40

REVERSE ENGINEERING OF MOLECULAR NETWORKS FROM A COMMON COMBINATORIAL APPROACH

Bhaskar DasGupta, Paola Vera-Licona, and Eduardo Sontag

40.1 INTRODUCTION

The understanding of molecular cell biology requires insight into the structure and dynamics of networks that are made up of thousands of interacting molecules of DNA, RNA, proteins, metabolites, and other components. One of the central goals of systems biology is the unraveling of the as yet poorly characterized complex web of interactions among these components. This work is made harder by the fact that new species and interactions are continuously discovered in experimental work, necessitating the development of adaptive and fast algorithms for network construction and updating. Thus, the "reverse engineering" of networks from data has emerged as one of the central concern of systems biology research.

A variety of reverse-engineering methods have been developed, based on tools from statistics, machine learning, and other mathematical domains. To use these methods effectively, it is essential to develop an understanding of the fundamental characteristics of these algorithms. With that in mind, this chapter is dedicated to the reverse engineering of biological systems.

Algorithms in Computational Molecular Biology, Edited by Mourad Elloumi and Albert Y. Zomaya
Copyright © 2011 John Wiley & Sons, Inc.

Specifically, we focus our attention on a particular class of methods for reverse engineering, namely those that rely algorithmically on the so-called “hitting set” problem, which is a classical combinatorial and computer science problem, Each of these methods uses a different algorithm to obtain an exact or an approximate solution of the hitting set problem. We will explore the ultimate impact that the alternative algorithms have on the inference of published *in silico* biological networks.

40.2 REVERSE-ENGINEERING OF BIOLOGICAL NETWORKS

Systems biology aims at a systems-level understanding of biology, viewing organisms as integrated and interacting networks of genes, proteins, and other molecular species through biochemical reactions that result in particular form and function (phenotype). Under this “system” conceptualization, it is the interactions among components that give rise to emerging properties.

Systems-level ideas have been a recurrent theme in biology for several decades, as exemplified by Cannon’s work on homeostasis [7], Wiener’s biological cybernetics [31], and Ludwig von Bertalanffy’s foundations of general systems theory [30]. So what has brought systems biology to the mainstream of biological science research in recent years? The answer can be found in large part in enabling technological advances, ranging from high-throughput biotechnology (gene expression arrays, mass spectrometers, *etc.*) to advances in information technology, that have revolutionized the way that biological knowledge is stored, retrieved, and processed.

A systems approach to understanding biology can be described as an iterative process that includes; (i) data collection and integration of all available information (ideally, regarding all the components and their relationships in the organism of interest), (ii) system modeling, (iii) experimentation at a global level, and (iv) generation of new hypotheses (see Figure 40.1).

The current chapter focuses on the system modeling aspects and, specifically, on the top-down modeling approach broadly known as the biological “reverse-engineering,” which can be very broadly described as follows:

The biological reverse-engineering problem is that of analyzing a given system in order to identify, from biological data, the components of the system and their relationships.

In broad terms, there are two very different levels of representation for biological networks. They are described as follows.

NETWORK TOPOLOGY REPRESENTATIONS. Also known as “wiring diagrams” or “static graphs,” these are coarse diagrams or maps that represent the connections (physical, chemical, or statistical) among the various molecular components of a network. At this level, no detailed kinetic information is included. A network of molecular interactions can be viewed as a graph: Cellular components are nodes in a network, and the interactions (binding, activation, inhibition, *etc.*) between these components are the edges that connect the nodes. A reconstruction of network

