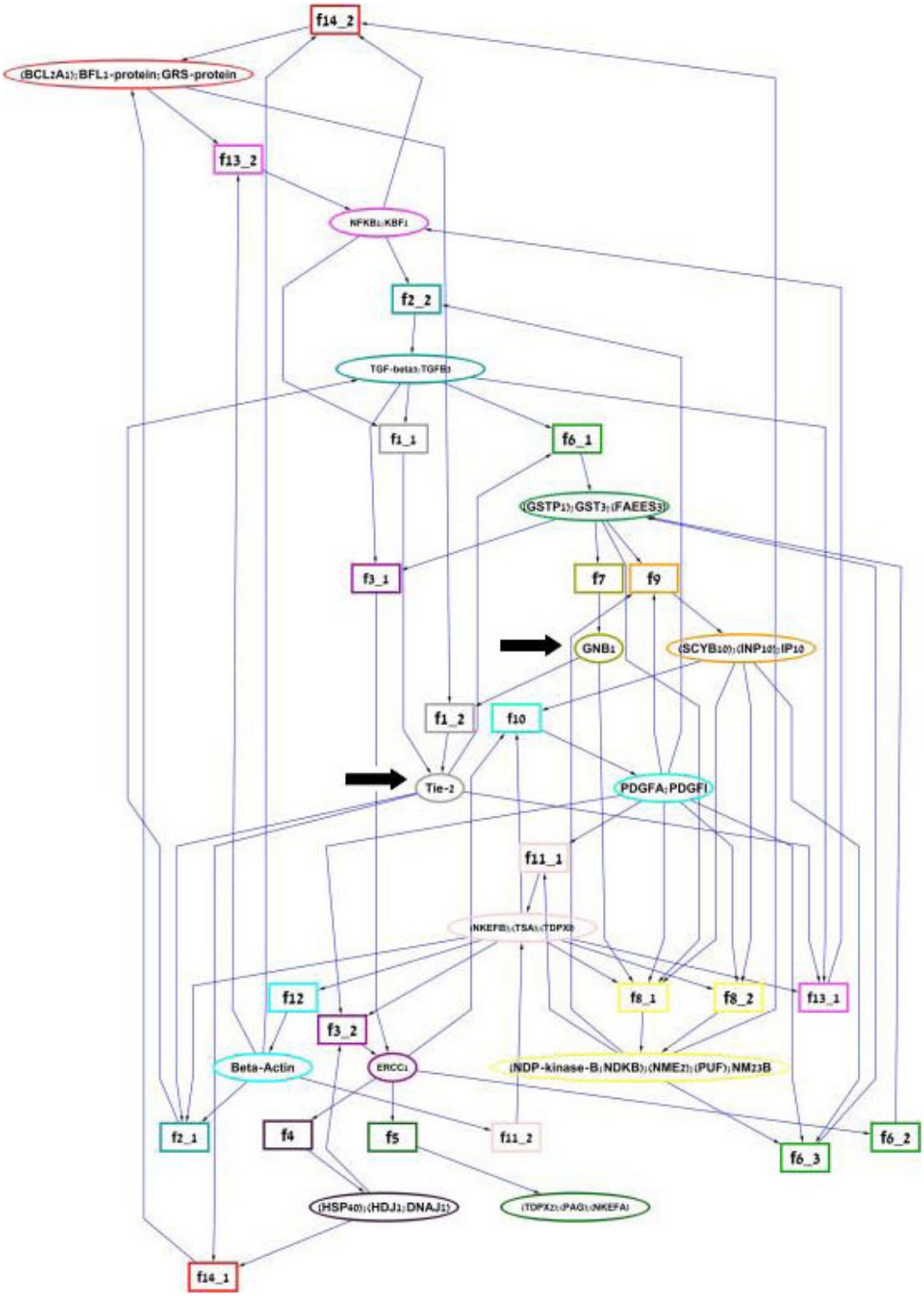


Figure 4 The glioma network. Genes (ovals) and their alternative regulation functions (rectangles) are bordered by frames of the same color. Ovals contain the name of the relevant human gene, following the nomenclature in [48]. Rectangles contain the name of the regulation function [49]. Regulation functions are connected by directed edges to the gene they regulate. Regulators are connected by directed edges to the regulation functions in which they are involved. The figure was generated using Cytoscape [59]. The bold arrows indicate the two entities that constitute the prohibited phenotype (see text).

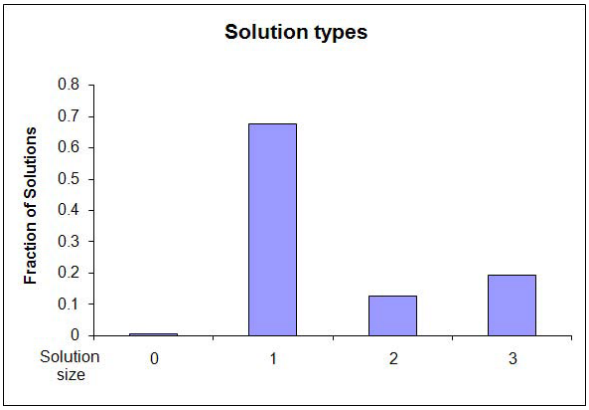


Figure 5 Frequency of perturbation size needed. The histogram plots the fraction of solutions of each size. "Size 0" indicates states from which the avoided phenotype is not reachable.

Thus, the combination of anti-apoptotic signals in addition to setting of cell migration signals in size 2 and size 3 perturbations may correspond to blocking of apoptosis and disrupting the formation of blood vessels, and halting the regulatory program at the second stage. The most common size 1 perturbation may correspond to prevention of the first stage of tumor progression.

We interpret our findings in light of existing experimental data as follows: GSTP can prevent the initiation of the vasculogenesis program. In later stages it is no longer effective, but other genes can be disrupted in order to halt this program, depending on the stage of vasculogenesis - the later the stage the larger the perturbation that is needed.

All executions were performed on x86 64 bits machines with Pentium IV or Zeon processor and at most 2 GB RAM. Jobs were run in a time-sharing environment and therefore the running times are only an upper bound. The program code was written in C. After 12 hours, 70% of the jobs finished. Since the rest of the jobs required more than 24 hours, we used only those 70% that concluded early in our analysis.

Discussion

System-level analysis presents researchers with new challenges and at the same time offers new opportunities for better understanding of the biology. The complexity of reconstructing biological networks and analyzing their dynamics makes computational tools essential for system-level approaches [56,57]

We have described a computational method that determines the minimum size perturbations required for obtaining (or avoiding) a specific phenotype. Because the function of genes depends on the global context in which they are active - the state of the system - the phenotype cannot be represented by the activity or inactivity of a single gene, but rather by the global state of the network. We therefore defined a phenotype based on network dynamics as a set of global states that must be preserved (or avoided), and designed an algorithm that follows this definition. The method was implemented for a probabilistic Boolean model, and was demonstrated on a glioma network.

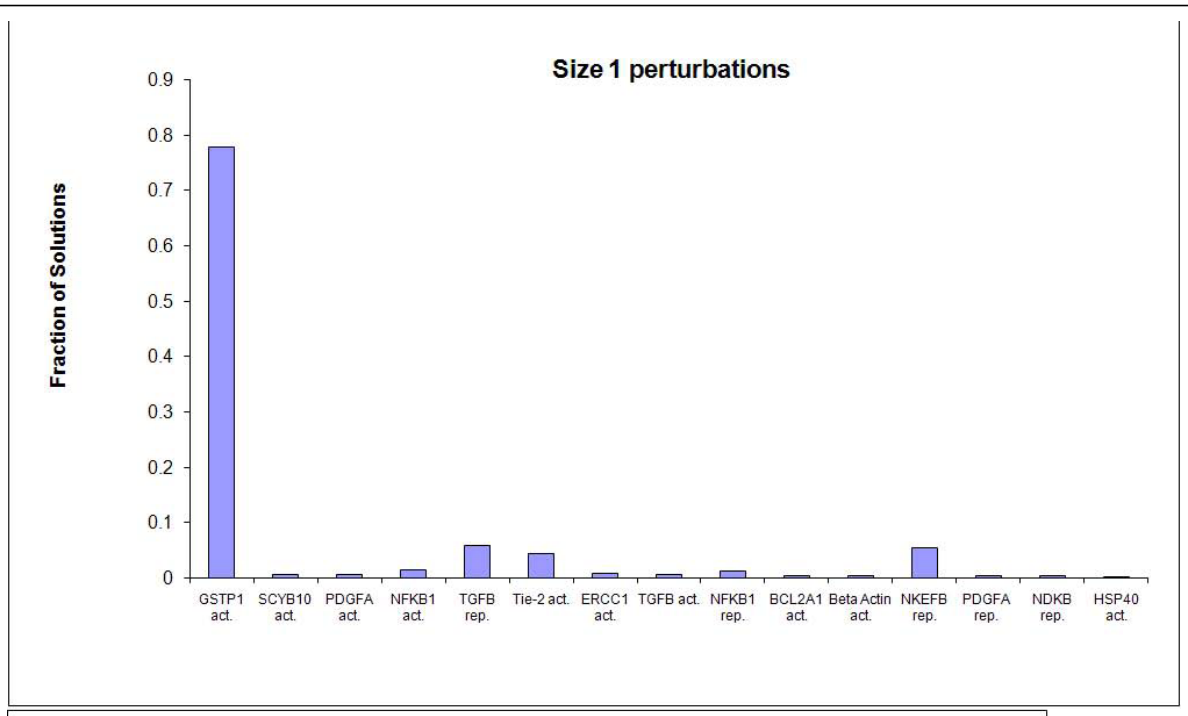
We showed that two major problems in network analysis, namely state explosion and partial knowledge, can be alleviated by translation to Petri nets and extensions of the unfolding technique. Our method demonstrates the power of computational analysis of the network's dynamics. On the glioma network it singled out one perturbation of size 1 whose effect on the phenotype was strongest. That perturbation has strong support in the literature. In addition, the most prominent perturbations of sizes 2 and 3 can be explained in the context of glioma progression. We expect this method can be used to derive such insights for other networks, because it does not require perfect knowledge and uses the broadly applicable Petri net semantics.

Though the paper focuses on GRNs, the suggested computational method can be applied to signaling or metabolic networks and to networks that integrate several layers, e.g. metabolic and regulatory. Petri nets have been used for modeling all these network types.

Our method has several limitations: Some instances of the problem still require exponential running time, making our method impractical for finding a minimal perturbation for large models. Our method is sensitive to modeling accuracy and depends on the correctness of prior knowledge, albeit in a probabilistic setting. In addition, we assume that the network is asynchronous, while in some cases the order of occurrence of regulation functions may be determined by large rate differences among them.

Improving the algorithm's performance is one of our future goals. The minimal perturbation algorithm can be used in practice only when the size of a perturbation is small; allowing larger perturbations requires new algorithmic ideas. However, to date it is impractical to perturb more than a few entities in the cell, making speedups useful primarily for analyzing larger networks. The case where some of the entities are synchronized and some are not can also be considered (Ref. [37] shows how synchronized networks can be modeled with Petri nets). Finally, the unfolding algorithm may be improved by modifying the cutoff criterion.

Other model checking techniques for Petri nets are described in ref. [58]. Though not directly related to unfolding, they provide alternative attempts to battle the state explosion problem when using the Petri net semantic.



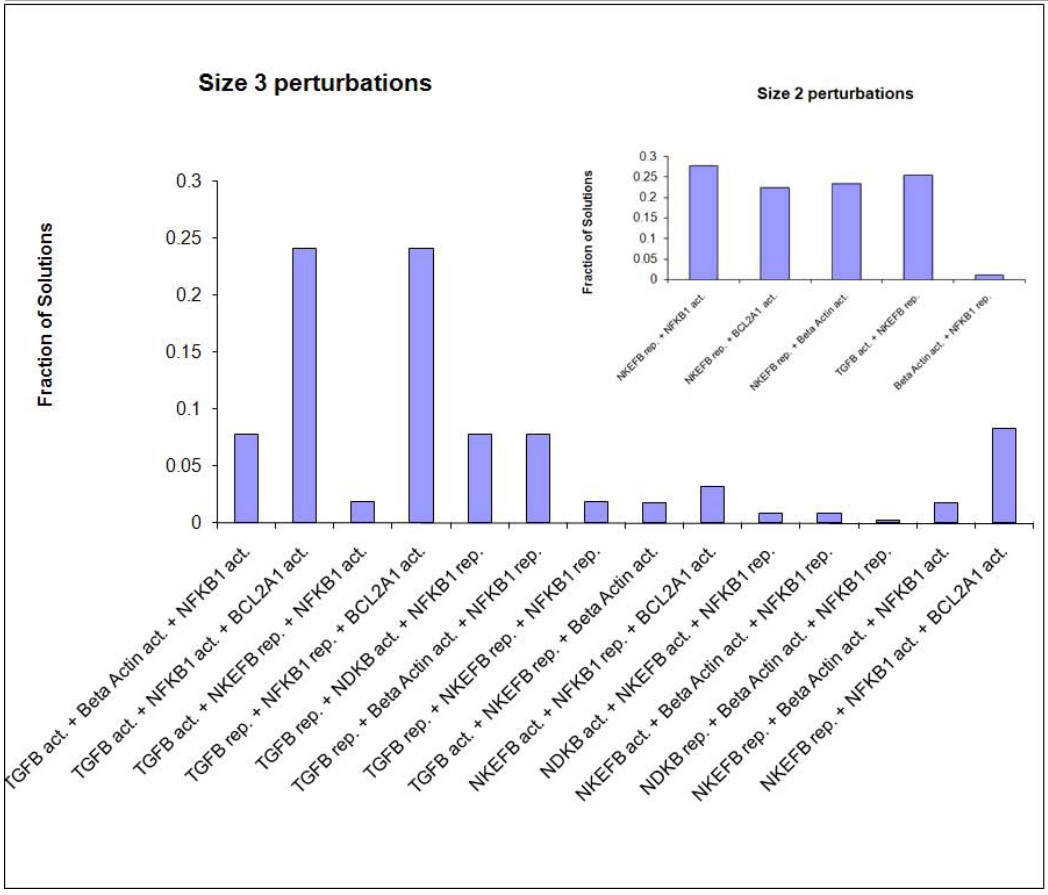


Figure 6 Frequency of minimal perturbations of sizes 1, 2 and 3. Each bar shows the proportion of the occurrences of a different perturbation. Act: activation; rep: repression

Conclusion

The ability to effectively manipulate a given network's dynamics in order to produce a desired behavior depends both on advances in experimental techniques and on the ability to computationally analyze the network. We presented a computational methodology for determining a minimum size perturbation yielding a desired phenotype that copes with some of the urgent difficulties in modeling. Application of this methodology to ongoing experimental projects and extension of its theoretical foundations are among our future goals.

Appendix

Theorem

Let P be a phenotype (respectively, a prohibited phenotype). If there is a set of regulation functions F with probability $\geq\alpha$ that generates a trajectory that does not remain in P (respectively, that leads to P), such a trajectory will be represented by a configuration C in the branching process graph and $\phi'(C) \subseteq F$.

Proof of the theorem

We prove the theorem for a phenotype. The proof for a prohibited phenotype is symmetric.

Let S be a state that does not belong to the phenotype, and let F be a set of regulation functions with probability $\geq\alpha$ such that F generates some trajectory that reaches S.

The proof is by induction on the number of state traversals (edges) in the state graph that are needed for reaching the state S. For purposes of the proof, we will use the term "infinite branching process graph" for a branching process graph in which cutoff points are not applied, and the term "finite branching process graph" for the branching process graph that is created by the algorithm.

Base

Zero state traversals, i.e. the initial state. The initial state is reachable by every set of regulation functions. In the branching process graph it is represented by the initial marking. The set ϕ of the initial marking is the empty set, and therefore the theorem holds for the base case.

Assumption

Every state that is reachable by a set of regulation functions with probability $\geq\alpha$ and N-1 edge traversals is represented in the branching process graph.

Step

Let π be the path in the state graph that leads to S, and let N be the length (number of state traversals) in π. We want to show that some trajectory leading to S that is generated by a set of regulation functions with probability $\geq\alpha$ is represented in the branching process graph.

Let e be the last edge (state traversal) in π. The resulting path $\pi' = \pi/\{e\}$ ends at some state S' and is of length N-1. Since there is a path of length N-1 to S' whose functions belong to the set F, by the inductive hypothesis S' is represented by some configuration C' in the branching process graph and $\phi'(C') \subseteq F$.

Let t' be the transition that represents the edge e. We want to show that t' can be added to the branching process graph to yield a configuration C that represents π.

First, all of the input places that t' requires are output places of C', and therefore are not in conflict. Add t' to the branching process graph such that it consumes from these places. Since $\phi'(C') \cup \{t'\} \subseteq F$, t' will not be set as a cutoff point according to cutoff criterion 2. If t' is not set as a cutoff point due to cutoff criterion 1 either, then we are finished, because we obtained a configuration C that represents S. Therefore assume that t' is set as a cutoff point according to cutoff criterion 1.

In the latter case, [t'] is already represented by some local configuration C''. According to cutoff criterion 1, $\phi'(C'') \subseteq \phi'([t']) \subseteq F$. if $\phi(C'') \not\subseteq F$, i.e. there are some regulation functions with probability 1.0 that are used in C'' and not in [t'], then we will build the configuration C for a set F' that has the same probability as F, i.e. probability $\geq\alpha$. Otherwise, we will build C for the set F. In any case we will have $\phi'(C') \subseteq F$.

Now, note that if [t'] was not set as a cutoff point, then the set of transitions C/[t'] could have been added to [t'] in the branching process graph to yield the configuration C. Intuitively, imagine that the branching process graph is infinite, i.e. cutoff points are not used at all. Since C'' corresponds to the same marking as [t'], transitions that correspond to the transitions of C/[t'] in the original Petri net can be added to C'' in an infinite branching process graph to produce a new configuration C''' that is smaller than C and $\phi'(C''') \subseteq F$. If the new configuration is not represented in the finite branching process graph, then it must also contain a cutoff point t''. We can repeat the same process with the cutoff point t'', until we get a configuration that has the same marking as C, uses only functions of F (or of a set with equal probability), and is represented in the finite branching process graph. We will surely obtain such a configuration because each time that we repeat this process the local configuration that corresponds to the cutoff point becomes smaller, and the minimal size of a configuration is 0.

Acknowledgements

We thank Yoel Kloog and Marcelo Ehrlich for helpful discussions and comments. This study was supported in part by the European Community's

Seventh Framework Programme under grant agreement n° HEALTH-F4-2007-200767 for the APO-SYS project. GK is supported by a fellowship from the Edmond J. Safra Bioinformatics Program at Tel Aviv University.

Authors' contributions

The authors designed and developed the method together. GK implemented the algorithm and carried out the testing. Both authors wrote the manuscript.

Received: 3 December 2009 Accepted: 25 February 2010
Published: 25 February 2010

References

1. Kitano H: **Systems biology: a brief overview.** *Science* 2002, **295**:1662-1664.
2. Noble D: *The music of life: biology beyond the genome* Oxford: Oxford University Press 2006.
3. Noble D: **Claude Bernard, the first systems biologist, and the future of physiology.** *Exp Physiol* 2008, **93**:16-26.
4. Noble D: **Genes and causation.** *Philos Transact A Math Phys Eng Sci* 2008, **366**:3001-3015.
5. Ihekwaba AE, Broomhead DS, Grimley R, Benson N, White MR, Kell DB: **Synergistic control of oscillations in the NF-kappaB signalling pathway.** *Syst Biol* 2005, **152**:153-160.
6. Bentele M, Lavrik I, Ulrich M, Stosser S, Heermann DW, Kalthoff H, Krammer PH, Eils R: **Mathematical modeling reveals threshold mechanism in CD95-induced apoptosis.** *J Cell Biol* 2004, **166**:839-851.
7. Li F, Long T, Lu Y, Ouyang Q, Tang C: **The yeast cell-cycle network is robustly designed.** *Proc Natl Acad Sci USA* 2004, **101**:4781-4786.
8. Herrgard MJ, Lee BS, Portnoy V, Palsson BO: **Integrated analysis of regulatory and metabolic networks reveals novel regulatory mechanisms in Saccharomyces cerevisiae.** *Genome Res* 2006, **16**:627-635.
9. Akutsu T, Hayashida M, Ching WK, Ng MK: **Control of Boolean networks: hardness results and algorithms for tree structured networks.** *J Theor Biol* 2007, **244**:670-679.
10. Sridhar P, Song B, Kahveci T, Ranka S: **Mining metabolic networks for optimal drug targets.** *Pac Symp Biocomput* 2008, 291-302.
11. Colbourn CJ: *The combinatorics of network reliability* New York: Oxford University Press 1987.
12. Smith J, Theodoris C, Davidson EH: **A gene regulatory network subcircuit drives a dynamic pattern of gene expression.** *Science* 2007, **318**:794-797.
13. Davidson EH, Rast JP, Oliveri P, Ransick A, Calestani C, Yuh CH, Minokawa T, Amore G, Hinman V, Arenas-Mena C, *et al*: **A genomic regulatory network for development.** *Science* 2002, **295**:1669-1678.
14. Kauffman S, Peterson C, Samuelsson B, Troein C: **Random Boolean network models and the yeast transcriptional network.** *Proc Natl Acad Sci USA* 2003, **100**:14796-14799.
15. Biondi EG, Reisinger SJ, Skerker JM, Arif M, Perchuk BS, Ryan KR, Laub MT: **Regulation of the bacterial cell cycle by an integrated genetic circuit.** *Nature* 2006, **444**:899-904.
16. Covert MW, Palsson BO: **Transcriptional regulation in constraints-based metabolic models of Escherichia coli.** *J Biol Chem* 2002, **277**:28058-28064.
17. Karlebach G, Shamir R: **Modelling and analysis of gene regulatory networks.** *Nat Rev Mol Cell Biol* 2008, **9**:770-780.
18. Yeang CH, Ideker T, Jaakkola T: **Physical network models.** *J Comput Biol* 2004, **11**:243-262.
19. Lähdesmäki H, Shmulevich I, Yli-Harja O: **On Learning Gene Regulatory Networks Under the Boolean Network Model.** *Machine Learning* 2003, **52**:147-167.
20. Gat-Viks I, Tanay A, Shamir R: **Modeling and analysis of heterogeneous regulation in biological networks.** *J Comput Biol* 2004, **11**:1034-1049.
21. Gat-Viks I, Tanay A, Raijman D, Shamir R: **A probabilistic methodology for integrating knowledge and experiments on biological networks.** *J Comput Biol* 2006, **13**:165-181.
22. Friedman N: **Inferring cellular networks using probabilistic graphical models.** *Science* 2004, **303**:799-805.
23. Assmus HE, Herwig R, Cho KH, Wolkenhauer O: **Dynamics of biological systems: role of systems biology in medical research.** *Expert Rev Mol Diagn* 2006, **6**:891-902.
24. Gibbs JB: **Mechanism-based target identification and drug discovery in cancer research.** *Science* 2000, **287**:1969-1973.
25. Sander C: **Genomic medicine and the future of health care.** *Science* 2000, **287**:1977-1978.
26. Andrianantoandro E, Basu S, Karig DK, Weiss R: **Synthetic biology: new engineering rules for an emerging discipline.** *Mol Syst Biol* 2006, **2, 2006 0028.**
27. Basu S, Gerchman Y, Collins CH, Arnold FH, Weiss R: **A synthetic multicellular system for programmed pattern formation.** *Nature* 2005, **434**:1130-1134.
28. Murata T: **Petri nets: Properties, analysis and applications.** *Proceedings of the IEEE* 1989, **77**:541-580.
29. McMillan KL, Probst DK: **A technique of state space search based on unfolding.** *Formal Methods in System Design* 1995, **6**:45-65.
30. Roden DM, George AL Jr: **The genetic basis of variability in drug responses.** *Nat Rev Drug Discov* 2002, **1**:37-44.
31. Lieberman PM, Berk AJ: **The Zta trans-activator protein stabilizes TFIID association with promoter DNA by direct protein-protein interaction.** *Genes Dev* 1991, **5**:2441-2454.
32. Garg A, Aggarwal BB: **Nuclear transcription factor-kappaB as a target for cancer drug development.** *Leukemia* 2002, **16**:1053-1068.
33. Nielsen LL, Maneval DC: **P53 tumor suppressor gene therapy for cancer.** *Cancer Gene Ther* 1998, **5**:52-63.
34. Gilbert D, Heiner M: *From Petri Nets to Differential Equations - An Integrative Approach for Biochemical Network Analysis* Heidelberg: Springer Berlin 2006.
35. Kuffner R, Zimmer R, Lengauer T: **Pathway analysis in metabolic databases via differential metabolic display (DMD).** *Bioinformatics* 2000, **16**:825-836.
36. Sackmann A, Heiner M, Koch I: **Application of Petri net based analysis techniques to signal transduction pathways.** *BMC Bioinformatics* 2006, **7**:482.
37. Steggles LJ, Banks R, Shaw O, Wipat A: **Qualitatively modelling and analysing genetic regulatory networks: a Petri net approach.** *Bioinformatics* 2007, **23**:336-343.
38. Koch I, Heiner M: **Petri nets.** *Analysis of Biological networks* Junker BH, Schreiber F 2008, 139-180, [Pan Y, Zomaya AY (Series Editor)].
39. Nagasaki M, Doi A, Matsuo Y, Miyano S: **Computational Modeling of Biological Processes with Petri Net-Based Architecture.** *Bioinformatics Technologies* Springer Berlin HeidelbergChen YP 2005, 179-243.
40. Sackmann A, Formanowicz D, Formanowicz P, Koch I, Blazewicz J: **An analysis of the Petri net based model of the human body iron homeostasis process.** *Computational Biology and Chemistry* 2007, **31**:1-10.
41. Reddy VN, Liebman MN, Mavrovouniotis ML: **Qualitative analysis of biochemical reaction systems.** *Comput Biol Med* 1996, **26**:9-24.
42. Chaouiya C, Remy E, Ruet P, Thieffry D: **Qualitative Modelling in Genetic Networks: From Logical Regulatory Graphs to Standard Petri Nets.** *Lecture Notes in Computer Science* 2004, **3099**:137-156.
43. Grunwald S, Speer A, Ackermann J, Koch I: **Petri net modelling of gene regulation of the Duchenne muscular dystrophy.** *Biosystems* 2008, **92**:189-205.
44. Kielbassa J, Bortfeldt R, Schuster S, Koch I: **Modeling of the U1 snRNP assembly pathway in alternative splicing in human cells using Petri nets.** *Comput Biol Chem* 2009, **33**:46-61.
45. Matsuno H, Doi A, Nagasaki M, Miyano S: **Hybrid Petri net representation of gene regulatory network.** *Pac Symp Biocomput* 2000, 341-352.
46. Mcmillan KL: **A Technique of State-Space Search Based on Unfolding.** *Formal Methods in System Design* 1995, **6**:45-65.
47. Esparza J, Romer S, Vogler W: **An improvement of McMillan's unfolding algorithm.** *Formal Methods in System Design* 2002, **20**:285-310.
48. Shmulevich I, Dougherty ER, Kim S, Zhang W: **Probabilistic Boolean Networks: a rule-based uncertainty model for gene regulatory networks.** *Bioinformatics* 2002, **18**:261-274.
49. Fuller GN, Rhee CH, Hess KR, Caskey LS, Wang R, Bruner JM, Yung WK, Zhang W: **Reactivation of insulin-like growth factor binding protein 2 expression in glioblastoma multiforme: a revelation by parallel gene expression profiling.** *Cancer Res* 1999, **59**:4228-4232.
50. Zhang SQ, Ching WK, Ng MK, Akutsu T: **Simulation study in Probabilistic Boolean Network models for genetic regulatory networks.** *Int J Data Min Bioinform* 2007, **1**:217-240.
51. Jones N, Dumont DJ: **Tek/Tie2 signaling: new and old partners.** *Cancer Metastasis Rev* 2000, **19**:13-17.
52. Downes GB, Gautam N: **The G protein subunit gene families.** *Genomics* 1999, **62**:544-552.

53. Feistritzer C, Mosheimer BA, Sturn DH, Bijuklic K, Patsch JR, Wiedermann CJ: **Expression and function of the angiopoietin receptor Tie-2 in human eosinophils.** *J Allergy Clin Immunol* 2004, **114**:1077-1084.
54. Offermanns S, Mancino V, Revel JP, Simon MI: **Vascular system defects and impaired cell chemokinesis as a result of Galpha13 deficiency.** *Science* 1997, **275**:533-536.
55. Tew KD: **Redox in redux: Emergent roles for glutathione S-transferase P (GSTP) in regulation of cell signaling and S-glutathionylation.** *Biochem Pharmacol* 2007, **73**:1257-1269.
56. Koch I, Junker BH, Heiner M: **Application of Petri net theory for modelling and validation of the sucrose breakdown pathway in the potato tuber.** *Bioinformatics* 2005, **21**:1219-1226.
57. Matsuno H, Tanaka Y, Aoshima H, Doi A, Matsui M, Miyano S: **Biopathways representation and simulation on hybrid functional Petri net.** *In Silico Biol* 2003, **3**:389-404.
58. Valmari A: **The State Explosion Problem.** *Lecture Notes on Petri Nets I: Basic Models. Volume 1* Berlin: SpringerGoos G, Hartmanis J, van Leeuwen J 1998, 429-529.
59. Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T: **Cytoscape: a software environment for integrated models of biomolecular interaction networks.** *Genome Res* 2003, **13**:2498-2504.

doi:10.1186/1752-0509-4-15
Cite this article as: Karlebach and Shamir: **Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case.** *BMC Systems Biology* 2010 **4**:15.

Submit your next manuscript to BioMed Central and take full advantage of:

- **Convenient online submission**
- **Thorough peer review**
- **No space constraints or color figure charges**
- **Immediate publication on acceptance**
- **Inclusion in PubMed, CAS, Scopus and Google Scholar**
- **Research which is freely available for redistribution**

Submit your manuscript at
www.biomedcentral.com/submit

BioMed Central

5. Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy Based Approach

JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 19, Number 1, 2012
© Mary Ann Liebert, Inc.
Pp. 30–41
DOI: 10.1089/cmb.2011.0100

Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor–DNA Interactions with Expression Data: An Entropy-Based Approach

GUY KARLEBACH and RON SHAMIR

ABSTRACT

Models of gene regulatory networks (GRNs) attempt to explain the complex processes that determine cells' behavior, such as differentiation, metabolism, and the cell cycle. The advent of high-throughput data generation technologies has allowed researchers to fit theoretical models to experimental data on gene-expression profiles. GRNs are often represented using logical models. These models require that real-valued measurements be converted to discrete levels, such as on/off, but the discretization often introduces inconsistencies into the data. Dimitrova et al. posed the problem of efficiently finding a parsimonious resolution of the introduced inconsistencies. We show that reconstruction of a logical GRN that minimizes the errors is NP-complete, so that an efficient exact algorithm for the problem is not likely to exist. We present a probabilistic formulation of the problem that circumvents discretization of expression data. We phrase the problem of error reduction as a minimum entropy problem, develop a heuristic algorithm for it, and evaluate its performance on mouse embryonic stem cell data. The constructed model displays high consistency with prior biological knowledge. Despite the oversimplification of a discrete model, we show that it is superior to raw experimental measurements and demonstrates a highly significant level of identical regulatory logic among co-regulated genes. A software implementing the method is freely available at: http://acgt.cs.tau.ac.il/modent

Key words: algorithms, computational molecular biology.

1. INTRODUCTION

GENE REGULATORY NETWORKS (GRNs) play an important role in orchestrating the complex processes of life. An understanding of these networks and their behavior can elucidate complex processes of disease progression. The logical modeling approach describes a GRN and its dynamics as a set of entities that take discrete levels (e.g., active/inactive). Each entity's level is a function of the levels of certain other entities. Models can assume synchronous or asynchronous updates. The first logical models in biology were presented in the 1970s by Kauffman, Thomas, and colleagues (Glass and Kauffman, 1973; Thomas, 1973). For a review on logical models, see Karlebach and Shamir (2008). In recent years, mapping between logical values and continuous measurements has been revisited and empowered by high-throughput experimental data.

Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel.

Akutsu et al. (1999) proposed a polynomial algorithm that infers regulatory interactions from experimental data by finding for each gene a Boolean function that predicts its level with maximal accuracy. The inputs of that function are the levels of the gene's regulators. This algorithm requires that continuous expression data first be discretized into Boolean values (i.e., that each real value will be converted into a Boolean one), and then it selects the function and regulators that are in best agreement with the discretized data. A later extension allows each discretized sample to be associated with a continuous confidence value (Lähdesmäki et al., 2003), namely the reliability of each microarray profile (a vector of gene expression values) in the dataset. Akutsu et al. (2009) also studied the case in which only partial experimental data are available, and showed that learning the regulation functions in this setting is NP-complete.

Segal et al. (2003) developed a methodology that uses expression data for inferring regulatory functions formulated as decision trees: each node of the tree corresponds to a regulator, and the level of the regulatee is determined by traversing the tree from root to leaf, selecting a child at each node by comparing the regulator's continuous expression level to some threshold value. The algorithm of Segal et al. (2003) clusters genes into groups that have a similar expression pattern and assigns to every cluster its set of regulators.

Shamir and Tanay presented an efficient algorithm that assumes a monotone relationship between a transcription factor's (TF) continuous level, its affinity to a target gene and the strength of regulation, and uses this assumption to determine whether or not a target gene is activated. Since their algorithm requires TF-target affinities, they also suggested a method for inferring the affinity of a TF to its target genes (Shamir and Tanay, 2003).

The logical rules that govern gene expression were also studied for specific systems. Cox et al. (Cox, et al, 2007) created $\sim$300 artificial *Escherichia coli* promoters and analyzed their regulatory logic and other properties, using population-level expression data. The promoters were composed from target sites of two activators and two repressors. The authors observed that basal activity level and strength of induction for genes regulated by a single activator are not correlated. This shows that naive discretization of expression data is likely to produce mistakes.

It should be noted here that inferring discrete logic from continuous measurements depends on the activity threshold of the regulated gene; for example, in a Boolean model, the output should be 1 when the regulated gene's product is present in a sufficient amount to perform its role in the model, such as activating another gene. Thus, the threshold may be specific to the regulated gene. In addition, the closer a real expression value is to the threshold, the greater the chance that the mapping to a discrete value is incorrect.

Tsong et al. (2006) identified mating genes that were negatively regulated in *Saccharomyces cerevisiae* and positively regulated in an ancestral specie. They showed that the change in logic occurred in two steps: first, expression became independent of an activator, and second, then it came under the influence of a repressor. The changes occurred due to mutations in regulatory sequences, suggesting that changes in regulatory logic may have played a major role in modifying organism fitness during evolution.

Mayo et al. (2006) mutated regulatory sequences in the *lac* operon of *E. coli* and showed that certain mutations can change the logic. They also found that the logic is plastic (i.e., many mutations do not cancel a regulation but rather change its logic). This finding further supports the notion that changes in regulatory logic may have played an important role in evolution.

In this study, we show that given a model and discretized expression data that contain errors, the problem of correcting these errors using a minimal number of changes is computationally hard. This resolves an open problem stated in Dimitrova et al. (2010). In the next section, Section 2, we reformulate the problem probabilistically, and present an algorithm for constructing a Boolean model from partial prior knowledge and real-valued expression data aimed at providing a practical solution to the problem. In Section 3, we demonstrate the effectiveness of the method by using the algorithm to construct a logical model of the mouse embryonic stem cell network, and make some observations about the properties of the inferred network.

A software called ModEnt implementing the method is freely available at: http://acgt.cs.tau.ac.il/modent

2. METHODS

In a Boolean network model of a GRN, every gene is associated with an entity that can take the levels 0 and 1, which correspond to the inactive and active states of the gene, respectively. Gene regulation is described by assigning a Boolean function to each gene: the levels of a gene's regulators are the inputs of that gene's regulation function, and the effect of the regulator levels on the target gene's level is the output of the function. The model is synchronous: If time-series data are available, the levels of the regulators of each gene at time t-1 determine its level at time t according to its specific regulation logic. More formally, if

$e^t_{(1)}, \ldots, e^t_{(K)}$ are the (discretized) levels of the entities at time t, and if $r^i_1, \ldots, r^i_{n_i}$ are the regulators of entity i and the regulatory logic is f, then $f(e^{t-1}_{(r^i_1)}, e^{t-1}_{(r^i_2)}, \ldots, e^{t-1}_{(r^i_{n_i})}) = e^t_{(i)}$ for every i, t. If such data are not available, or if the time intervals between measurements are relatively long, a steady state can be assumed, in which case the regulation function produces an output at time t that agrees with its inputs at the same time t. For the sake of discussion, let us assume from now on that the data are steady state, though the same method applies to time-series data.

Comparison of a given model to discretized expression data may reveal discrepancies. A discrepancy occurs when the same inputs of a regulation function produce more than one output. For example, if a gene has two regulators that take level 0 in two profiles, but the gene itself has level 0 in one experiment and level 1 in the other, a discrepancy occurs. The source of the discrepancy can be noise or wrong assignment of discrete value to the target gene or to one of the regulators. Dimitrova et al. (2010) state the need for systematic handling of discrepancies as an open problem. When there are multiple discrepancies, we seek here the simplest explanation—the one that requires a minimal number of changes to the profiles of both the regulators and regulatees. We next show that this problem is NP-hard.

Theorem. Given the topology of a Boolean network model and binary expression profiles of the network's genes, resolving the discrepancies with a minimum number of changes is NP-hard.

Proof. We will show a reduction from the NP-complete problem Vertex Cover (Karp, 1972) to the decision problem: Given a GRN, a set of discretized microarray profiles and a number k, can all the discrepancies be resolved by at most k changes to the profiles?

Let (G(V, E), k) be the input for the Vertex Cover problem, where G(V,E) is an undirected graph and k is an integer between 1 and |V|. Construct a GRN as follows: For every vertex v in V, add a gene entity v to the GRN. For every edge e = (u,v) in E, define a new gene e_{uv} and identify the genes that correspond to u and v as the common regulators of e_{uv} (the regulatee). Figure 1a illustrates this construction. Hence, the original vertices are regulators (and are not regulated), and the new vertices correspond to regulatees. The set of microarray experiments will contain two profiles. In the first the levels of all the genes will be 0. In the second, the levels of all the regulators will be 0 and the levels of all the regulatee genes will be 1 (Fig. 1b). Since the levels of the regulators are the same in both profiles, and the levels of the regulatees are not, there are discrepancies. Clearly, this reduction can be performed in polynomial time.

Suppose there is a vertex cover S of size at most k. For every vertex u that belongs to S, change the level of the corresponding gene in the second experiment to 1. Since every regulatee corresponds to an edge in G,

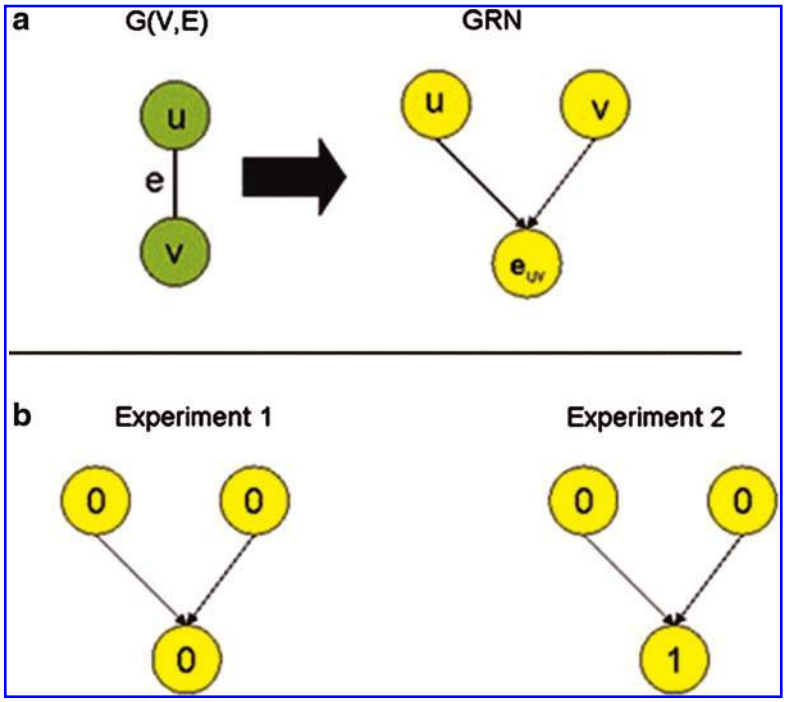


FIG. 1. Reduction from Vertex Cover to resolution of discrepancies in microarray expression data with respect to a gene regulatory network (GRN). **(a)** The graph that is the input for Vertex Cover is shown in green, and the resulting GRN is shown in yellow. **(b)** The Boolean values of every entity in the two microarray experiments.

and its regulators are vertices that are adjacent to that edge, for every regulatee at least one of its regulators changes in experiment 2. Therefore, all the discrepancies are resolved by at most k changes.

Now, assume conversely that there are k changes that resolve all discrepancies. If after the changes there is a regulatee that has the same level in the two profiles (i.e., its level was changed by the solution) and each of its regulators has the same level in the two profiles, we will restore that regulatee's level to 0 in profile 1 and 1 in profile 2, and change the level of one of its regulators in profile 1. This does not increase the total number of changes: The regulatee has regained the levels it had before any changes took place, which cancels at least one change, and a single change was made to a regulator's level. We repeat this for every regulatee that changes its levels from the original levels assigned by the reduction, and thus obtain a set of at most k changes—all of which are in regulator levels—with no discrepancies. Now define a set S that contains the nodes corresponding to every regulator that has different values in the two profiles. This set is of size at most k. For every edge in G, there is a vertex in S that is adjacent to it, because every regulatee has at least one regulator that has different levels in the two experiments. Therefore, S is a vertex cover.