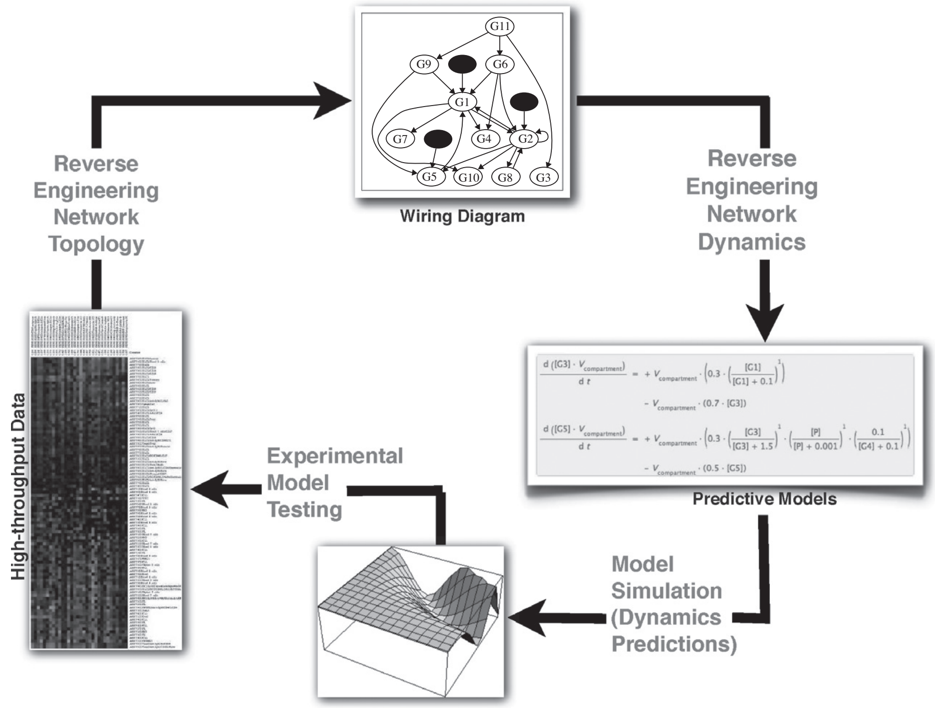
G11
G9
G6
G1
G7
G4
G2
G5
G10
G8
G3
Wiring Diagram
Reverse Engineering Network Topology
Reverse Engineering Network Dynamics
High-throughput Data
Predictive Models
Experimental Model Testing
Model Simulation (Dynamics Predictions)

Figure 40.1 Iterative process in systems biology.

topology allows one to understand properties that might remain hidden without the model or with a less relevant model.

These type of models can be enriched by adding information on nodes or edges. For instance, "+" or "−" labels on edges may be used in order to indicate positive or negative regulatory influences. The existence of an edge might be specified as being conditional on the object being studied (for instance, a cell) being in a specific global state, or on a particular gene that regulates that particular interaction being expressed above a given threshold. These latter types of additional information, however, refer implicitly to notions of state and temporal evolution, and thus, they lead naturally toward qualitative dynamical models.

Different reverse-engineering methods for topology identification differ on the types of graphs considered. For example, in the work in [3, 9, 11, 24, 26, 27, 32, 33], edges represent statistical correlation between variables. In [10, 13, 15, 17], edges represent causal relationships among nodes.

NETWORK DYNAMICAL MODELS. Dynamical models represent the time-varying behavior of the different molecular components in the network and, thus, provide a more accurate representation of biological function.

Models can be used to simulate the biological system under study. Different choices of values for parameters correspond either to unknown system characteristics or to environmental conditions. The comparison of simulated dynamics with experimental measurements helps refine the model and provide insight on qualitative properties of behavior, such as the identification of steady states or limit cycles, multistable (*e.g.*, switch-like) behavior, the characterization of the role of various parts of the network in terms of signal processing (such as amplifiers, differentiators and integrators, and logic gates), and the assessment of robustness to environmental changes or genetic perturbations.

Examples of this type of inference include those leading to various types of Boolean networks [2, 20–22] or systems of differential equations [12, 16, 28], as well as multistate discrete models [19].

Depending on the type of network analyzed, data availability and quality, network size, and so forth, the different reverse-engineering methods offer different advantages and disadvantages relative to each other. In Section 40.3.1, we will explore some of the common approaches to their systematic evaluation and comparison.

40.2.1 Evaluation of the Performance of Reverse-Engineering Methods

The reverse-engineering problem is by its very nature highly "ill-posed," in the sense that solutions will be far from unique. This lack of uniqueness stems from the many sources of uncertainty: measurement error, lack of knowledge of all the molecular species that are involved in the behavior being analyzed ("hidden variables"), stochasticity of molecular processes, and so forth. In that sense, reverse-engineering methods can at best provide approximate solutions for the network that one wishes to reconstruct, making it very difficult to evaluate their performance through a theoretical study. Instead, their performance is usually assessed empirically, in the following two ways:

Experimental testing of predictions. After a model has been inferred, the newly found interactions or predictions can be tested experimentally for network topology and network dynamics inference, respectively.

Benchmarking testing. This type of performance evaluation consists on measuring how "close" the method of our interest is from recovering a known network, referred to as the **"gold standard"** for the problem. In the case of dynamical models, one evaluates the ability of the method of interest to reproduce observations that were not taken into account in the "training" phase involved in the construction of the model. On the another hand, for methods that only reconstruct the network topology (wiring diagram), a varierty of standard metrics may be applied.