It remains to show that the problem is in NP. Given k changes, we perform them and check in polynomial time whether there are any discrepancies left. ■

We now approach the problem from a different direction: we return to the real-valued expression profiles, and instead of discretizing them, a process that may cause discrepancies that are difficult to resolve, we take a probabilistic approach. We interpret the real-valued profiles probabilistically, select a set of TF-target interactions that minimizes the total entropy, and use the selected topology and the probabilistically-interpreted profiles to resolve discrepancies. Our algorithm is outlined in Figure 2.

Following is a detailed description of the algorithm. We interpret a vector of continuous values as a probability distribution over all possible Boolean vectors of the same dimension. In other words, instead of creating a single Boolean vector with probability 1 for a given continuous vector, we create all possible Boolean vectors of the same dimension, and assign each such vector a probability. The probabilities are chosen as follows: First, normalize the continuous expression values of every gene to have mean 0 and standard deviation 1.5 (a value determined empirically). Second, after normalization, set the probability that a single (one-dimensional) real value c corresponds to the Boolean value 1 to $\lambda(c) = \frac{1}{1+e^{-c}}$ (the logistic function with parameter value c). The probability that a real-valued vector $\overline{c}$ corresponds to a specific Boolean vector $\overline{b}$ then becomes

$$p(\overline{b} \mid \overline{c}) = \prod_{i|b_i=1} \frac{1}{1+e^{-c_i}} \prod_{i|b_i=0} \left(1 - \frac{1}{1+e^{-c_i}}\right),$$

where c_i (b_i) is the value of the i^{th} entry of $\overline{c}$ ($\overline{b}$). Note that by setting the standard deviation value for all the genes one avoids using any parameters in the logistic function.

Given a continuous dataset of n i.i.d. profiles, the probability of seeing the Boolean vector $\overline{b}$ in this dataset is:

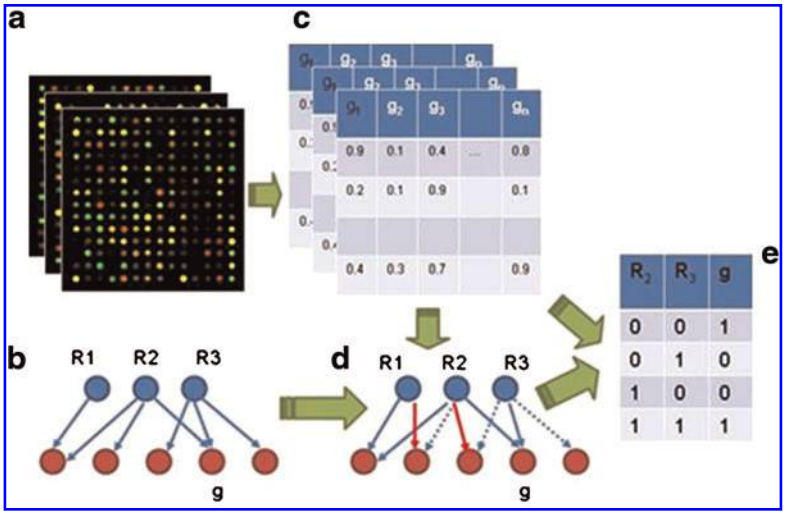


FIG. 2. An outline of our algorithm. The input consists of real valued expression profiles **(a)** and a set of putative regulations of genes by transcription factors **(b)**. The expression profiles are interpreted probabilistically **(c)** and used for determining the topology of the network by selecting a set of regulators that minimize the entropy **(d)**. In this process, some putative interactions are rejected (dashed arrows), and some new interactions are added (red arrows). The network's regulation functions **(e)** are determined using the probabilistically interpreted expression profiles and the inferred topology.

$$P(\bar{b}) = \frac{1}{n} \sum_{\bar{c}^j \in profiles} p(\bar{b} \mid \bar{c}^j) \qquad (^*)$$

In other words, for each Boolean vector, the probabilities that each continuous vector corresponds to it are averaged. In practice, the samples may not be i.i.d, but that assumption is made for the sake of this analysis.

With the probability distribution over all Boolean vectors at hand, information theory can be used to evaluate different topologies of the network. Suppose we know which of the genes are transcription factors (TFs) and assume that all regulators are TFs. Denote by $H^C(x|Y_x)$ the conditional entropy for a gene x and a set Y_x of regulators as computed using continuous data. We use this notation in order to stress that the conditional entropy is a function of continuous values—a fact that will be used by our algorithm. Select for every gene x the set Y_x of regulators that gives the best $H^C(x|Y_x)$ score among all sets of TFs.

Since in practice a larger set of regulators will tend to score better than a smaller one, a threshold that will separate significant improvement from insignificant improvement is needed: when increasing the set of regulators, any improvement less than the threshold will be considered insignificant. This threshold can be estimated empirically by computing the average and standard deviation of the improvement in entropy that occurs when non-regulator genes are assigned as regulators. Improvement that surpasses the average by 3 standard deviations will be interpreted as non-random. We refer to this threshold value as τ.

After the network structure is constructed, steepest descent can be used for decreasing the entropy: given the set Y_x minimizing the score $H^C(x|Y_x)$ for every gene x, perform steepest descent on the score $\sum_{x \in genes} H^C(x|Y_x)$, i.e. on the total entropy of the network. We compute the derivative of the total entropy function with respect to each gene and regulator and change their profiles in the direction of the gradient, and repeat this iteratively until the change in entropy is very small.

If we had discrete profiles and change a level from 0 to 1, the value of the conditional entropy will also change. Since we do not discretize, we have continuous profiles, and every function $H^C(x|Y_x)$ is a function of continuous values. Therefore, $H^C(x|Y_x)$ will change with every change of one of its continuous parameters. Given the real level c_{ij} of gene i at profile j, the partial derivative of the total entropy with respect to c_{ij} can be computed exactly. By the chain rule for conditional entropy, we have:

$$\frac{\partial}{\partial c_{ij}} H^C(x|Y_x) = \frac{\partial}{\partial c_{ij}} H^C(x, Y_x) - \frac{\partial}{\partial c_{ij}} H^C(Y_x)$$

We show how to compute $H^C(Y_x)$. The computation of $H^C(x,Y_x)$ only differs in indices and is omitted:

$$\begin{aligned}
&\frac{\partial}{\partial c_{ij}} H^C(Y_x) = \frac{\partial}{\partial c_{ij}} \sum_{Y_x} P(Y_x) \log P(Y_x) = \\
&= -\sum_{Y_x} \left[\frac{\partial}{\partial c_{ij}} P(Y_x) \right] \log P(Y_x) - \sum_{Y} P(Y_x) \left[\frac{\partial}{\partial c_{ij}} \log P(Y_x) \right] = \\
&= -\sum_{Y_x} \left[\frac{\partial}{\partial c_{ij}} P(Y_x) \right] \log P(Y_x) - \sum_{Y} \frac{\partial}{\partial c_{ij}} P(Y_x)
\end{aligned}$$

where the sum is over all Boolean values of the vector Y_x. The probability of a specific Boolean value of the vector Y_x is given by (*), and for $\frac{\partial P(Y_x)}{\partial c_{ij}}$, we have the following sum:

$$\frac{1}{n} \cdot \frac{\partial}{\partial c_{ij}} \sum_{\bar{c}^j \in profiles} p(Y_x \mid \bar{c}^j) = \frac{1}{n} \cdot \frac{\partial}{\partial c_{ij}} p(Y_x \mid \bar{c}^j)$$

where the latter equality is due to the fact that the derivative is 0 for profiles other than the j^{th} profile, which contains c_{ij}. Now in order to find the latter derivative, we recall that it is a product of the logistic function λ or (1- λ), and only one of the factors is $\lambda(c_{ij})$ or (1- $\lambda(c_{ij})$). For example, if Y is the vector (1,1,…,1), the derivative would be:

$$\frac{\partial}{\partial c_{ij}} p(Y \mid \bar{c}^j) = \left(\prod_{c_{kj} \in \bar{c}^j,\, k \neq i} \lambda(c_{kj}) \right) \cdot \frac{\partial}{\partial c_{ij}} \lambda(c_{ij}) = \left(\prod_{c_{kj} \in \bar{c}^j,\, k \neq i} \lambda(c_{kj}) \right) \cdot \lambda(c_{ij}) \cdot (1 - \lambda(c_{ij}))$$

Now we can compute the gradient of the function $\sum_{x \in genes} H^C(x \mid Y_x)$.

Every iteration, we make a step of size 1 in the opposite direction of the gradient, until the change in entropy is very small. Changing the real value has the effect of reducing the entropy, which reflects the discrepancies.

After steepest descent converges, a truth table (i.e., regulation logic) needs to be assigned for each gene. First, note that the probability to observe a certain line in the truth table, with output x = α and input $Y = \overline{\beta}$, is the value $P(x = \alpha, Y = \overline{\beta})$, which is computed as discussed above. Second, for every regulator there should be at least one input in which changing that regulator's value will change the output of the regulation function. If the latter property holds, the regulation function is said to be *non-redundant*. We use a simple branch and bound algorithm to find a consistent regulation function with maximum probability. Given a partial choice of outputs, a bound on the maximal probability of every non-redundant function that contains this choice can be obtained by completing it with the most probable output choices. An initial bound is obtained by picking the maximal probability of a non-redundant function from the following set: the function F* that is formed from the highest probability choice for every output, and the functions that are formed from changing one of the outputs of F* (we set the initial bound to zero if the set does not contain non-redundant functions).

We implemented the method in a program called ModEnt (for entropy-based modeling). The implementation is freely available at: http://acgt.cs.tau.ac.il/modent

3. A CASE STUDY

The GRNs that regulate differentiation in mammalian embryonic stem cells (ESCs) control a fascinating process whose understanding can lead to far-reaching breakthroughs in medicine, making them the subject of extensive research (Chickarmane et al., 2006; Novershtern et al., 2011; Xu et al., 2010; Zhou et al., 2007). We used our method to construct a logical model of mouse ESC GRN by integrating putative TF-DNA interactions with expression data. More specifically, we combined the core20 network that is available in the Integrated Stem Cell Molecular Interaction Database (MacArthur et al., 2009), the mouse ESC network of Zhou et al. (2007), and the expression data of Ivanova et al. (2006) to obtain 728 reported putative interactions between 25 potential regulators and 236 target genes. The number of regulators per gene varied between 1 and 14 (mean 3.15). The number of regulated genes per TF varied between 1 and 170 (mean 9.24). In addition, we used 70 microarray profiles from Ivanova et al. (2006).

For each gene x, a subset of its putative regulators Y was selected such that the conditional entropy $H^C(x|Y)$ was minimized (a steady state was assumed for every profile). Since not all the genes had the same number of reported interactions, addition of more regulators was allowed in case all of the reported regulators were selected. When computing $H^C(x|Y)$, we excluded those profiles in which the regulatee was knocked-out.

The maximal number of regulators for a gene in the set of reported interactions was 14. Thus, for each gene, we tested every set of regulators of size ≤ 14 out of the total 25 regulators. A set S_1 of size n was preferred over a set S_2 of size $m < n$ if the difference in conditional entropy was greater than $(n\text{-}m) \cdot \tau$, where $\tau = 0.00775244$ is the value of the threshold defined in the previous section.

Our reconstructed model contained 449 edges (interactions), of which 298 belong to the published interaction set. The appendix contains the network topology, list of regulation functions, and list of cohorts (the appendix is available at the authors' website: http://acgt.cs.tau.ac.il/modent). Since we picked regulators to minimize the discrepancies with expression data, whereas the reported interactions were based on binding assays, we expected to see a different distribution of regulator-regulatee edges, and this was indeed the case (Fig. 3). Some of these differences are attributed to the false positives and false negatives in the reported interactions, although a true positive will not be inferred without proper expression data. For example, if we know that R regulates G, but in all the available expression profiles R is knocked down, we will not be able to use our knowledge in a model. Similarly, if the reported interactions are insufficient to produce a regulation function that satisfactorily predicts the target gene's level, unreported interactions need to be selected. The lower frame of Figure 3 shows that, for the ESC network, often one of the latter cases applied. Figure 4 illustrates the number of common target genes for each pair of regulators. Figure 5 illustrates the cohorts and their regulators; as can be seen, the TFs Pou5f1 and Sox2 regulate the two largest cohorts, while each of Nanog, Esrrb, Tcf7, and Etv5 regulate cohorts of intermediate size. Four cohorts are each regulated by four regulators, including Pou5f1, Rnf2, Zfp281, Dax1, Etv5, Sox2, Nr5a2,

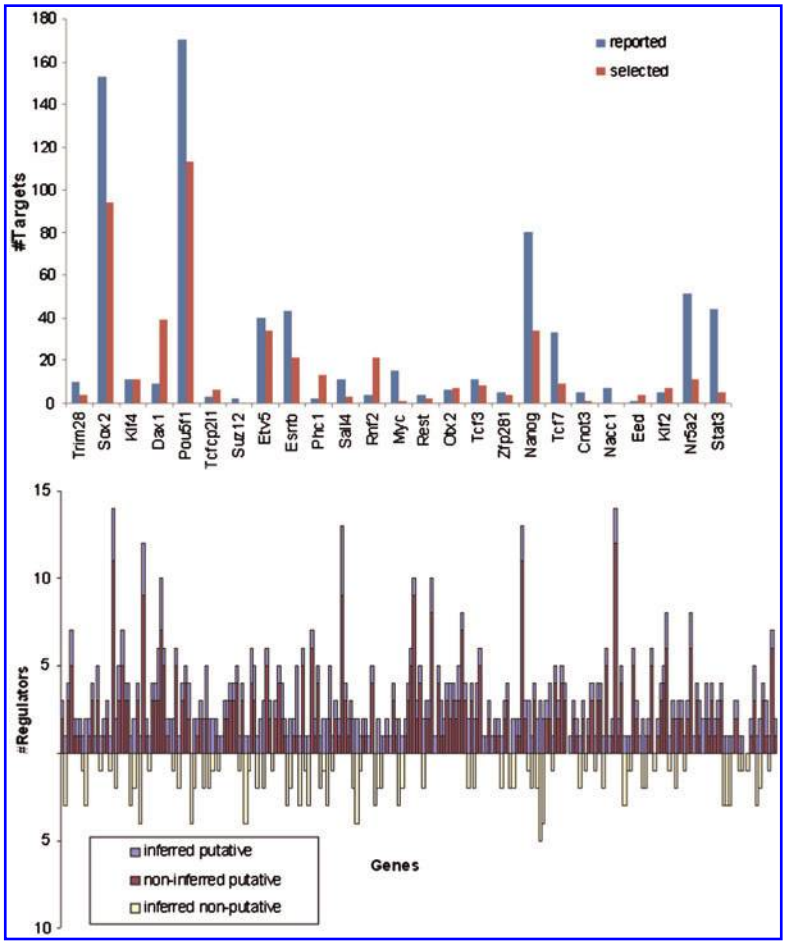


FIG. 3. Comparison of reported interactions and interactions selected from expression profiles. Top frame: The number of reported targets compared to the number of selected targets for every transcription factor (TF). Bottom frame: For every gene, the number of reported TFs that were selected, the number of reported TFs that were not selected, and the number of unreported TFs that were selected. For clarity, the gene names were omitted.

Phc1, and Otx2. It is reasonable to assume that genes that have more regulators are subjected to a more complex regulatory program, and therefore may have roles in more specific contexts compared to other genes; a better understanding of this network's behavior requires analysis of the dynamics involved.

We turned to the dataset of Young and colleagues, found in Marson et al. (2008), to assess the quality of the selected interactions. In this study, ChIP-seq technique was used to measure binding of five TFs: Pou5f1, Sox2, Nanog, Tcf3, and Suz12, to regulatory regions of 200 genes in our network. The dataset corresponds to a 200×5 Boolean matrix M, in which the entry in the i^{th} row and the j^{th} column is 1 if TF j ($1 \le j \le 5$) binds gene i according to the ChIP-seq data. Now if S is the set of regulators of gene i in the reconstructed model, we define the similarity between S and row i in the matrix M as $\frac{\sum_{j \in S} M[i,j]}{|S|}$. The average similarity between the regulators of a gene in the reconstructed model and the matrix M was 0.63. To assess the significance of this result, we randomly permuted each row in the matrix M independently and computed the average similarity. By repeating the randomization 10,000 times we conclude that this overlap value is significant at p-value $< 10^{-4}$ (Fig. 6). Figure 7 compares the number of common regulators and the number of different regulators for each gene in the inferred model and in Marson et al. (2008).

We call a set of genes that have the exact same regulators a *cohort*. We wanted to test whether genes that share the same set of regulators tend to have the same regulatory logic. We define similarity between two regulation functions as the fraction of inputs that produce identical outputs. The average similarity in a cohort is the average similarity between pairs of genes in that cohort. In order to eliminate genes whose levels may have been incorrectly modeled, genes with truth tables that were on average less than 50% similar to all other genes in the cohort were excluded. This filtering left 144 out of 184 genes that belong to cohorts, excluding no more than a third of the genes in any cohort. The average percentage of logic similarity that was obtained among the remaining genes in each cohort is 84%. To assess the significance, we permuted edges in the network of Young and colleagues, found in Marson et al. (2008), by conducting a long series of edge swaps, a

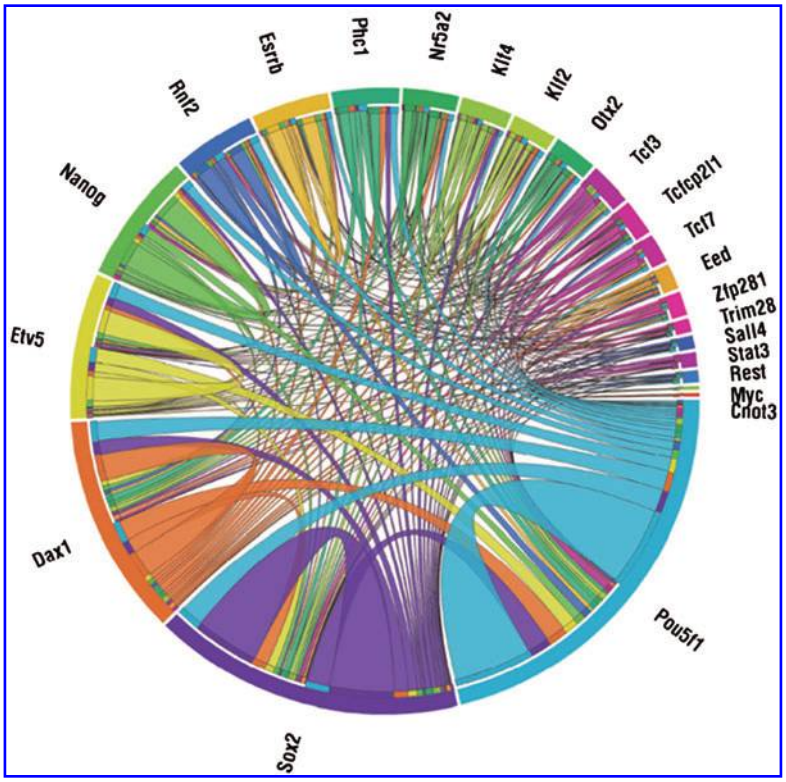


FIG. 4. The number of common target genes for each pair of transcription factors (TFs). The colored arcs along the circumference indicate the inferred targets of TFs where, for clarity, each TF is represented by a different color. The internal arcs connect two groups of targets of two TFs and are colored by one of the two colors of the TFs. The size of an internal arc between two TFs is proportional to the number of common targets they share. An internal arc from a TF to itself indicates the total number of target genes of that TF. The figure was generated using Circos.

process that preserves the degree of each node, and then reconstructing the model given the permuted network (Ulitsky, et al, 2010). For every permutation, the average percentage of cohort similarity and the number of excluded genes were computed as described above, and compared to the values that were obtained for the model. We considered a solution as scoring better only if (i) the similarity was equal or higher and (ii) the number of genes that were included in cohorts was equal or higher. Both conditions must be taken into account, since otherwise similarity is maximized by reducing the number and size of cohorts through gene exclusion or edge swap. Repeating the process 10^5 times showed that the logic similarity was significant at p-value $< 10^{-5}$. Figure 8 compares the scores of 1000 random permutations and the score obtained by the real topology. In order to make sure that our exclusion scheme does not generate any biases, we repeated the test by applying criteria (i) and (ii) without excluding genes from cohorts and obtained a p-value of $1.1 \cdot 10^{-4}$. In order for the simulation to run sufficiently fast, a speed-up of the selection procedure was used in which

FIG. 5. Cohorts and their sets of regulators. Each cohort is represented by a trapezoid, and the corresponding set of regulators is represented by an ellipse that is connected to its cohort's trapezoid. The names of the genes or regulators that belong to each set are given inside the shapes.

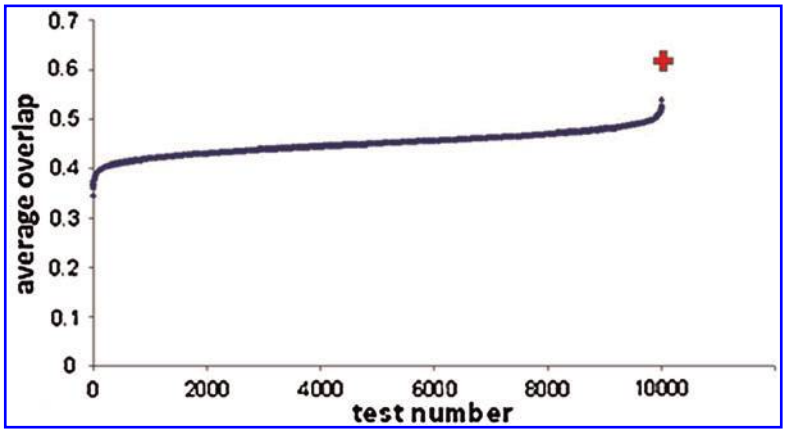


FIG. 6. Comparison of inferred and measured transcription factor (TF)–gene interactions. The average similarity of TF–gene interactions between the inferred network and the ChIP-seq interactions reported by Young et al., found in Marson et al. (2008), for five TFs was computed for randomized and real datasets. The figure shows the values for 10,000 random permutations of the Chip-seq dataset of Marson et al. (2008) and for the real dataset. Each blue dot represents the values obtained for one permutation. The red plus sign corresponds to the score of the real dataset.

regulators are added incrementally to the regulators set as long as the entropy improves significantly. A similar speedup was used in Hashimoto et al. (2003), using discrete data and a different score.

Figure 9 shows the similarity of regulation function of all the genes in the network. The network is seen to contain cohorts with highly similar regulation functions. There are some similarities in the regulation functions of genes that belong to different cohorts (depicted as edges that cross the interior of the circle), due perhaps to reuse of certain ''regulatory logic motifs'' in gene regulation (Milo et al., 2002).

These results are in line with the common assumption that regulatory logics within cohorts are similar, and also with the more general observation that networks contain ''reusable components'' (Milo et al., 2002). The term ''reusable components'' means that regulatory elements can be used similarly for different parts of the network. Segal et al. (2003) based their method on this assumption. Since our method does not impose any constraints on logic within cohorts and we still observe a high level of identical regulation within cohorts, we conclude that the reconstructed model is reasonably reliable.

4. DISCUSSION

We have presented an algorithm for constructing a logical model and resolving discrepancies between the model and experimental data. After demonstrating that the general problem of resolving discrepancies is computationally hard and there is probably no efficient algorithm that solves it, we adopted a proba-

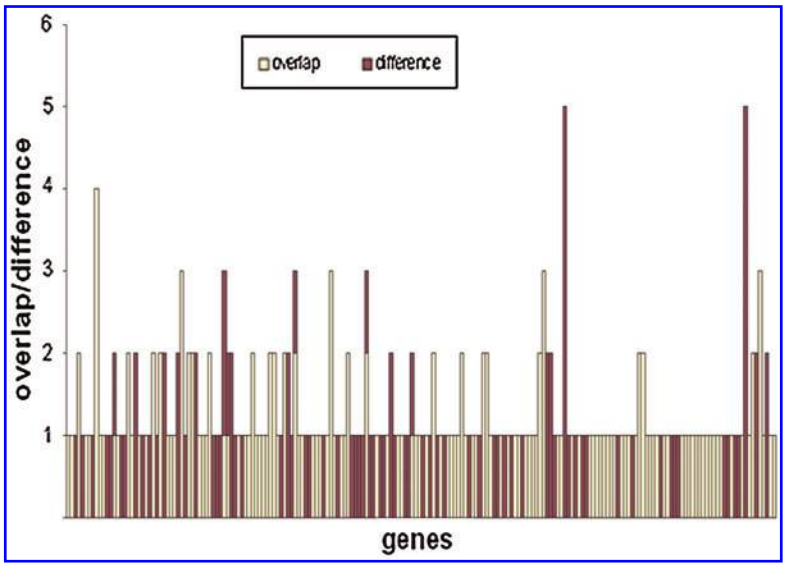


FIG. 7. The number of common transcription factors (TFs) and different TFs for each gene in the inferred network and the dataset of Marson et al. (2008).

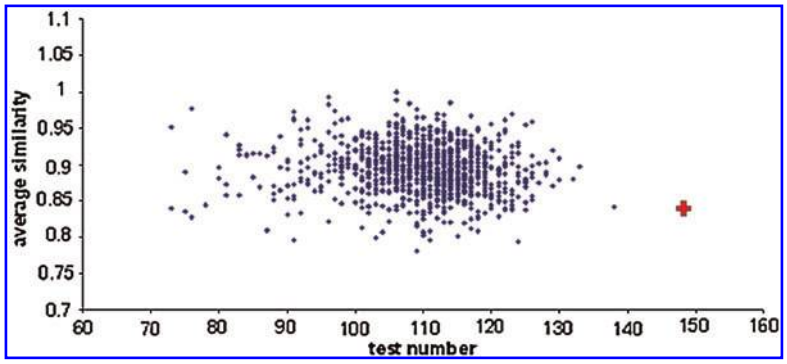


FIG. 8. Logic similarity and cohort sizes for randomized and real networks. The figure shows the values for 1000 random permutations of the embryonic stem cell (ESC) network and for the real topology. Each blue dot is a value obtained for one permutation, where the *y*-coordinate is the within-cohort similarity and the *x*-coordinate is the number of genes in cohorts of size at least 2. The red plus sign corresponds to the score of the real topology.

bilistic approach to the network reconstruction problem. We developed an algorithm that uses reported interactions and expression data to select a set of regulators for each gene, and that resolves discrepancies in the resulting logical model. We used our algorithm to construct a logical model of the mouse ESC GRN. The model supports the notion that genes which share the same regulators have similar regulatory logics.

Unlike Dimitrova et al. (2010) and other discretization methods, our algorithm refrains from directly discretizing the data, thereby avoiding the errors that are inherent to this process and the intractability of minimizing them. Instead, it assigns a probability to each discrete value and adjusts the input real values to improve model consistency, as reflected by the conditional entropy. Other methods that use information theory for selecting regulators discretize the data, but do not provide a means of discrepancy resolution (Liang et al., 1998; Lopes et al., 2008). Our algorithm can be applied when using only expression profiles as input, but can also utilize information on putative regulations (e.g., from ChIP-chip or ChIP-seq data) to improve the prediction. Given a set of such putative interactions, it can reject those that lack support in expression data. A disadvantage of our method is that we normalize the expression profiles of all the genes using the same parameters, which may be inferior to preprocessing using gene-specific parameters. Another disadvantage is that the inferred discrete logic is a necessary oversimplification of the biological reality.

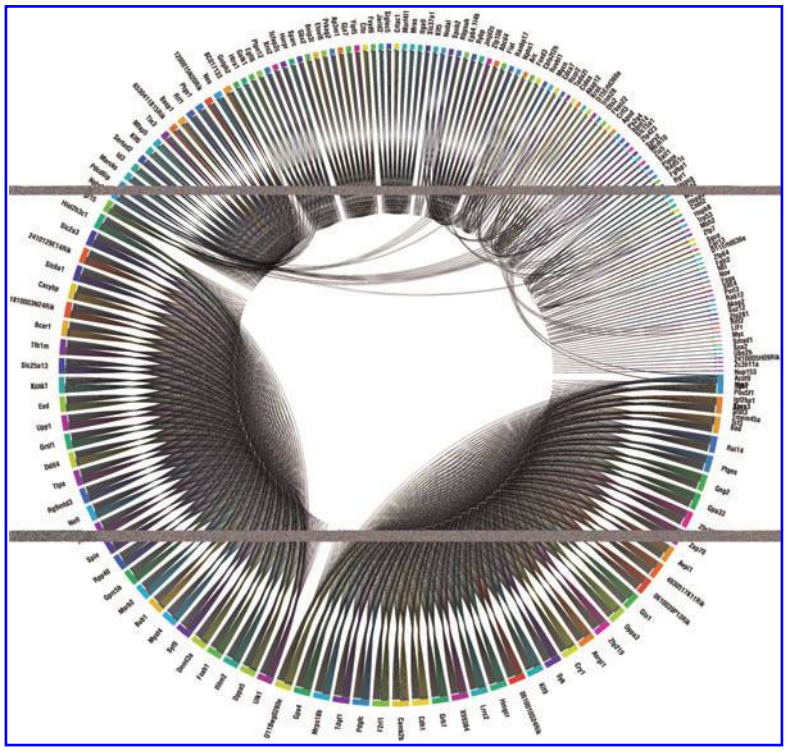


FIG. 9. The similarity of regulation functions for every pair of genes that have at least one common regulator. Similarity is measured as the fraction of identical outputs. Similarity of regulation functions of genes that have a different number of regulators are also compared: each output of the function with less regulators is compared to several outputs of the function with more regulators. For clarity, only genes that share $>75\%$ similarity are connected. The figure was generated using Circos.

Finally, as the general problem is computationally hard, we resort to heuristics, and at least for some instances of the problem, we may not find the optimal solution.

At this point, it is natural to ask whether one can obtain logical models that are sufficiently accurate. A model that contains even a small number of errors can produce erroneous predictions. Theoretical examples in which a small error in the model has a large impact on its predictions are easily found (Lorenz, 1993). Further research is required to determine whether domain-specific algorithms can produce accurate logical models. Another approach to the problem is developing algorithms that analyze a model without trying to resolve all the ambiguities in it (Karlebach and Shamir, 2010).

The probabilistic approach to discretization that we describe could be applied to other purposes in bioinformatics. Because discretization of expression data is used in methods such as clustering (Ben-Dor et al., 1999; Koyuturk et al., 2004) and feature selection (Saeys et al., 2007; Akutsu and Miyano, 2001), resolving discrepancies in discretized expression data can be performed as a preliminary step.

We intend to proceed with the analysis of the mouse ESC model, including its dynamic behavior and the effect of perturbations. Our reconstruction algorithm should be tested on other datasets in order to further characterize its advantages and disadvantages. Reconstruction of accurate logical models and their use for generating useful predictions are objectives that require further exploration.

ACKNOWLEDGMENTS

We would like to thank Qing Zhou, Wing Wong, and Avi Ma'ayan for help and advice with data that were used in this work, and Hershel Safer for helpful comments on this manuscript. This study was supported in part by a grant to the APO-SYS consortium from the European Community's Seventh Framework Programme. G.K. was supported in part by a fellowship from the Edmond J. Safra Bioinformatics Program at Tel Aviv University, Israel.

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Akutsu, T., and Miyano, S. 2001. Selecting informative genes for cancer classification using gene expression data. *Proc. IEEE-EURASIP Workshop Nonlinear Signal Image Process.* 3–6.

Akutsu, T., Tamura, T., and Horimoto, K. 2009. Completing networks using observed data. *Algorithmic Learn. Theor.* 126–140.

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

Ben-Dor, A., Shamir, R., and Yakhini, Z. 1999. Clustering gene expression patterns. *J. Comput. Biol.* 6, 281–297.

Chickarmane, V., Troein, C., Nuber, U.A., et al. 2006. Transcriptional dynamics of the embryonic stem cell switch. *PLoS Comput. Biol.* 2, e123.

Cox, R.S., 3rd, Surette, M.G., and Elowitz, M.B. 2007. Programming gene expression with combinatorial promoters. *Mol. Syst. Biol.* 3, 145.

Dimitrova, E.S., Licona, M.P., McGee, J., et al. 2010. Discretization of time series data. *J. Comput. Biol.* 17, 853–868.

Glass, L., and Kauffman, S. 1973. The logical analysis of continuous, non-linear biochemical control networks. *J. Theor. Biol.* 39, 103–129.

Hashimoto, R., Dougherty, E., Brun, M., et al. 2003. Efficient selection of feature sets possessing high coefficients of determination based on incremental determinations. *Signal Process.* 83, 695–712.

Ivanova, N., Dobrin, R., Lu, R., et al. 2006. Dissecting self-renewal in stem cells with RNA interference. *Nature* 442, 533–538.

Karlebach, G., and Shamir, R. 2010. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case. *BMC Syst. Biol.* 4, 15.

Karlebach, G., and Shamir, R. 2008. Modelling and analysis of gene regulatory networks. *Nat. Rev. Mol. Cell Biol.* 9, 770–780.

Karp, R. 1972. Reducibility Among Combinatorial Problems. *Complexity of Computer Computations*. 85–103.
Koyuturk, M., Szpankowski, W., and Grama, A. 2004. Biclustering gene-feature matrices for statistically significant dense patterns. *Proc. IEEE Comput. Syst. Bioinformatics Conf. (CSB'04)* 480–484.
Krzywinski, M., Schein, J., Birol, I., et al. 2009. Circos: an information aesthetic for comparative genomics. *Genome Res.* 19, 1639–1645.
Lähdesmäki, H., Shmulevich, I., and Yli-Harja. 2003. On learning gene regulatory networks under the Boolean network model. *Machine Learn.* 52, 147–167.
Liang, S., Fuhrman, S., and Somogyi, R. 1998. Reveal, a general reverse engineering algorithm for inference of genetic network architectures. *Pac. Symp. Biocomput.* 18–29.
Lopes, F.M., Martins, D.C., Jr., and Cesar, R.M., Jr. 2008. Feature selection environment for genomic applications. *BMC Bioinform.* 9, 451.
Lorenz, E.N. 1993. *The Essence of Chaos*. University of Washington Press, Seattle.
MacArthur, B., Ma'ayan, A., and Lemischka, I. 2009. Systems biology of stem cell fate and cellular reprogramming. *Nat. Rev. Mol. Cell Biol.* 10, 672–681.
Marson, A., Levine, S.S., Cole, M.F., et al. 2008. Connecting microRNA genes to the core transcriptional regulatory circuitry of embryonic stem cells. *Cell* 134, 521–533.
Mayo, A.E., Setty, Y., Shavit, S., et al. 2006. Plasticity of the cis-regulatory input function of a gene. *PLoS Biol.* 4, e45.
Milo, R., Shen-Orr, S., Itzkovitz, S., et al. 2002. Network motifs: simple building blocks of complex networks. *Science* 298, 824–827.
Novershtern, N., Subramanian, A., Lawton, L.N., et al. 2011. Densely interconnected transcriptional circuits control cell states in human hematopoiesis. *Cell* 144, 296–309.
Saeys, Y., Inza, I., and Larranaga, P. 2007. A review of feature selection techniques in bioinformatics. *Bioinformatics* 23, 2507–2517.
Segal, E., Shapira, M., Regev, A., et al. 2003. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat. Genet.* 34, 166–176.
Shamir, R., and Tanay, A. 2003. Modeling transcription programs: inferring binding site activity and dose-response model optimization. *Proc. RECOMB'03.*
Thomas, R. 1973. Boolean formalization of genetic control circuits. *J. Theor. Biol.* 42, 563–585.
Tsong, A.E., Tuch, B.B., Li, H., et al. 2006. Evolution of alternative transcriptional circuits with identical logic. *Nature* 443, 415–420.
Ulitsky, I., Laurent, L.C., and Shamir, R. 2010. Towards computational prediction of microRNA function and activity. *Nucleic Acids Res.* 38, e160.
Xu, H., Schaniel, C., Lemischka, I.R., et al. 2010. Toward a complete in silico, multi-layered embryonic stem cell regulatory network. *Wiley Interdiscip. Rev. Syst. Biol. Med.* 2, 708–733.
Zhou, Q., Chipperfield, H., Melton, et al. 2007. A gene regulatory network in mouse embryonic stem cells. *Proc. Natl. Acad. Sci. U.S.A.* 104, 16438–16443.

Address correspondence to:
Dr. Ron Shamir
Blavatnik School of Computer Science
Tel Aviv University
Tel Aviv 69978, Israel

E-mail: rshamir@tau.ac.il

6. A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets

A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets

Guy Karlebach[1] and Ron Shamir[1]

[1] The Blavatnik School of Computer Science, Tel Aviv University
Tel Aviv, 69978 Israel
rshamir@tau.ac.il

Abstract. Petri nets are a modeling formalism for concurrent systems. Given a Petri net and its initial state, determining if a target state or a set of states is reachable is a fundamental problem. McMillan's unfolding algorithm constructs a compact representation of a Petri net's state space. However, the algorithm is limited in practice to solving relatively small reachability problems, due to the computational resources required. We developed a Monte-Carlo algorithm based on McMillan's unfolding for solving the reachability problem on Petri nets. The algorithm repeatedly constructs random prefixes of the state space representation, thereby avoiding some of the computational problems that arise when the full representation is constructed. Our tests show that the randomized algorithm can solve problems of size greater than 100 within seconds, and it is faster than the deterministic algorithm by several orders of magnitude.

Keywords: Petri net, Unfolding, Monte-Carlo Algorithms

1 Introduction

A Petri net (1) is a modeling formalism that was developed in the 1960's for modeling of concurrent systems and has been studied extensively since. It has broad applications in numerous areas, including software design and verification, discrete process control, workflow management and diagnosis (2-6). A key difficulty in utilizing Petri nets is the explosion of the state space, which prevents utilizing the model for large or even moderate problems. McMillan (7) described a seminal method called unfolding to explore the state space more efficiently, and Esparza, Romer and Vogler (8) improved unfolding further. Still, in some cases the size of problems that can be handled is restricted, and methods were developed to improve its performance (9)(10)(11). In this study we describe a new way to speed up the exploration significantly, while sacrificing some accuracy. Our method is a randomized (Monte Carlo) relaxation of the unfolding method. We show that it can solve certain reachability problems several orders of magnitude faster.

We start by an informal introduction of the problem and of our method. A more formal representation is provided in the next section. A Petri net is specified by a directed bipartite graph that is composed of two types of nodes: places and transitions. The places represent the state of the network – each place contains a number of tokens that may change as a result of the network's dynamic behavior.

Transitions represent events that generate the network's dynamic behavior – a transition consumes tokens from places from which it has incoming edges and produces tokens into places to which it has outgoing edges. One of the notable properties in which Petri nets differ from traditional automata is their non-determinism: Petri nets describe concurrent events, and therefore do not impose synchronization or a specified order of occurrence on events. Testing properties such as the existence of deadlocks, reachability of a given state or the infinite accumulation of tokens in a place are some examples for computations that can be performed using a Petri net model.

A key computational problem in Petri net models and other discrete models (12) is how to check reachability when the state space is too large to be represented explicitly: a network that has N Boolean entities has a state space of size 2^N (the state of the network is described by the N-long vector describing the level of each of its entities). One approach for dealing with this problem is to create an implicit representation of the state space in the form of a new graph called a branching process (BP) (13): a directed, acyclic graph that captures the causality and concurrency relationships between network events. This technique, called unfolding, often provides relatively modest state space representations and has been studied extensively (reviewed in a recent book by Esparza and Helijanko (14)). McMillan presented an algorithm that generates a *finite* prefix of the branching process graph by introducing the concept of a cutoff point – a terminal node from which the BP graph need not be further extended (7). He showed that this prefix graph captures all the reachable states in the full (typically infinite) BP. Subsequently, Esparza, Romer and Vogler generalized the original cutoff criterion, gave an alternative cutoff criterion, and showed that the prefix it produces is always equal or smaller than McMillan's. Moreover, they demonstrated that in some cases it produces a graph of polynomial size while McMillan's criterion produces one of exponential size (8).

In this work we present a new randomized (Monte Carlo) method that uses unfolding to test reachability and that alleviates some of the problems in constructing the BP efficiently. The method randomly generates subgraphs of the BP that are computationally easier to construct than the BP itself. If the state that is sought is reachable, then it is represented in at least one of these subgraphs with high probability. The gains in efficiency enable fast reachability testing for some families of large models that cannot be handled by the deterministic algorithm in reasonable time. In particular, we demonstrate reachability testing on Boolean networks and cellular automata of up to 144 nodes within seconds.

Randomized methods for model checking based on sampling random trajectories were previously developed (15)(16). Our method differs from them in two main aspects: First, the models that we apply our method to are not stochastic but rather concurrent. Second, instead of generating random single runs of the model, we generate random representations of fractions of the state space. Hence, in one sampling we test multiple alternative runs of the system.

The next section provides a more detailed introduction to the concepts of a Petri net and a branching process. We then describe the randomized technique and prove its properties. Finally, we demonstrate it on test cases.

2 Methods

2.1 Petri Nets and Their Branching Processes

We now define the model more formally. For a fuller (and more rigorous) formulation see (8). A Petri net is a tuple N=(P,T,F,M_0) where P is a set of nodes called ***places***, T is a set of nodes called ***transitions***, $F \subseteq (P \times T) \cup (T \times P)$ is a set of directed edges between P and T. Each place has a non-negative integer number of ***tokens***. That number represents its state. The vector of tokens of the places of P is called a ***marking***, and it represents the network's state. M_0 is a marking called the ***initial marking*** of N. In a 1-safe Petri net, at any given time each place contains at most one token. We will assume that our nets are 1-safe. For extensions of the unfolding technique beyond the class of 1-safe Petri nets see, e.g., (14).

We call the places from which there is a directed edge to some transition its ***input places***. Similarly, we call the places to which a transition has directed edges its ***output places***. A transition is ***enabled*** if all of its input places contain tokens. An enabled transition can ***fire***: consume a token from each of its input places and produce a token to each of its output places. Hence, at any marking, an enabled transition can fire and produce a new marking, and the process can repeat indefinitely. Enabled transitions can fire at any order, i.e. the firing order is non-deterministic. A marking that can be obtained by some firing sequence starting from the initial marking is called a ***reachable marking***. **Fig. 1** shows a Boolean circuit and its representation as a Petri net.

The ***branching process*** of a Petri net N, denoted by BP, is an acyclic graph that represents every reachable marking. It is defined iteratively. Initially, there is one node in BP for every place in N that contains a token in M_0. In order to explain how BP is extended, we define two types of relationships between places in BP:

1. Place x is ***causal*** of place y if there is a path from x to y in BP.
2. Places x, y are ***in conflict*** if there is a place z in BP such that one can reach both x and y by exiting z through different edges.

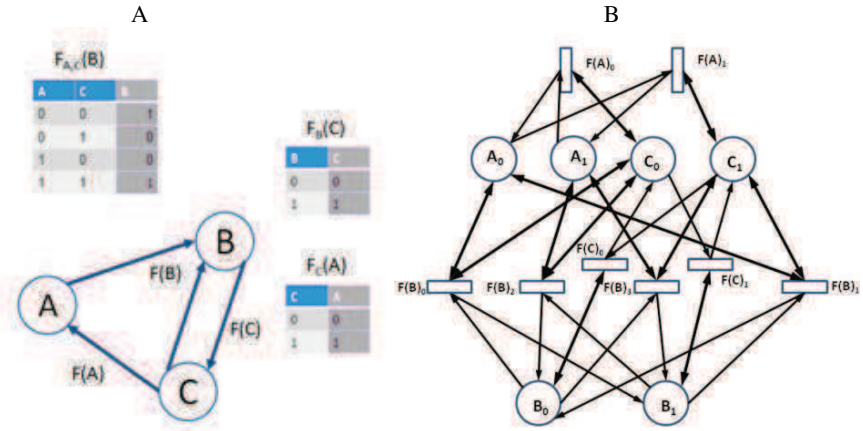


Fig. 1: A: A Boolean circuit. The circuit contains three Boolean entities, A, B and C. Entities A and C determine the value of entity B, B determines the value of C, and C determines the value of A. Each table shows the level of the output entity when its corresponding Boolean function acts, depending on the values of the inputs. **B**: A Petri net model for the Boolean circuit in A. Places are represented by circles and transitions by rectangles. The pairs of places (A_0,A_1), (B_0,B_1), and (C_0,C_1) are place invariants, which means that the sum of tokens in every pair of places is always 1. Every initial marking assigns one token to one place of each pair. The label of each transition is the regulated entity in parenthesis and the number of line in the Boolean truth table as a subscript in A. For example, $F(A)_0$ is the label of the transition that corresponds to the first line in the truth table $F_C(A)$.

Relationship 1 means that y represents the token that is produced by a firing sequence that consumes the token that x represents. Relationship 2 means that x and y represent tokens that are produced by two different firing sequences out of which only one can occur in the same run of N, since they both consume the same token - the one that is represented by z.

If a set of places does not contain any pair of places in conflict or in causal relationship we say that the places in that set are ***concurrent***. After the initialization all the places in BP are concurrent. However, as BP is extended places that are not concurrent are added to it. In order to extend BP, we pick a transition *t* in the Petri net and find a set of concurrent places in BP that correspond to its input places. Then we add *t* to BP, add outgoing edges from these places in BP to *t*, and also add places to BP that correspond to *t*'s output places, with incoming edges from *t*. The new transition that was added to BP represents the firing of *t*, and its output places represent assignment of tokens to the output places of *t*. Thus, there can be multiple transitions (places) in BP that correspond to the same single transition (place) in the Petri net.

The iterative process can generate a finite or an infinite BP. McMillan defined a method that disallows continuation from specific transitions and creates only a finite prefix of the BP but still contains the information on all reachable markings. In order to explain the method works, we need to define another concept, a local configuration. Every reachable marking has a subset of transitions in BP that corresponds to a firing sequence that generates it. These subsets are called

configurations. For a transition t, the set of transitions from which there is a directed path to t is referred to as t's ***local configuration*** (denoted [t]), and is associated with a marking. Note that [t] includes t itself. The marking of [t] is the marking obtained by firing all the transitions that belong to [t]. McMillan (7) showed that a finite BP still represents every reachable marking if extension is halted in transitions t for which the local configuration [t] is associated with a marking that is also associated with some other local configuration [t'] that contains *fewer transitions*. He called these transitions ***cutoff points***. **Fig. 2** shows the unfolding of the Petri net from **Fig. 1B**.

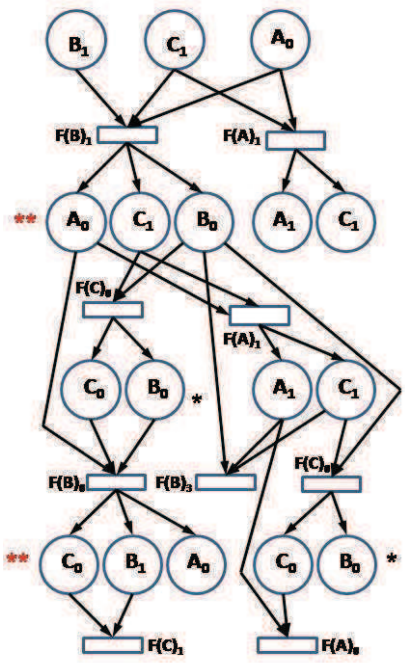


Fig. 2: Part of the branching process graph of the Petri net from Fig. 1B starting from state (OFF, ON, ON). Places are represented by circles and transitions by rectangles. For clarity, places and transitions in the branching process graph that correspond to the same place/transition in the Petri net carry the same label. The transition $F(C)_1$ (bottom) is an example for a cutoff point. The marking of its local configuration is the same as the initial marking, i.e. (OFF, ON, ON) (note that the initial marking is also considered a local configuration). The two places marked by two red stars are in a causal relationship. The two places that are marked by a black star are in a conflict relationship. The places that have directed edges to $F(C)_1$ are an example for concurrent places. The local configuration of $F(C)_1$ is { $F(C)_1$, $F(B)_0$, $F(C)_0$, $F(B)_2$ }.

One of the main tasks for which a BP is useful is answering reachability queries about the Petri net. Once the truncated BP has been constructed, it can be used for testing whether a given target marking or set of markings are reachable. Given a target marking and a set of places in the Petri net that contains tokens in that marking, add an artificial absorbing transition to the Petri net that consumes from all the places in the set and upon any addition to the BP check first if that transition can fire. If it can - stop with a positive answer. This is equivalent to testing if there is a concurrent combination of places in the BP that represents the target marking, in which case the answer is positive, or none was found, in which case the answer is negative.

Esparza, Romer and Vogler (8) showed that McMillan's criterion can sometimes produce a BP of exponential size even though the actual state space size is polynomial, and presented an improved cutoff criterion. According to the criterion, every local configuration is assigned a string that is a lexicographic concatenation of

the labels of transitions that occur in it. Each label is concatenated a number of times equal to the number of its appearances in the local configuration. For example, if t_1 appears once and t_2 appears twice, the label would be $t_1t_2t_2$. Now a transition is a cutoff point if there is another transition that represents the same marking whose local configuration is smaller, or if the other transition's local configuration is of identical size but has a lexicographically smaller label.

We review the proof of (8) that the resulting graph represents every reachable marking, because we will use a similar idea in the proof of our method. Let us call the original, typically infinite, BP the ***full BP***, and its finite prefix obtained using a cutoff criterion the ***truncated BP***. Suppose that we add to the BP transitions as long as we can, so that it can even become infinite. This full BP is then an interleaving of every possible firing sequence, and every reachable marking has a configuration in the full BP that corresponds to it. Consider some reachable marking and some configuration C that corresponds to it in the full BP. The truncated BP may not contain C, because one of its transitions t can become a cutoff point. In that case, according to the cutoff criterion there must be a transition t' in the truncated BP that satisfies either $|[t']|<|[t]|$ or $|[t']|=|[t]|$ and the label of [t'] is lexicographically smaller. So we can add to [t'] all the transitions in C\[t'], and obtain a new configuration C' that represents the same marking as C. If C' also contains a cutoff point, i.e. a transition that fires only in the full BP and not in the truncated BP, we can repeat the same construction. The process must stop because in every step we move towards a minimal element in a finite order of local configurations.

2.2 A Monte Carlo Method

The unfolding algorithm does more than determining whether a marking is reachable or not: if the target state is reachable, it also finds a sequence of transitions whose firing lead to that marking. Now suppose that we limit the unfolding algorithm using some criterion such that only a fraction of the markings will be represented in the truncated BP. In that case, if a marking is represented - it is reachable with certainty, and we can show a sequence of transitions that lead to it. However, if a marking is not represented - it can either be reachable or not.

Based on this observation, we propose a randomized Monte Carlo algorithm: If the randomized algorithm answers 'no' (i.e. that a state is unreachable), it has a constant probability smaller than 1 of making a mistake. If it answers 'yes' then the probability that the state is reachable is 1. By repeating the randomized algorithm sufficiently many times and answering 'reachable' if and only if the answer was 'reachable' in at least one of the repetitions, we get a small mistake probability. Note that it is essential that the probability of making a mistake in every repetition is independent of the other repetitions. By allowing the randomized algorithm a certain probability of mistake, it will run much faster than the deterministic unfolding algorithm. The improvement in running time is so significant that even when repeating the randomized algorithm many times it is still much faster the deterministic algorithm.

The algorithm depends on three parameters: The number of repetitions is denoted by ρ. Another parameter, $0<\varphi<1$, determines the probability of making a mistake in

a single repetition. N is an integer number greater than the maximum size of a configuration. A small N is generally not known in advance, and in the next section we discuss how to select one. The idea of bounding the number of steps in a trajectory in order to gain running time improvements has been used before in the context of bounded model checking (17).

The randomized algorithm adds a new type of cutoff point to the BP, which we call a ***randomized cutoff point*** (we refer to the other cutoff points as "ordinary"). A transition t becomes a randomized cutoff point according to the following criterion:

$$\text{Set t to be a cutoff point with probability } \varphi^{N-|[t]|}$$

where |[t]| is the size of the local configuration of t. Note that like any other transition the local configuration of a transition that becomes a randomized cutoff point is still recorded among all the local configurations detected by the algorithm. Therefore, other transitions, even if not originally direct descendants of it, can become normal cutoff points if their local configuration represents the same marking. The process is repeated at most ρ times and is stopped earlier with success if the desired marking is reached. A repetition ends with failure if no extension is possible, i.e. no transition can be added from the leaves of the current BP. If ρ repetitions ended with failure, the algorithm reports that the target is not reachable and terminates.

Lemma: The error probability in a single run is $1-\prod_{i=1}^{N}(1-\varphi^i)$.

Proof: Let C be some configuration of size *j* in the full BP. We would like to upper-bound the probability that it will not be represented in the truncated BP. Let $s_1,\ldots s_j$ be the sizes of the local configurations of the transitions in C. The probability that none of these transitions is a randomized cutoff point is $\prod_{i=1}^{j}(1-\varphi^{N-s_i})$. It can easily be shown that this probability is minimal for a configuration that corresponds to a chain of *j* transitions, i.e., the second transition consumes from the output of the first transition, the third from the output of the second, and so on. Hence, the probability that C does not contain a cutoff point is at least $\prod_{i=1}^{j}(1-\varphi^{N-s_i}) \geq \prod_{i=1}^{N}(1-\varphi^i)$, where the last inequality follows since $j<N$. Hence, the probability that C is not in the truncated in BP is at most $1-\prod_{i=1}^{N}(1-\varphi^i)$

Theorem: For any ε≥0, using the randomized cutoff criterion, we can obtain an error probability ≤ε by performing $\hat{\rho}=\left(\frac{1}{1-\varphi}^{\frac{1}{1-\varphi}}\right)\cdot\ln\left(\frac{1}{\varepsilon}\right)$ repetitions.

Proof. We return to the original algorithm's proof and use a similar rationale. Assume first that there are no ordinary cutoff points in the truncated BP, only randomized ones.

Define $n \equiv 1/\prod_{i=1}^{N}(1-\varphi^i)$. 1-1/n is the maximum error probability of the algorithm in a single run. The probability that C is never in the truncated BP in $\hat{\rho}$ repetitions is at most $\left(1-\frac{1}{n}\right)^{\hat{\rho}}$. Define $\hat{\rho} = \beta n$. Then $(1\text{-}1/n)^{\beta n} \sim e^{-\beta}$ for n sufficiently large. By setting $\beta \geq \ln(1/\varepsilon)$ we get $e^{-\beta} \leq \varepsilon$.

It now remains to evaluate n in terms of φ and α. Rewriting the expression for n we get: $\ln(n) = -\sum_{i=1}^{N}\ln(1-\varphi^i)$ From the Taylor series for ln(1+x) we get:

$$\ln(n) = -\sum_{j=1}^{N}\sum_{i=1}^{\infty}\frac{(-1)^{i+1}}{i}\cdot\left(-\varphi^j\right)^i = -\sum_{j=1}^{N}\sum_{i=1}^{\infty}\frac{(-1)^{2i+1}}{i}\cdot\left(\varphi^j\right)^i =$$

$$\sum_{j=1}^{N}\sum_{i=1}^{\infty}\frac{1}{i}\cdot\left(\varphi^j\right)^i = \sum_{i=1}^{\infty}\frac{1}{i}\sum_{j=1}^{N}\left(\varphi^i\right)^j \leq \sum_{i=1}^{\infty}\frac{1}{i}\cdot\frac{\varphi^i}{1-\varphi^i} < \sum_{i=1}^{\infty}\frac{1}{i}\cdot\frac{\varphi^i}{1-\varphi}$$

Using the Taylor series for ln(1-x) we have:

$$\ln(n) \leq \frac{1}{1-\varphi}\cdot\sum_{i=1}^{\infty}\frac{\varphi^i}{i} = \frac{1}{1-\varphi}\cdot\ln\frac{1}{1-\varphi}$$

Or $n \leq \frac{1}{1-\varphi}^{\frac{1}{1-\varphi}}$. Hence, $\hat{\rho} = \left(\frac{1}{1-\varphi}^{\frac{1}{1-\varphi}}\right)\cdot\ln\left(\frac{1}{\varepsilon}\right)$ repetitions guarantee an error of at most ε.

So far the analysis ignored ordinary cutoff points. Now we will extend the proof to include them (since we use ordinary points as well in the randomized version). Let C be a configuration in the full BP that contains an ordinary cutoff point **t** in the truncated BP. Since **t** is an ordinary cutoff point, there is another transition **t'** in the truncated BP such that [**t'**] has the same marking and either |[**t'**]|<|[**t**]|, or |[**t'**]|=|[**t**]| and the label of [**t'**] is lexicographically smaller. We build a new configuration C' as in the original proof, by adding to [**t'**] the transitions in C\ [**t'**]. It must hold that C' is smaller than C in the partial order of local configurations. Now if C' still contains an ordinary cutoff point, we repeat the process exactly as in the proof of the original algorithm. Note that as long as the configuration C' that we have contains ordinary cutoff points we can continue this process, even if C' contains randomized cutoff points, because the local configuration of an ordinary cutoff point in the truncated BP does not contain randomized cutoff points. Eventually we obtain a configuration that does not contain any ordinary cutoff points. From here the arguments presented earlier apply.

Note that if we have M disjoint configurations that represent the same marking as C, we can divide $\hat{\rho}$ by M. M can even be larger when we are not seeking one specific marking, but rather any marking in which all the places in a *prespecified subset*

contain tokens, irrespective of the other places. The latter problem is well studied in the context of Petri net modeling and has been shown to be PSPACE-Hard (7,18). Schwoon et al. demonstrate a factorial growth in the number of representations of places and transitions in a Petri net that contains read arcs (19). This redundancy has a central role in the usefulness of random partial state space representations. In particular, it counterbalances a high rate of random cutoffs that would otherwise cause failure. Networks that do not display this kind of redundancy are more likely to be handled efficiently by the deterministic algorithm.

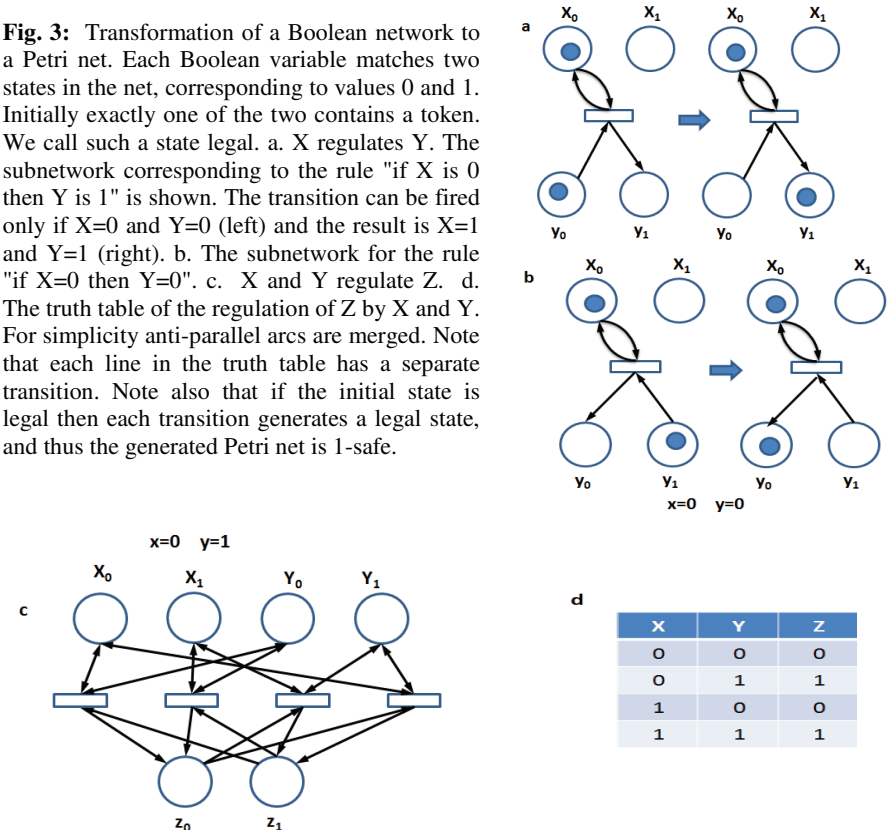
The theorem implies that with a sufficient (but possibly very large) number of repetitions a constant probability of error is assured, even if every marking is represented by exactly one configuration. In practice, one can apply the algorithm for a much smaller number of repetitions, which we denote by ρ, and obtain good results without the theoretical guarantee. For some types of networks and markings the number of configurations that represent the marking that we seek will grow very fast with the size of the model, and then we can adjust the parameters φ and ρ in order to cope with larger networks - increasing φ will decrease the running time and increase the probability of making an error, and increasing ρ will increase the running time and decrease the probability of making an error. As we show in the next section, these parameters can be empirically calibrated to get a small error probability and a running time of seconds for networks of 100 nodes.

3 Experimental Results

To test the performance of our algorithm, we first randomly generated asynchronous Boolean networks, and transformed them into 1-safe Petri nets. An initial state was chosen randomly for each net, a long trajectory of states was followed starting from it, a subset of nodes was chosen randomly, and the algorithm was asked to check if the state of these nodes at the end state (the target marking) can be reached from the initial state. Hence, all tests had a 'true' answer. We applied both the randomized algorithm and the deterministic algorithm of (8) to each instance. In both algorithms we tested the target marking after each increment of the BP, and halted the construction of the BP when the target marking is found. In subsequent tests we evaluated different network topologies and sizes as well as unreachable instances.

Generating Boolean networks. The structure of the Boolean network we constructed is as follows: Each node in the network is regulated by either one or two other nodes, with equal probability to each number of regulators. We select the regulator or the two regulators uniformly and independently (disallowing self regulations). The regulation logic is determined by setting the level of the regulated node to 0 or 1 with equal probability independently for each combination of regulators values. Hence, the value of the regulated node in each row of its truth table is set independently to 0 or 1 with probability 0.5. The use of random topologies is aimed to test the applicability of our method to realistic design problems, in which strict constraints on topology do not always emerge. Some examples for uses of

random topologies can be found in (9,20). The initial state is determined by setting the value of each node to 0 or 1 with equal probability independently of the other nodes. The target state is generated by repeatedly performing the following step: randomly choose a node and update its level. The target marking here is the resulting state after k updates, where k is a parameter of the experiment. Hence, the target marking is reachable, i.e. a 'yes' instance. We use this method instead of random selection of the target state since in many of our examples, when selecting the target state randomly the deterministic algorithm did not terminate even after a very long time, so we could not be certain that the target state is not reachable.



X	Y	Z
0	0	0
0	1	1
1	0	0
1	1	1

Fig. 3: Transformation of a Boolean network to a Petri net. Each Boolean variable matches two states in the net, corresponding to values 0 and 1. Initially exactly one of the two contains a token. We call such a state legal. a. X regulates Y. The subnetwork corresponding to the rule "if X is 0 then Y is 1" is shown. The transition can be fired only if X=0 and Y=0 (left) and the result is X=1 and Y=1 (right). b. The subnetwork for the rule "if X=0 then Y=0". c. X and Y regulate Z. d. The truth table of the regulation of Z by X and Y. For simplicity anti-parallel arcs are merged. Note that each line in the truth table has a separate transition. Note also that if the initial state is legal then each transition generates a legal state, and thus the generated Petri net is 1-safe.

To represent the Boolean network as a Petri net we replace each node by a pair of places corresponding to the 0 and 1 levels of the variable, and add a transition for each row in its truth table. See **Fig. 3** for an example. Note that this standard representation always produces a 1-safe network (21). The resulting Petri net will have $2s$ places and $4s$-$2t$ transitions if the Boolean network had t nodes with one regulator and s-t nodes with two regulators.

Testing the theoretical bound. In order to test the theoretical bound on ρ that we computed in the previous section, we analyzed Boolean networks of limited problem

size that is common in contemporary applications of model checking(20,22,23). We generated 100 networks with s=15 nodes, for each one generated a trajectory of size s and recorded the target state as the final local state of every entity, generating a target state of size 15. We ran the deterministic and the randomized unfolding algorithms on this test set. For the randomized algorithm, we selected φ=0.85 in order to obtain a truncated BP within a few seconds. According to the Theorem, for an error probability ε=0.01 we set the number of repetitions to ρ=310,975 ln(100). N is set to s as we know that this is the maximal possible local configuration size.

Out of the 100 instances, 20 runs of the deterministic algorithm did not terminate within five days and were stopped. **Fig. 4** compares the running times of the two algorithms on the remaining 80 instances. The randomized algorithm terminated on all 100 instances and always found the correct target state, in accordance with the prediction of the theoretical upper bound. The randomized algorithm was on average ~437 times faster. The figure also shows correlation between the running times of the two algorithms. The instance for which the randomized algorithm required the longest time was solved quickly by the deterministic algorithm, suggesting that in that case the randomized algorithm performed a high number of repetitions that missed the target state. The average running time of the randomized algorithm on the 20 instances for which the deterministic algorithm did not terminate was ~4.5 hours, indicating that on average these instances were more difficult than the other 80 for both algorithms.

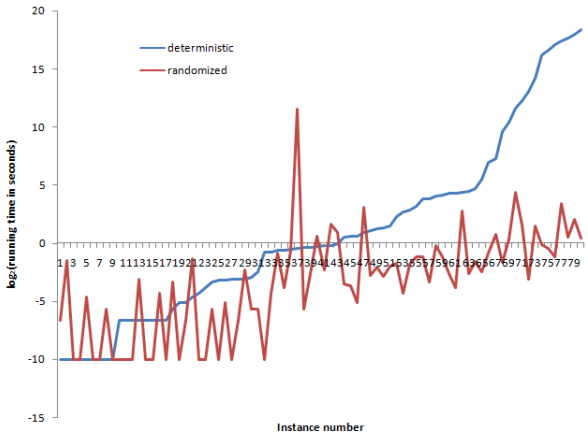


Fig. 4 Running times of the deterministic and randomized algorithms on random Boolean networks. The instances are sorted according to the running time of the deterministic algorithm. The randomized algorithm is faster or comparable to the deterministic algorithm in most cases. Of 100 tested networks, only the 80 on which the deterministic algorithm terminated within five days are shown.

Tests on larger Boolean networks with smaller number of repetitions. The randomized algorithm can be practical in some cases even when using a much lower value of ρ than the theoretical upper bound. In order to demonstrate that, we proceed by showing that the randomized algorithm can successfully test reachability for Boolean networks of 50 nodes and a target marking composed of five places.

Fig. 5 plots the average and standard deviation of the running time of the algorithm for networks of 50 nodes and trajectory length L=50,70,90,…,450. For every value of L we generated 100 random Boolean networks that have a trajectory of length L.

and φ grow with the size of the model. The deterministic unfolding algorithm (8) was run on the same networks, until the target state of the chosen nodes is reached. Its average running time is also given in the table. On some instances it did not terminate after 24 hours, and at this point we halted the algorithm and considered its running time to be 24 hours, although it would in fact be longer. The table shows the number of instances on which the algorithm terminated.

Generating cellular automata. Next, to test a different model topology, we generated asynchronous cellular automata. This model has been used in numerous different studies, for example as a model for the migration of glioma cells [1] and ecological systems (24,25). We generated the automata using a two-dimensional square lattice. For a *kxk* lattice, the cells correspond to lattice points $\{(i,j), 1 \leq i,j \leq k\}$, and the regulators of a cell (i,j) are its neighbors {(i-1,j), (i+1,j), (i,j-1), (i,j+1)} that fall within the lattice boundaries. Therefore, the inner cells have four regulators each, and cells at the border have 3 or 2 neighbors. The cells change their level from 0 to 1 or vice versa asynchronously, according to the state of their regulators. The regulatory logic of each cell is a majority vote over its regulators' levels, where equality in the vote gives output 0. Hence, a corner cell will attain level 1 if and only if both its neighbors are at level 1. A boundary (resp. internal) cell will attain state 1 iff at least two (resp. three) of its regulators are at state 1.

Impact of the topology on the algorithm's parameters. We generated automata in square lattices of sizes s=16,25,36,..100. Note that for each size there is only one automaton. Therefore, what is chosen at random is only the initial state. Every cell is set to 0 or 1 independently with probability 0.5. For every lattice size we generated 100 random initial lattice states. For each initial state we generated a target state by following a trajectory of length s and recording the state of two randomly selected nodes in the final state as above. N was set to s. The encoding of an automaton as a 1-safe Petri net uses the same transformation as in **Fig. 3**, but with 2-4 regulators for each state.

Table 2 shows the parameters ρ and φ for each lattice size, and the results for the deterministic algorithm. On both network structures, the randomized algorithm handles all the networks up to size 100 within seconds without errors, while the deterministic algorithm requires on average a long time for sizes beyond 20-30. The deterministic algorithm has somewhat higher termination rates with the cellular automaton model, which can be attributed to the restricted topology and to our choice of uniform logic for this model, but it is still slower for most of the large datasets. Interestingly, even for the larger networks, the deterministic algorithm tended either to give a solution within seconds or to consume the whole 24 CPU hours without reaching a solution. The fraction of the slow instances grew with the problem size.

Size	Places	Transitions	Randomized Algorithm			Deterministic Alg.	
			φ	ρ	Time (sec)	Time (sec)	% ended
10	20	20-40	0.5	1	0.01	0.02	100
20	40	40-80	0.97	100	0.01	22466	74
30	60	60-120	0.98	100	0.03	31149	64
40	80	80-160	0.991	100	0.02	16935	81
50	100	100-200	0.992	100	0.04	24451	72
60	120	120-240	0.993	100	0.07	39617	54
70	140	140-280	0.994	100	0.11	31198	64
80	160	160-320	0.995	100	0.57	30873	65
90	180	180-360	0.996	100	0.21	36295	58
100	200	200-400	0.997	100	0.28	42337	51

Table 1: Algorithm performance and parameters for asynchronous Boolean networks of different sizes. For each size, 100 networks were randomly generated and a random trajectory of length equal to the size was generated in each (networks that did not allow such a trajectory were rejected). The average running times of the deterministic and randomized algorithm are listed. The deterministic algorithm was terminated after 24 hours if no solution was reached. "% ended" indicates the fraction of instances that gave a solution. Failed instances contribute 24 hours to the average running times.

The unreachability analysis and the tests in **Tables 1** and **2** were performed on x86 64 bits machines with a Pentium IV or Xeon processor and at most 2 GB RAM. Jobs were run in a time-sharing environment. The results described in **Fig. 4** were generated sequentially on an x86 64 bit machine with a Xeon processor and 8 GB RAM. The code was written in C and compiled in Linux. Interestingly, the Windows random number generator proved to be inappropriate for the randomized algorithm and using it reduced the probability of finding the target state Writing an ad-hoc generator solved the problem. For the deterministic and the randomized algorithms, the Esparza-Romer-Vogler (8) version of the unfolding algorithm was implemented, with a speedup for selection of concurrent places (details are given in the appendix). Our source code is freely available at http://www.tau.ac.il/~guykarle/CAV2012.zip.

4 Discussion

A branching process is a compact representation of a Petri net's state space. Constructing a small representation is vital for alleviating the state explosion problem. Even with the best deterministic algorithms for testing reachability, the problem is intractable for all but very small models.

Size	Places	Transitions	Randomized Algorithm			Deterministic Alg	
			φ	ρ	Time (sec)	Time (sec)	% ended
9	18	48	0.5	1	0.01	0.01	100
16	32	112	0.91	10	0.49	14128	84
25	50	208	0.97	10	0.01	4321	95
36	72	336	0.99	100	0.01	15554	82
49	98	496	0.991	10	0.14	37183	57
64	128	688	0.993	10	0.09	39745	54
81	162	912	0.995	10	0.74	46656	46
100	200	1168	0.995	10	0.03	2636	97
121	242	1456	0.998	1000	0.29	6921	92
144	288	1776	0.9985	1000	0.93	30412	65

Table 2: Algorithm performance and parameters for different network sizes of asynchronous cellular automata. The average running times of the deterministic and randomized algorithm are compared. The percentage of instances on which the deterministic algorithm ended within 24 hours is given in the rightmost column Time is the average running time (including unfinished instances) over 100 random initializations.

We presented a fast new randomized method and showed that it can handle problem instances that existing algorithms cannot. The method is based on randomly creating representations of prefixes of a state space representation, checking them for reachability, and repeating the process to reduce the chance for error. For all but the smallest tested problem instances, the deterministic algorithm requires on some of the instances a running time larger by orders of magnitude from that of the randomized one. The randomized algorithm requires seconds for networks of up to 144 nodes.

One shortcoming of our algorithm is that it requires calibration of parameters depending on the input. We observed that different input sizes require different parameters, and used a training set of networks for calibration. While the process was quick and effective on our problems, it is heuristic, and developing a better method for selecting these parameters is of interest. Automatic implementation can utilize binary search in order to increase efficiency. Another shortcoming is pointed out by theoretical analysis. We showed that ρ has an exponential dependency on $(1-\varphi)^{-1}$, and since larger models require larger values of φ, the number of repetitions to reduce the error will become high. For the model sizes examined in this work the values of ρ were small enough to obtain an efficient algorithm. Ultimately a characterization of network classes for which specific values of φ perform well will obviate the need for this calibration.

Our algorithm is randomized and is not guaranteed to work for every instance and every reachability query. It will not outperform the deterministic algorithm on all problem types. Since disjoint local configurations of the same marking will speed up the algorithm, the algorithm is expected to perform well when it seeks markings that can be reached by many (sometimes even exponentially many) firing sequences. Such situations are common in many applications (21,26). If the "breadth" of the unfolding grows fast relative to its "depth", then pruning the truncated BP is less likely to lose

information about reachability of the target state, while a deterministic approach would slow down significantly as depth increases. It should be noted in this context that the randomized algorithm can be easily parallelized, in which case the number of repetitions is divided by the number of processors. In the future we will further test the applicability of the algorithm on additional types and sizes of Petri nets.

Acknowledgments
The authors would like to thank Javier Esparza and the anonymous referees of the TACAS12 conference for their helpful comments on this manuscript. This study was supported in part by the European Community's Seventh Framework Program (grant HEALTH-F4-2007-200767 for the APO-SYS project) and by the Israeli Science Foundation (grant 802/08). GK was supported in part by a fellowship from the Edmond J. Safra Bioinformatics program at Tel-Aviv University.

Appendix: Selection of Concurrent Places

We address here the selection of sets of concurrent places in the BP. Exhaustively testing all sets of places in BP is highly inefficient. For example, if t is a transition that consumes from 3 places, we have to find all the sets of places in BP that correspond to these 3 places and are concurrent. A more efficient way to select places for a transition that has L input places is the following procedure:

```
BT_SELECT_SET(i)
1. Select the next candidate for the transition's ith
   input place from BP, or if there is no such place
   return.
2. If the selected place is in conflict with any of
   the previous places selected (1,..,i-1), go back to
   1.
3. Otherwise
      a. If i=L, output the concurrent set and return.
      b. Otherwise BT_SELECT_SET(i+1)
```

The procedure described above is a backtracking procedure. It adds places to the concurrent set until it finds a complete set of input places. At every depth of the recursion, the procedure selects one place from a list of places in BP that represent the same place in the corresponding Petri net. When no place can be added at depth i^{th} of the recursion, which means that all the places that correspond to the i^{th} input of the transition are in conflict with the previously selected places, the algorithm backtracks.

Claim 1: If p and p' are places in BP that represent the same place in the corresponding Petri net, and p' was added before p, then exactly one of the following must hold:

1. There is a directed path from p' to p
2. p and p' are in conflict

Proof: Assume towards contradiction that the claim is not true. p cannot have a path to p' because it was added after p' and BP is acyclic. Then according to the assumption p and p' represent the same place and they are not in conflict. Therefore, there exists a firing sequence in the corresponding Petri net that puts one token in the place that p represents and one token in the place that p' represents. However, that is the same place, in contradiction to the fact that the Petri net is 1-safe.

How can we use claim 1 to improve the selection of concurrent sets? First, we make BT_SELECT_SET() go over the places in step 1 in decreasing order of addition time to BP. I.e., it will first consider the last place that was added to BP, then the one that was added before the last and so on. Now we implement another backtrack criterion:

Claim 2: If the following conditions hold:

1. The backtracking procedure leaves depth i from step 3b (a recursive call)
2. It returns to depth i without reaching step 3a (finding a solution)
3. Every place at depths i+1,…,L that is in conflict with the place at depth i, and is concurrent with the places at depths 1,…,i-1 has a path to it from the place at depth i

then the procedure can backtrack to depth i-1 without performing any additional recursive calls from depth i, and solutions will not be missed.

Proof: Let p_i be the place that was selected at depth i. Due to Claim 1, every place p' that the procedure did not yet consider at position i either has a path to p_i or is in conflict with it. If p' has a path to p_i, then due to condition 3 of claim 2, it also has a path to every place that is concurrent with the places at depths 1,…,i-1. If it is in conflict with p_i, then there is a path that starts in some place p", exits through different transitions and reaches p_i and p'. Since p_i has a path to every place that is concurrent with the places at depths 1,…,i-1, p' is in conflict with each one of these places as well. Therefore, any p' that was not considered yet can be skipped, and the procedure can backtrack.

References:

(1) Carl Adam Petri. Kommunikation mit Automaten, Universitaet Hamburg; 1962.

(2) Karatkevich A. Dynamic analysis of Petri net-based discrete systems. Berlin: Springer; 2007.

(3) Karlebach G, Shamir R. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case. BMC Syst Biol 2010 Feb 25;4:15.

(4) Koch I, Reisig W, Schreiber F. Modeling in systems biology the Petri net approach. London: Springer; 2011.

(5) Li Z, Zhou M. Deadlock resolution in automated manufacturing systems: a novel Petri net approach. Berlin ; London: Springer; 2009.

(6) Yakovlev AV, Gomes L, Lavagno L. Hardware design and Petri nets. Boston, Mass.: Kluwer Academic Publishers; 2000.

(7) McMillan KL, Probst DK. A technique of state space search based on unfolding. Formal Methods Syst Des 1995;6(1):45-65.

(8) Esparza J, Romer S, Vogler W. An improvement of McMillan's unfolding algorithm. Lecture Notes in Computer Science;Tools and Algorithms for the Construction and Analysis of Systems 1996;1055:87-106.

(9) Bonet B, Haslum P, Hickmott S, Thiébaux S. Directed Unfolding of Petri Nets. Transactions on Petri Nets and other Models of Concurrency 2008;5100:172-198.

(10) Khomenko V, Koutny M. Towards an Efficient Algorithm for Unfolding Petri Nets. Concurrency Theory 2001;2154:366-380.

(11) Heljanko K, Khomenko V, Koutny M. Parallelisation of the Petri Net Unfolding Algorithm. Tools and Algorithms for the Construction and Analysis of Systems 2002;2280:371-385.

(12) Karlebach G, Shamir R. Modelling and analysis of gene regulatory networks. Nat Rev Mol Cell Biol 2008 Oct;9(10):770-780.

(13) Petri C. Non-sequential processes. Gesselschaft fur Matematik unt Datenverarbeitung. Technical Report ISF-77-5 1977.

(14) Esparza J, Heljanko K. Unfoldings -- A Partial-Order Approach to Model Checking. : Springer-Verlag; 2008.

(15) Younes H, Simmons R. Probabilistic Verification of Discrete Event Systems Using Acceptance Sampling. In: Brinksma E, Larsen K, editors. Computer Aided Verification: Springer Berlin / Heidelberg; 2002. p. 23-39.

(16) Grosu R, Smolka S. Monte Carlo Model Checking. Tools and Algorithms for the Construction and Analysis of Systems 2005;3440:271-286.

(17) Biere A, Cimatti A, Clarke EM, Strichman O, Zhu Y. Bounded Model Checking. Advances in Computers: Elsevier; 2003. p. 117.

(18) Esparza J. Decidability and complexity of Petri net problems — An introduction. Lectures on Petri Nets I: Basic Models 1998;1491:374-428.

(19) Rodriguez C, Schwoon S, Baldan P. Efficient Contextual Unfolding. CONCUR 2011 – Concurrency Theory 2011;Volume 6901:342-357.

(20) Ay F, Xu F, Kahveci T. Scalable steady state analysis of Boolean biological regulatory networks. PLoS One 2009 Dec 1;4(12):e7992.

(21) Steggles LJ, Banks R, Shaw O, Wipat A. Qualitatively modelling and analysing genetic regulatory networks: a Petri net approach. Bioinformatics 2007 Feb 1;23(3):336-343.

(22) Parameter inference for asynchronous logical networks using discrete time series. Proceedings of the 9th International Conference on Computational Methods in Systems Biology New York, NY, USA: ACM; 2011.

(23) Corblin F, Fanchon E, Trilling L. Applications of a formal approach to decipher discrete genetic networks. BMC Bioinformatics 2010 Jul 20;11:385.

(24) Aubert M, Badoual M, Fereol S, Christov C, Grammaticos B. A cellular automaton model for the migration of glioma cells. Phys Biol 2006 Apr 13;3(2):93-100.

(25) Gronewold A, Sonnenschein M. Event-based modelling of ecological systems with asynchronous cellular automata. Ecol Model 1998;108(1-3):37.

(26) Baldan P, Corradini A, Konig B, Schwoon S. McMillan's Complete Prefix for Contextual Nets. Transactions on Petri Nets 2008;5100:199-220.

7. Discussion

In this chapter I discuss my conclusions regarding analysis of GRNs using logical models. I try to predict their use in regulatory network analysis, and to pinpoint some of the most urgent problems that need to be studied. At the end of the chapter I discuss how the methods that I developed in this thesis may be extended and improved.

7.1 Logical models as the gold standard for gene network analysis

The relatively low level of detail in logical models matches well with the current limited mechanistic knowledge about gene networks. On the other hand experimental data are not straightforwardly interpreted by these models. Hence if technology will advance in a direction that will improve our understanding of how gene regulation operates, continuous models may replace logical models, and ultimately molecular level models will replace continuous models. The difficulty in analyzing the more detailed models could be overcome by simulation or approximation techniques (66).

At this stage a lot is still unknown about gene regulation. For example, the mechanism by which TFs find the exact location of their targets in reasonable time has not been conclusively determined yet (67), and there are many open questions regarding the transport of gene products between the nucleus and the cytoplasm and its impact on gene regulation (68). If mechanistic understanding will not advance so quickly, then logical models may become a gold standard, provided that experimental measurements could be interpreted correctly in a logical context (69).

It should be remembered that in the past scientific models always evolved from simple models that neglect some factors(e.g. Galileo's experiments with falling objects did not include air resistance) into more accurate descriptions that take into account more and more refined phenomena. The view that a logical description of gene networks encompasses complex pathological phenomena like cancer supports their potential as a scientific tool (70,71).

7.2 Open questions

There are several open questions regarding logical modeling of GRNs, some directly related to biology and some more theoretical but motivated by their biological relevance.

The first question is whether to use synchronous or asynchronous models (49). All that has been discovered about gene regulation does not indicate a synchronization mechanism in any direct manner. On the other hand, some studies using synchronous networks, including the ones discussed in Chapter 1, claim to be quite good approximations to what occurs in nature, and some theoretical claims have also been made (72). The state transition graph of synchronous networks has fewer edges, and therefore it is much simpler to analyze. If the asynchronous scheme is to be adopted, methods for more efficient analysis need to be developed.

Another question is how to discretize experimental data and fit them to the logics of a logical model. Experimental data provide us with clues to the interactions between TFs and target genes, and these interactions are assigned continuous certainty levels. The data also contain continuous gene expression values. In order to construct a logical model we need to integrate these data, since separately each data type only gives a partial picture (73,74), and we also need to select a discretization criterion. How to select this criterion, and how accurate such a selection can be, are problems that need to be studied more deeply. As we have seen in Chapter 5, recovering the network logic is an NP-Hard problem (3).

The connection between network behavior and observed phenotype also needs to be characterized better. Kauffman conjectured that steady states and state cycles (attractors) correspond to phenotypes, by qualitatively comparing these behaviors to cellular phenotypes like the cell cycle and differentiation (75). He based his conjecture on statistical analysis of randomly generated Boolean networks in which he observed cycles that are proportional in length to the number of nodes and multiple steady states and state cycles that can be traversed by applying small perturbations. Various contemporary models tried to apply this definition to real networks (13,16,76). However sometimes a perturbation of a single TF can induce a rather complex phenotype like speech impairment or change in metabolic flux (77,78), and it has been suggested that a relatively modest number of genes can define the state of a cell (79). One may ask thus if it is necessary to define a network phenotype using more than a small subset of its nodes. In addition, if a phenotype is to be seen as a steady state behavior, then the contribution of epigenetic modifications to steady state expression should be studied and the method of describing them as a part of a Boolean model should be determined (80). Finally, a biological steady state may not be adequately represented by a steady state of a Boolean network due to inaccuracies in the Boolean terminology, and therefore definition of a steady state as phenotype requires proving

that biological GRNs produce a behavior that is comparable to a deterministic, logical behavior (27).

7.3 Future research

In the future I would like to continue the research developed in this thesis in several directions.

- **Network reconstruction**: The network reconstruction algorithm presented in this work needs to be studied further. The entropy landscape that it generates and its dependence on the quality of expression data should be better quantified. The introduction of an asynchronous update policy should be considered. While our current experience indicates that for reconstruction purposes an asynchronous model may be a good approximation, more datasets need to be tested. A comparison between the scores of different regulation functions or sets of regulators can also lead to important insights, and can improve the utilization of the reconstructed model for dynamical analysis. The process of discrepancy reduction may also be harnessed for more accurate selection of regulators.
- **The embryonic stem cell network:** Large amounts of experimental data are available for the mouse embryonic stem cell network. The reconstructed ESC model should be subjected to dynamical analysis in order to study the initialization of embryonic development in mouse. The randomized unfolding algorithm can be used for searching the space of regulatory states and compare it to evidence obtained from time-series expression profiles. Results of perturbations can be compared to the effects of known pluripotency factors (81).
- **Dynamical analysis:** The analysis of network dynamics is a provably hard undertaking. The randomized unfolding algorithm is a new approach to model checking that utilizes randomness. The role of randomness in state space construction should be studied further. Application to other model checking techniques and to other network queries, such as steady state detection, needs to be examined. In addition, other model checking techniques can be adapted to test the probabilistic phenotype defined in Chapter 4, and their performance can be compared to Petri net unfolding.
- **Using networks for correcting measurement errors in gene expression data:** Regulatory networks are the mechanism that underlies gene expression

patterns in cells, and consequently they are also a promising tool for correcting errors in gene expression data. The network reconstruction algorithm presented in this work contains a procedure for minimizing discrepancies between expression data and a network topology. This procedure can be extended and adapted for the purpose of correcting errors in gene expression data, and thereby improve the results of other algorithms such as clustering (82) and feature selection (83).

In summary, I developed methods that address problems arising in different aspects of modeling – network reconstruction, incorporation of uncertainties and dynamical analysis. These methods are meant to improve our understanding of gene regulatory networks and our ability to utilize them for the advancement of science and medicine. Several applications display the novelty and robustness of these methods, and produce predictions that can be further tested in the lab. In the future I would like to continue both the theoretical and practical study of gene regulatory network models.

Bibliography

(1) Karlebach G, Shamir R. Modelling and analysis of gene regulatory networks. Nat Rev Mol Cell Biol 2008 Oct;9(10):770-780.

(2) Karlebach G, Shamir R. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case. BMC Syst Biol 2010 Feb 25;4:15.

(3) Karlebach G, Shamir R. Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy-Based Approach. J Comput Biol 2012 Jan;19(1):30-41.

(4) Karlebach G, Shamir R. A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets. Submitted 2011.

(5) Kitano H. Computational systems biology. Nature 2002 Nov 14;420(6912):206-210.

(6) Kitano H. Systems Biology: A Brief Overview. Science 2002 March 01;295(5560):1662-1664.

(7) Stormo GD, Zhao Y. Determining the specificity of protein-DNA interactions. Nat Rev Genet 2010 Nov;11(11):751-760.

(8) Yu X, Schneiderhan-Marra N, Joos TO. Protein microarrays and personalized medicine. Ann Biol Clin (Paris) 2011 Jan-Feb;69(1):17-29.

(9) Liu F, Kuo WP, Jenssen TK, Hovig E. Performance comparison of multiple microarray platforms for gene expression profiling. Methods Mol Biol 2012;802:141-155.

(10) Roy NC, Altermann E, Park ZA, McNabb WC. A comparison of analog and Next-Generation transcriptomic tools for mammalian studies. Brief Funct Genomics 2011 May;10(3):135-150.

(11) Wang B, Howel P, Bruheim S, Ju J, Owen LB, Fodstad O, et al. Systematic evaluation of three microRNA profiling platforms: microarray, beads array, and quantitative real-time PCR array. PLoS One 2011 Feb 11;6(2):e17167.

(12) French AP. Newtonian mechanics. London: Nelson; 1971.

(13) Shlomi T, Cabili MN, Herrgard MJ, Palsson BO, Ruppin E. Network-based prediction of human tissue-specific metabolism. Nat Biotechnol 2008 Sep;26(9):1003-1010.

(14) Thomas R. Boolean formalization of genetic control circuits. J Theor Biol 1973 Dec;42(3):563-585.

(15) Glass L, Kauffman SA. The logical analysis of continuous, non-linear biochemical control networks. J Theor Biol 1973 Apr;39(1):103-129.

(16) Li F, Long T, Lu Y, Ouyang Q, Tang C. The yeast cell-cycle network is robustly designed. Proc Natl Acad Sci U S A 2004 Apr 6;101(14):4781-4786.

(17) Kingsmore SF. Multiplexed protein measurement: technologies and applications of protein and antibody arrays. Nat Rev Drug Discov 2006 Apr;5(4):310-320.

(18) Ness SA. Microarray analysis: basic strategies for successful experiments. Mol Biotechnol 2007 Jul;36(3):205-219.

(19) Segal E, Raveh-Sadka T, Schroeder M, Unnerstall U, Gaul U. Predicting expression patterns from regulatory sequence in Drosophila segmentation. Nature 2008 Jan 31;451(7178):535-540.

(20) Weaver DC, Workman CT, Stormo GD. Modeling regulatory networks with weight matrices. Pac Symp Biocomput 1999:112-123.

(21) Yeung MK, Tegner J, Collins JJ. Reverse engineering gene networks using singular value decomposition and robust regression. Proc Natl Acad Sci U S A 2002 Apr 30;99(9):6163-6168.

(22) Edwards JS, Ibarra RU, Palsson BO. In silico predictions of Escherichia coli metabolic capabilities are consistent with experimental data. Nat Biotechnol 2001 Feb;19(2):125-130.

(23) Klipp E. Systems biology in practice: concepts, implementation and application. Weinheim: Wiley-VCH; 2005.

(24) Raj A, Peskin CS, Tranchina D, Vargas DY, Tyagi S. Stochastic mRNA synthesis in mammalian cells. PLoS Biol 2006 Oct;4(10):e309.

(25) Golding I, Paulsson J, Zawilski SM, Cox EC. Real-time kinetics of gene activity in individual bacteria. Cell 2005 Dec 16;123(6):1025-1036.

(26) Ross IL, Browne CM, Hume DA. Transcription of individual genes in eukaryotic cells occurs randomly and infrequently. Immunol Cell Biol 1994 Apr;72(2):177-185.

(27) McAdams HH, Arkin A. It's a noisy business! Genetic regulation at the nanomolar scale. Trends Genet 1999 Feb;15(2):65-69.

(28) Guptasarma P. Does replication-induced transcription regulate synthesis of the myriad low copy number proteins of Escherichia coli? Bioessays 1995 Nov;17(11):987-997.

(29) Bae K, Lee C, Hardin PE, Edery I. dCLOCK is present in limiting amounts and likely mediates daily interactions between the dCLOCK-CYC transcription factor and the PER-TIM complex. J Neurosci 2000 Mar 1;20(5):1746-1753.

(30) Arkin A, Ross J, McAdams HH. Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected Escherichia coli cells. Genetics 1998 Aug;149(4):1633-1648.

(31) Gillespie DT. Stochastic simulation of chemical kinetics. Annu Rev Phys Chem 2007;58:35-55.

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

(33) Lähdesmäki, H., Shmulevich, I. & Yli-Harja. On learning gene regulatory networks under the Boolean network model. Machine Learning 2003;52:147-167.

(34) Akutsu T, Tamura T, Horimoto K. Completing Networks Using Observed Data. Algorithmic Learning Theory 2009:126-140.

(35) Segal E, Shapira M, Regev A, Pe'er D, Botstein D, Koller D, et al. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat Genet 2003 Jun;34(2):166-176.

(36) Shamir R, Tanay A. Modeling Transcription Programs: Inferring Binding Site Activity and Dose-Response Model Optimization. RECOMB'03 (ACM, Berlin) 2003.

(37) Huynh-Thu VA, Irrthum A, Wehenkel L, Geurts P. Inferring regulatory networks from expression data using tree-based methods. PLoS One 2010 Sep 28;5(9):e12776.

(38) Lopes FM, Martins DC,Jr, Cesar RM,Jr. Feature selection environment for genomic applications. BMC Bioinformatics 2008 Oct 22;9:451.

(39) Shmulevich I, Gluhovsky I, Hashimoto RF, Dougherty ER, Zhang W. Steady-state analysis of genetic regulatory networks modelled by probabilistic boolean networks. Comp Funct Genomics 2003;4(6):601-608.

(40) Shmulevich I, Dougherty ER, Kim S, Zhang W. Probabilistic Boolean Networks: a rule-based uncertainty model for gene regulatory networks. Bioinformatics 2002 Feb;18(2):261-274.

(41) Steggles LJ, Banks R, Shaw O, Wipat A. Qualitatively modelling and analysing genetic regulatory networks: a Petri net approach. Bioinformatics 2007 Feb 1;23(3):336-343.

(42) Corblin F, Tripodi S, Fanchon E, Ropers D, Trilling L. A declarative constraint-based method for analyzing discrete genetic regulatory networks. BioSystems 2009 Nov;98(2):91-104.

(43) Klarner H, Siebert H, Bockmayr A. Parameter inference for asynchronous logical networks using discrete time series. Proceedings of the 9th International Conference on Computational Methods in Systems Biology New York, NY, USA: ACM; 2011.

(44) Saez-Rodriguez J, Alexopoulos LG, Epperlein J, Samaga R, Lauffenburger DA, Klamt S, et al. Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction. Mol Syst Biol 2009;5:331.

(45) Kauffman. Random Boolean network models and the yeast transcriptional network. Proceedings of the National Academy of Sciences - PNAS 2003;100(25):14796.

(46) Akustsu T, Kuhara S, Maruyama O, Miyano S. Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions. Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms Philadelphia, PA, USA: Society for Industrial and Applied Mathematics; 1998.

(47) Akutsu T, Kuhara S, Maruyama O, Miyano S. A System for Identifying Genetic Networks from Gene Expression Patterns Produced by Gene Disruptions and Overexpressions. Genome Inform Ser Workshop Genome Inform 1998;9:151-160.

(48) Akutsu T, Hayashida M, Ching W, Ng MK. Control of Boolean networks: Hardness results and algorithms for tree structured networks. J Theor Biol 2007 2/21;244(4):670-679.

(49) Harvey I, Bossomaier T. Time Out of Joint: Attractors in Asynchronous Random Boolean Networks. Proceedings of the Fourth European Conference on Artificial Life (ECAL97: MIT Press; 1997.

(50) Davidich, MI, Bornholdt, S. Boolean Network Model Predicts Cell Cycle Sequence of Fission Yeast. PLoS ONE 2008 02;3(2):e1672.

(51) Kauffman SA. The Origins of Order: Self-Organization and Selection in Evolution. 1st ed.: Oxford University Press, USA; 1993.

(52) Covert MW, Schilling CH, Palsson B. Regulation of gene expression in flux balance models of metabolism. J Theor Biol 2001 Nov 7;213(1):73-88.

(53) Sistla AP, Clarke EM. The complexity of propositional linear temporal logics. J.ACM 1985 July;32(3):733-749.

(54) Esparza J. Decidability and complexity of Petri net problems — An introduction. 1998;1491:374-428.

(55) McMillan KL, Probst DK. A technique of state space search based on unfolding. Formal Methods Syst Des 1995;6(1):45-65.

(56) Calzone L, Tournier L, Fourquet S, Thieffry D, Zhivotovsky B, Barillot E, Zinovyev A. Mathematical modelling of cell-fate decision in response to death receptor engagement. PLoS Computational Biology 2010;6(3)

(57) Naldi A, Carneiro J, Chaouiya C, Thieffry D. Diversity and plasticity of Th cell types predicted from regulatory network modelling. PLoS Computational Biology 2010;6(9)

(58) Ingram P, Stumpf M, Stark J. Network motifs: structure does not determine function. BMC Genomics 2006;7(1):108.

(59) Bentele M, Lavrik I, Ulrich M, Stosser S, Heermann DW, Kalthoff H, et al. Mathematical modeling reveals threshold mechanism in CD95-induced apoptosis. J Cell Biol 2004 Sep 13;166(6):839-851.

(60) Lorenz EN. The essence of chaos. Seattle: University of Washington Press; 1993.

(61) Karp R. Reducibility Among Combinatorial Problems. Complexity of Computer Computations 1972:85-103.

(62) Avriel M. Nonlinear programming: analysis and methods. Mineola: Dover Publications; 2003.

(63) Brusco MJ, Stahl S, SpringerLink. Branch-and-bound applications in combinatorial data analysis. New York: Springer; 2005.

(64) Esparza J, Romer S, Vogler W. An improvement of McMillan's unfolding algorithm. Lecture Notes in Computer Science; Tools and Algorithms for the Construction and Analysis of Systems 1996;1055:87-106.

(65) Leo JC, Guo C, Woon CT, Aw SE, Lin VC. Glucocorticoid and mineralocorticoid cross-talk with progesterone receptor to induce focal adhesion and growth inhibition in breast cancer cells. Endocrinology 2004 Mar;145(3):1314-1321.

(66) Gillespie DT. Approximate accelerated stochastic simulation of chemically reacting systems. The Journal of Chemical Physics 2001 Jul;115:1716-1733.

(67) Halford SE, Marko JF. How do site-specific DNA-binding proteins find their targets? Nucleic Acids Res 2004 Jun 3;32(10):3040-3052.

(68) Terry LJ, Shows EB, Wente SR. Crossing the Nuclear Envelope: Hierarchical Regulation of Nucleocytoplasmic Transport. Science 2007;318(5855):1412-1416.

(69) Liu W, Lahdesmaki H, Dougherty ER, Shmulevich I. Inference of Boolean networks using sensitivity regularization. EURASIP J Bioinform Syst Biol 2008:780541.

(70) Huang S. On the intrinsic inevitability of cancer: from foetal to fatal attraction. Semin Cancer Biol 2011 Jun;21(3):183-199.

(71) Huang S, Ernberg I, Kauffman S. Cancer attractors: a systems view of tumors from a gene network dynamics and developmental perspective. Semin Cell Dev Biol 2009 Sep;20(7):869-876.

(72) Gershenson C. Updating Schemes in Random Boolean Networks: Do They Really Matter? eprint arXiv:nlin/0402006 2004 feb.

(73) Siggers T, Duyzend MH, Reddy J, Khan S, Bulyk ML. Non-DNA-binding cofactors enhance DNA-binding specificity of a transcriptional regulatory complex. Mol Syst Biol 2011 Dec 6;7:555.

(74) Margolin AA, Nemenman I, Basso K, Wiggins C, Stolovitzky G, Favera R, et al. ARACNE: An Algorithm for the Reconstruction of Gene Regulatory Networks in a Mammalian Cellular Context. BMC Bioinformatics 2006;7; 7(Suppl 1).

(75) Kauffman SA. Metabolic stability and epigenesis in randomly constructed genetic nets. J Theor Biol 1969 Mar;22(3):437-467.

(76) Ulitsky I, Gat-Viks I, Shamir R. MetaReg: a platform for modeling, analysis and visualization of biological systems using large-scale experimental data. Genome Biol 2008 Jan 2;9(1):R1.

(77) French CA, Jin X, Campbell TG, Gerfen E, Groszer M, Fisher SE, et al. An aetiological Foxp2 mutation causes aberrant striatal activity and alters plasticity during skill learning. Mol Psychiatry 2011 Aug 30.

(78) van Rijsewijk H, Bart R, Nanchen A, Nallet S, Kleijn RJ, Sauer U. Large-scale 13C-flux analysis reveals distinct transcriptional control of respiratory and fermentative metabolism in Escherichia coli. Mol Syst Biol 2011 03/29;7.

(79) Hwang PI, Wu HB, Wang CD, Lin BL, Chen CT, Yuan S, et al. Tissue-specific gene expression templates for accurate molecular characterization of the normal physiological states of multiple human tissues with implication in development and cancer studies. BMC Genomics 2011 Sep 1;12:439.

(80) Wong AH, Gottesman II, Petronis A. Phenotypic differences in genetically identical organisms: the epigenetic perspective. Hum Mol Genet 2005 Apr 15;14 Spec No 1:R11-8.

(81) Takahashi K, Yamanaka S. Induction of pluripotent stem cells from mouse embryonic and adult fibroblast cultures by defined factors. Cell 2006 Aug 25;126(4):663-676.

(82) Ben-Dor A, Shamir R, Yakhini Z. Clustering gene expression patterns. J Comput Biol 1999 Fall-Winter;6(3-4):281-297.

(83) Saeys Y, Inza I, Larranaga P. A review of feature selection techniques in bioinformatics. Bioinformatics 2007 Oct 1;23(19):2507-2517.

Appendix

Glossary

Stochasticity - The property of a system whose behavior depends on probabilities. In a model with stochasticity, a single initial state can evolve into several different trajectories, each with an associated probability. Stochastic models in the context of biological systems assume that stochasticity is a property that is derived from the physical reality of the system.

Local state - At any time point, the value representing the status of an entity in a network is its (local) state. For example, the state of a protein may indicate whether it is phosphorylated or not (a Boolean value), or the time since its last phosphorylation (a real value).

Global state - The combination of all the local states of a model at one time point.

Trajectory -In logical models, a trajectory is a sequence of global states that occur consecutively. In continuous models, a trajectory is the change of the level of an entity over time.

Synchronous model -A model wherein the time steps at which the global state changes are discrete and (usually) equally spaced. On each step, all the states are updated simultaneously, depending on the model's regulation functions and on the global state at the previous step. In asynchronous models, system changes are not confined to specific times and global states do not progress according to 'a common clock'. Time is often continuous, and entities may change their states at different times.

State transition graph - A directed graph whose nodes correspond to global states and in which an edge from node u to node v indicates that the system can traverse directly from the global state represented by u to the global state represented by v.

ChIP on chip - A high-throughput experimental technique that gives indication for the binding intensities of transcription factors to DNA sequences.

PPI - High-throughput measurements of interactions between proteins.

Markov chain - A stochastic process in which the next state depends only on the present state, regardless of the trajectory that led to the present state.

Steady state - A global state that, once reached, always repeats itself in a trajectory. Another important dynamic behavior in biological systems is a cycle of global states. For example, the oscillations observed in the cell cycle.

Discretization - A process that transforms continuous numerical values into discrete ones. For example, real-valued measurements can be discretized to 0,1 or 2, corresponding to low, medium and high levels.

Time series data – Experimental measurements from several consecutive time points.

ביבליוגרפיה

(1) Matys V, Kel-Margoulis OV, Fricke E, Liebich I, Land S, Barre-Dirrie A, et al. TRANSFAC® and its module TRANSCompel®: transcriptional gene regulation in eukaryotes. Nucleic Acids Research ;34(suppl 1):D108-D110.

(2) Lachmann A, Xu H, Krishnan J, Berger SI, Mazloom AR, Ma'ayan A. ChEA: transcription factor regulation inferred from integrating genome-wide ChIP-X experiments. Bioinformatics 2010 Oct 1;26(19):2438-2444.

(3) Xie Z, Hu S, Blackshaw S, Zhu H, Qian J. hPDI: a database of experimental human protein–DNA interactions. Bioinformatics 2010 January 15;26(2):287-289.

(4) Karlebach G, Shamir R. Modelling and analysis of gene regulatory networks. Nat Rev Mol Cell Biol 2008 Oct;9(10):770-780.

(5) Karlebach G, Shamir R. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case. BMC Syst Biol 2010 Feb 25;4:15.

(6) Karlebach G, Shamir R. Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy-Based Approach. J Comput Biol 2012 Jan;19(1):30-41.

(7) Karlebach G, Shamir R. A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets. Submitted 2011.

הניתן. אנו מראים כי בעיה זו היא שלמה ב NP, ולכן קרוב לוודאי שלא קיים אלגוריתם יעיל לפתרונה. אנו מציגים ניסוח הסתברותי של הבעיה שעוקף דיסקרטיזציה של ערכי ביטוי, ומנסחים אותה כבעיית מינימיזציה של אנטרופיה. אנו מפתחים אלגוריתם היוריסטי למציאת פתרון ומדגימים אותו על מדידות מתאים עובריים של עכבר. המודל שאנו משחזרים הנו בעלי תאימות גבוהה לידע ביולוגי, ובפרט מפגין אחידות בין אופן הבקרה של גנים המבוקרים על ידי אותם פקטורי שעתוק.

4. A Fast Randomized Unfolding Algorithm for Solving Reachability Problems on Petri Nets

Guy Karlebach and Ron Shamir

Submitted (7)

רשתות פטרי הינן פורמליזם למידול מערכות מקביליות. ההחלטה האם רשת פטרי נתונה יכולה להגיע למצב מסוים או קבוצת מצבים היא בעיה יסודית בתחום. האלגוריתם של מקמילן (McMillan) מייצר ייצוג קומפקטי של מרחב המצבים של רשת פטרי. עם זאת, האלגוריתם יכול לפתור בפועל רק בעיות עבור רשתות קטנות, עקב העלות החישובית הגבוהה. אנו פיתחנו אלגוריתם מונטה קרלו המבוסס על האלגוריתם של מקמילן על מנת לפתור את בעיית הישיגות ברשתות פטרי. האלגוריתם חוזר מספר פעמים על בניית ייצוג של חלק אקראי ממרחב המצבים ובכך נמנע מהבעיות החישוביות שכרוכות בבניית ייצוג של המרחב כולו. הבדיקות שערכנו מראות שהאלגוריתם ההסתברותי מסוגל לפתור בעיות מעל גודל 100 בתוך שניות, והוא מהיר מהאלגוריתם הדטרמיניסטי בסדרי גודל על בעיות מסוימות.

2. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case

Guy Karlebach and Ron Shamir,

Published in *BMC Systems Biology*(5).

מודלים מתמטיים של רשתות ביולוגיות הוא חלק בלתי נפרד מביולוגיה של מערכות. פיתוח מודלים כאלו ושימוש בהם לחיזוי התנהגות רשתות הוא אתגר מרכזי בתחום. בעבודה זו אנו מציגים אלגוריתם אשר קובע את מספר הפרטורבציות המינימאלי הנדרש לשנות את ההתנהגות הדינמית של רשת בקרת גנים המתוארת כרשת פטרי, כאשר מטרת השינוי היא להשרות פנוטיפ ספציפי או להימנע ממנו. על ידי שינוי האלגוריתם של מקמילן אנו מתמודדים עם מידע חלקי ומפחיתים את העלות החישובית הכרוכה בכך. המתודולוגיה מודגמת על רשת גליומה. מתוך הפרטורבציות של גן בודד, אקטיבציה של הגן GSTP1 היא היעילה ביותר בבלימת הפנוטיפ הסרטני. מתוך זוגות של פרטורבציות, הגנים NFkB ו TGF-ß היו בעלי ההשפעה המשותפת הגדולה ביותר, בהתאמה לתפקידם בתהליך ה EMT. השיטה שפיתחנו מאפשרת ניתוח פרטורבציות של רשת הידועה רק באופן חלקי. היא יכולה לסייע בזיהוי מטרות לתרופות ובקביעת עדיפות בין ניסויי פרטורבציה.

3. Constructing Logical Models of Gene Regulatory Networks by Integrating Transcription Factor-DNA Interactions with Expression Data: An Entropy Based Approach

Guy Karlebach and Ron Shamir

Published in *Journal of Computational Biology*(6).

מודלים של רשתות בקרת גנים נועדו להסביר את התהליכים המורכבים המושלים בהתנהגות התא כגון דיפרנציאציה, מטבוליזם ומחזור התא. השפע של מידע ניסויי רחב היקף מאפשר לחוקרים להתאים מודלים תיאורטיים למדידות רמות ביטוי של גנים. מודל רשת שכיח הוא המודל הלוגי הדורש המרה של ערכים מדידה רציפים לערכים בדידים ולעתים קרובות כרוך בהופעה של אי התאמות בין המדידות למודל. דימיטרובה (Dimitrova) ואחרים הציגו את בעיית פתרון אי ההתאמות בצורה חסכנית ויעילה ככל

תקציר המאמרים הכלולים בתזה

עבודה זו מסתמכת על ארבעה מאמרים אשר התפרסמו בכתבי עת מדעיים או הוגשו לפרסום.להלן פירוט תקצירי המאמרים:

1. Modeling and Analysis of Regulatory Networks

Guy Karlebach and Ron Shamir

Published in *Nature Reviews Molecular Cell Biology*(4)

מודלים חישוביים רבים פותחו לצורך מידול רשתות בקרת גנים. ניתן לחלק מודלים אלו לשלושה סוגים עיקריים. הסוג הראשון, מודלים לוגיים, מתאר רשתות בקרה באופן איכותי. מודלים לוגיים מאפשרים למשתמשים להשיג הבנה של פונקציות שונות של רשת נתונה תחת תנאים שונים. אופיים האיכותי עושה אותם גמישים ונוחים להתאמה לתופעות ביולוגיות, אך יחד עם זאת הם יכולים לענות רק על שאלות איכותיות. כדי להבין ולהשפיע על התנהגויות התלויות בתזמון עדין יותר וריכוזי מולקולות רציפים פותח הסוג השני של מודלים– מודלים רציפים. על מנת לדמות למשל את השפעת הגבלת כמות הקלוריות על תאים יש צורך בתאור ברמה של מודל רציף. סוג שלישי של מודלים הוצע לאחר שהובן שברשתות בקרה ישנו אפקט של סטוכסטיות. מכיוון שמודלים מהסוג האחרון מתארים לרוב את האינטראקציות בין מולקולות בודדות הם נקראים מודלים ברמת המולקולה הבודדת. מודלים אלו חוקרים את הקשר בין רעש ובקרה של גנים. בעבודה זו אנו סוקרים את המתודולוגיות הקיימות למידול וניתוח רשתות בקרה. אנו מתארים הצלחות של קבוצות שונות בתיאור תופעות ביולוגיות ובניתוח בקרת ביטוי גנים. אנו משווים את המתודולוגיות, דנים ביתרונות והמגבלות של כל גישה, ומונים את השאלות הפתוחות בתחום, כגון מהו הקשר בין מבנה ופונקציה, כיצד אורגניזמים משתמשים ברשתות בקרה על מנת להתאים עצמם לסביבתם ומהי מהות האינטראקציה עם רשתות אחרות כמו רשתות מטבוליות.

את תכונותיהם. לעומת זאת גם התחזיות שהם מייצרים נתונות על ידי מספר קטן של ערכים דיסקרטיים ולכן מתאימות ביותר לאמירות כלליות לגבי התנהגות הרשת. כמו כן המיפוי בין מדידות ניסוייות למודל דורש טרנספורמציה.

סוג נוסף של מודלים הוא מודלים רציפים, בהם רמת הביטוי של גן היא ערך רציף והבקרה מתבצעת ביחס לציר זמן רציף. למרות שמודלים אלו אינם דורשים טרנספורמציה של מדידות ניסוייות עליהם להשתמש בכלים חישוביים כמו אינטרפולציה ורגרסיה על מנת להכליל מדידות כאלו. אחד הפנים האטרקטיביים בגישה הרציפה היא שהשימוש במשוואות דיפרנציאליות לצורך תאור מערכות ביוכימיות נחל הצלחה בתחום מידול של מערכות מטבוליות קטנות, ונשען על תיאוריה פיסיקלית. למרות זאת הכללה לרשתות גדולות מציבה קשיים חישוביים וניסויים כאחד. במערכת כללית של משוואות דיפרנציאליות נעשה לרוב שימוש באינטגרציה נומרית על מנת לדמות את התנהגות הרשת.

המודלים הקרובים ביותר למציאות הפיסיקלית הם מודלים ברמת המולקולה הבודדת. מודלים אלה אומצו על מנת לייצג תכונת סטוכסטיות של רשתות בקרת גנים. בעוד שמשוואות דיפרנציאליות מדייקות כאשר הן מתארות ריאקציות בין מספר גדול של מולקולות, כאשר מספר המולקולות קטן התנהגות הרשת מכילה גורם אקראי. למרות המורכבות הרבה של תאור רשת בקרה באופן זה, הושגו הצלחות מרשימות, למשל, בהבנת מחזור ההדבקה של וירוסים שבו החלטות על שהייה במצב רדום או מעבר למצב פעיל תלויות ברשת בקרה עם אלמנטים סטוכסטיים. החיסרון בגישה זו הוא יכולת מוגבלת להתאמה למידע ניסויי והתלות בסימולציות לצורך חקר התנהגות הרשת.

ביעילות בכל מצב. לפיכך יש להגביל את היעדים של הניתוח או לגלות אלגוריתמים בעלי ביצועים טובים במצבים מסוימים ולאפיין מצבים אלו.

הסקת מבנה הרשת

שיטות רבות ומימושן פותחו על מנת להסיק מבנה של רשת בקרת גנים ממדידות רמות הביטוי שלהם. התחרות השנתית DREAM נוסדה בשאיפה לעודד קידום מטרה זו ולייצר דיאלוג בין קבוצות המחקר השונות. אחת הבעיות המרכזיות בנתונים משבבי דנ"א היא שהסקת סיבתיות ממדידות המופרדות זו מזו בפרק זמן מוטה לשגיאה ותופעה זו מחמירה ככל שפרקי הזמן המפרידים גדולים יותר כיוון ששבבי דנ"א מודדים ביטוי במספר נקודות זמן בלבד קשה להפריד בין השפעות ישירות ועקיפות בין גנים רק על סמך רמות הביטוי.

ChIP on chip היא טכנולוגיה שעושה שימוש בשבבי דנ"א לצורך מציאת רצפי הקישור הגנומיים של פקטורי שעתוק מאפשרת לקבוע את הבקרה בין גנים ביתר וודאות. בשלב הראשון מבצע החוקר קישור כימי בין פקטור השעתוק והדנ"א, לאחר מכן משרה פרוק של רצפי הדנ"א שאינם קשורים והרצפים הנותרים עוברים היברידיזציה עם שבב דנ"א. למרות שהמדידות יכולות להצביע על קישור שאינו משפיע על רגולציה ויכולות להכיל רעש ניסויי, שילובן עם מדידת ביטוי גנים מאפשרת חיזוק הדדי של שני סוגי המידע והסקת סוג ההשפעה של הבקרה, למשל הפעלה או דיכוי(השתקה).

בנוסף לטכנולוגיית ChIP on chip ישנו מידע רב על קישור פקטורי שעתוק במאגרי נתונים ביולוגיים(3-1). אמנם מידע זה מגיע לרוב ממקורות מגוונים, אך השוואתו לסוגי המידע האחרים מאפשרת לסנן טעויות או אינפורמציה שהינה ספציפית למערכת ניסויית מסוימת. לפיכך, האתגר המרכזי בהסקת מבנה הרשת הוא פירוש נכון של מידע ניסויי ושילוב מידע ממקורות שונים.

מודלים מתמטיים של רשתות

ניתן לחלק את המודלים השונים של רשתות בקרת גנים לשלושה סוגים עיקריים. הסוג הראשון נקרא מודלים לוגיים והוא כולל מודלים המשתמשים בקבוצה קטנה של ערכים בדידים ובפונקציות לוגיות כדי לתאר את ערכי הביטוי של גנים ואופן שינויים. תכונה זו מאפשרת להתאים להם מדידות כאשר היכולת לתקן רעש מוגבלת ולנצל כלים ממדעי המחשב כדי לנתח

גנים בעלי תפקידים אחרים בתא. רשת שהבנתה הייתה בעלת השפעה מכרעת על התפתחות התחום היא רשת ההתפתחות העוברית בקיפוד ים. מודל הרשת שהורכב במעבדתו של אריק דייוידסון (**Eric Davidson**) במשך שני עשורים ממחיש כיצד סיגנלים אימהיים הנוכחים בעובר המתפתח מפורשים על ידי רשת בקרת גנים ומתורגמים לתבניות ביטוי שונות בתאי בת שונים של הביצית המופרית. דוגמא קצת שונה היא מחזור התא תאים יכולים להתחלק באופן מחזורי ללא סיגנל חיצוני, וזאת על ידי רשת בקרת גנים שמכתיבה תבנית ביטוי מחזורית. על מנת לתרגם דוגמאות מפתח אלו להבנה כוללת של רשתות בקרה ועל מנת לנצל הבנה זו לצרכים רפואיים, יש צורך ביכולת אוטומטית לבנות ולחזות את התנהגותן.

שיטות חישוביות רבות הוצעו לצורך ניצול מידע ניסויי במודלים של רשת בקרת גנים. ניתן להציב שיטות אלו על ציר אשר בקצהו האחד מורכבות ובקצהו השני יכולת ניתוח יעילה. כלומר, ככל שהמודל הנבנה מורכב יותר, כך יותר קשה לנתח את התנהגותו ולהתאים אליו מידע ניסויי. מצד שני מודל פשוט מדי עלול לאבד אלמנטים חיוניים להסברת התנהגות הרשת. אחד המודלים שהינם יחסית פשוטים ובעלי יכולת תאור טובה הם רשתות בוליאניות. ברשתות אלו רמת הביטוי של גנים נבחרת מהערכים 0 או 1, והבקרה מבוטאת על ידי פונקציות בוליאניות. אלגוריתמים הממפים מידע הנמדד בשבבי דנ"א לרשתות בוליאניות כבר פותחו לפני כיותר מעשור. שיפור בשיטות אלו מהווה כיוון מבטיח להשגת היעד של הבנת הבקרה של ביטוי גנים.

אתגרים חישוביים בניתוח רשתות

אחת הבעיות בניתוח רשתות שנלמדו לעומק עוד לפני עידן שבבי הדנ"א היא גדילה אקספוננציאלית של מספר מצבי הרשת האפשריים. בבעיה זו נתקלו מדעני המחשב שפיתחו את התשתית החישובית לאיתור התנהגויות חריגות של שבבי מחשב, והיא מעסיקה קהילה זו עד עצם היום הזה. בהקשר הביולוגי, מכיוון שכל גן יכול באופן פוטנציאלי להיות משופעל או לא משופעל, מספר הצירופים שנוצר ומגדיר את מצבי הרשת האפשריים ולפיכך את הפנוטיפ הוא גדול מאוד. גם אם בוחנים רק את הגנים בעלי תפקיד רגולטורי בעיה זו עדיין מגבילה מאוד את יכולת הניתוח של הרשת.

התיאוריה של מדעי המחשב מציבה גבולות ליכולת הניתוח של המופעים הקשים ביותר של הבעיה, ולרוב כשאפיון זה מצביע על עלות חישובית גבוהה הוא חל על הרבה מופעים אחרים. תיאוריה זו אכן מראה שקרוב לוודאי לא ניתן יהיה לפתח אלגוריתמים לניתוח רשתות שיעבדו

תקציר

רקע כללי

בעשרות השנים האחרונות חשפה הביולוגיה המולקולארית מערכות בעלות מורכבות מופלאה המתקיימות בעולם המיקרוסקופי וממלאות תפקידי מפתח בתופעת החיים. יחד עם התקדמות זו הובן שעל מנת לשלוט במערכות אלו באופן יעיל, בראש ובראשונה לצרכים רפואיים, יש צורך ביכולת ניתוח שתאפשר חיזוי של התנהגויות שונות, ממש כמו במערכות מעשה ידי אדם.

שלא כמערכות שיצר האדם, רבים מהרכיבים של מערכות ביולוגיות אינם ידועים מראש ויש צורך בניתוח מידע ניסויי על מנת להסיק אותם. אחת השיטות הניסוייות הנפוצות ביותר היא טכנולוגית שבבי דנ"א. טכנולוגיות זו מאפשרת למדוד את רמות הביטוי של עשרות אלפי גנים בו זמנית. העיקרון עליו מבוססת הטכנולוגיה הוא היברידיזציה של תוצרי הגנים עם גלאים המהווים רצף בסיסי דנ"א משלים עבורם. החוקר מפיק רנ"א מאוכלוסיית תאים, מסמן אותו באופן פלורסנטי ומאפשר לו לעבור היברידיזציה עם הגלאים על שבב הדנ"א. שבבי דנ"א גם מאפשרים לסרוק את הגנום לצורך מציאת רצפי דנ"א אליהם נקשרים פקטורי שעתוק, בשיטה הנקראת **ChIP on chip**. בעזרת הטכנולוגיות הנ"ל מנסים החוקרים להסיק את מערכת האינטראקציות המורכבת בין פקטורי שעתוק וגנים האחראית לתופעות מורכבות כמו מחזור התא, התפתחות עוברית ושעונים ביולוגיים.

רמת התיאור הבסיסית ביותר של מערכת אינטראקציות בין מולקולות היא גרף מכוון, כאשר חץ ממולקולה א' למולקולה ב' מייצג סיבתיות, כלומר א' משפיעה על ב'. עם השתרשותה של התפיסה המערכתית הורכבו מודלים כאלו על מנת לאגד את הידע שנאסף במעבדות ביולוגיות רבות. למרות ההבנה הבסיסית שהם מספקים לגבי תפקודה של המערכת, אין הם מאפשרים לבצע חיזוי של תוצאות ניסויים, מכיוון שהם מתארים מבנה אך לא את אופן הפעולה שלו. בנוסף, ברוב המקרים הידע שנאסף בגישה הרדוקטיבית, בה כל רכיב נלמד בנפרד, חסר רכיבים החיוניים להסבר התנהגות המערכת.

רשתות בקרת גנים

תהליכים ביולוגיים המתבצעים בדיוק מרשים,כמו, למשל, התפתחות עוברית, הם אחד התוצרים המופלאים של התהליך האבולוציוני. תהליכים אלו נשלטים על ידי רשתות בקרת גנים אשר מורכבות מגנים בעלי תפקיד רגולטורי המבקרים את רמת הביטוי הן של גנים רגולטוריים והן של

תמצית

התיאוריה המסבירה את תבניות הביטוי של גנים בתאים ע"י רשתות בקרה הועלתה לפני כמה עשורים, ובכך הקדימה את היכולות הטכנולוגיות של העשור האחרון. כעת המצב הפוך מידע ניסויי רב נאגר ויש צורך ליצור יכולות חישוביות שיאפשרו להסביר אותו.

בעוד שתאור רשתות ברמת הריאקציה הכימית ניצב בפני קשיים פרקטיים ותיאורטיים כאחד, רבים סבורים כי מודל לוגי מופשט יותר מסוגל לשחזר את התנהגותה של רשת בקרת גנים בצורה מספקת לצורך הבנה עקרונית של אופן פעולתה וביצוע תחזיות. ישנם מספר אתגרים חישוביים שיש לענות עליהם על מנת לבנות מודל כזה. ראשית היכולת לבחור את זמן המדידה ביחס לתחילת פעולתה של הרשת הוא מוגבל, שכן רשתות ביולוגיות הן מערכות שלא ניתן לאתחל ולעצור כרצוננו. בנוסף, מספר המדידות מוגבל ולכן בדרך כלל עומדות לרשות החוקר מספר נקודות זמן שיש לקשר ביניהן. אתגר נוסף הוא למפות בין רמת המדידה לרמה לוגית, כלומר בין ערך שהינו לרוב רציף או הנלקח מקבוצה גדולה של ערכים אפשריים לערך שהוא בד"כ בינארי או טרינארי. לבסוף, פרט לרעש הניסויי האינהרנטי רוב הטכנולוגיות הקיימות מודדות ערכים מאוכלוסיה של תאים ולכן כל מדידה הינה ממוצע של מצבן של רשתות רבות ולא מצב רשת בתא מסוים.

שיטות רבות פותחו או הושאלו מתחומים שונים על מנת להתמודד עם אתגרים אלו בהצלחה. למרות דיווחים על מודלים בעלי יכולת חיזוי טובה עדיין אין תמימות דעים לגבי שיטת המידול המיטבית.

בעבודה זו אני מאפיין את הבעיות העכשוויות בניתוח ביולוגי רשתות בקרת גנים ומציע מספר שיטות שנועדו להתמודד עם בעיות אלו. קביעת המבנה של רשת אינו יכול להתבצע באופן ניסויי בלבד ודורש ניתוח מורכב של התוצאות. כמו כן, אני מראה כי בהינתן מבנה רשת הבעיה של התאמת מודל לוגי עקבי עם נתוני ביטוי גנים היא **NP** קשה. אני מציע אלגוריתם קביעת מודל לוגי העוקף כמה מהקשיים החישוביים, ואלגוריתם נוסף שמסוגל לבצע תחזיות למרות חוסר וודאות לגבי מבנה הרשת המדויק.

לחלק מהבעיות בניתוח רשתות קיימות מקבילות בתחום של אימות חומרה ותוכנה.אני מפתח אלגוריתם להאצת שיטות קיימות באימות חומרה ומיישם אותן למידול רשתות בקרת גנים.

אוניברסיטת תל-אביב TEL AVIV UNIVERSITY

הפקולטה למדעים מדויקים ע"ש ריימונד ובברלי סאקלר

בית הספר למדעי המחשב ע"ש בלבטניק

בניית מודלים לרשתות בקרת גנים וניתוח פרטורבציות בהן

חיבור לשם קבלת תואר "דוקטור לפילוסופיה"

מאת גיא קרליבך

בהנחייתו של פרופ' רון שמיר

הוגש לסנאט של אוניברסיטת ת"א

יולי 2012