METRICS FOR NETWORK TOPOLOGY BENCHMARKING. Suppose that Γ is the graph representing the network topology of a chosen "gold standard" network. Let Γ_i be the graph representing the inferred network topology. Each one of the interactions in Γ_i can be classified into one of the these four classes, when comparing with the gold standard:

(a) Correct interactions inferred (true positives, TP)

(b) Incorrect interactions inferred (false positives, FP)

(c) Correct noninteractions inferred (true negatives, TN)
(d) Incorrect noninteractions inferred (false negatives FN)

From this classification of the interactions, we compute the following metrics:

- The Recall or True Positive Rate TPR = TP/(TP + FN).
- The False Positive Rate FPR = FP/(FP + TN).
- The Accuracy ACC = (TP + TN)/Tot$I$, where Tot$I$ is the total number of possible interactions in a network.
- The Precision or Positive Predictive Value PPV = TP/(TP + FP).

As mentioned, the reverse-engineering problem is underconstrained. Every algorithm will have one or more free parameters that helps select a "best" possible prediction. Hence, a more objective evaluation of performance has to somehow involve a range of parameter values. One way to evaluate performance across ranges of parameters is the **receiver operating characteristic (ROC)** method, based on the plot of FPR vs. TPR values. The resulting **ROC plot** depicts relative trade-offs between true positive predictions and false positive prediction across different parameter values (see Figure. 40.2). A closely related approach is the **Recall-Precision plot**, obtained by plotting TPR vs. PPV values.

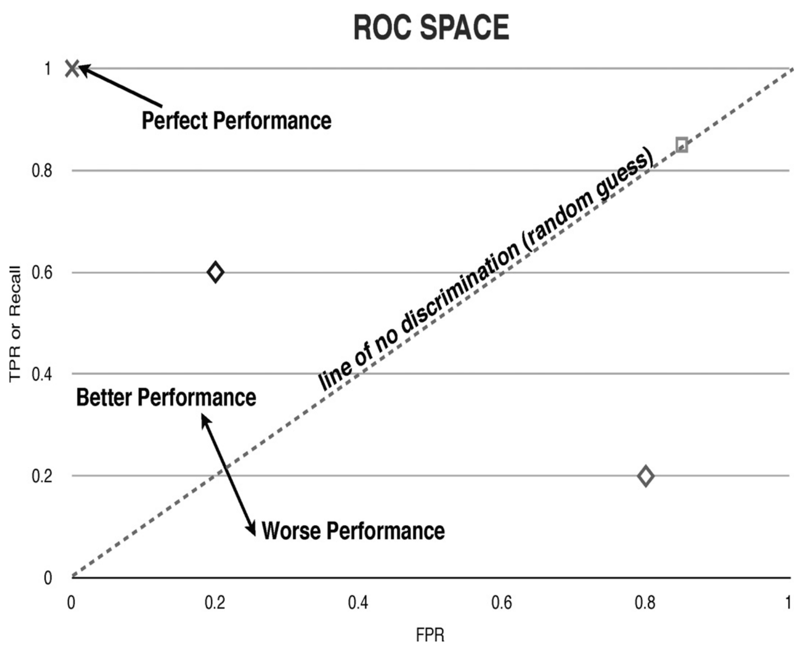

Figure 40.2 Receiver operating characteristic (ROC)-space. Defined by FPR vs. TPR values in a two-dimensional coordinate system: A perfect reverse-engineering method will ideally have score (FPR, TPR) = (0, 1), whereas the worst possible network will have coordinates (FPR, TPR) = (1, 0), and scores below the identity line (diagonal) indicate methods that perform no better than a random guess.

40.3 CLASSICAL COMBINATORIAL ALGORITHMS: A CASE STUDY

We have briefly discussed some basic aspects of reverse engineering of biological systems. Next, as a case of study, we focus our attention on some reverse-engineering algorithms that rely on the solution of the so-called "hitting set problem." The hitting set problem is a classical problem in combinatorics and computer science. It is defined as follows:

Problem 40.1 (HITTING SET Problem) *Given a collection $\mathcal{H}$ of subsets of $E = \{1, \ldots, n\}$, find the smallest set $L \subseteq E$ such that $L \cap \mathcal{X} \neq \emptyset$ for all $\mathcal{X} \in \mathcal{H}$.*

The hitting set problem is NP-hard, as can be shown via transformation from its dual, the (minimum) set cover problem [14].

We next introduce some reverse-engineering methods based on the hitting set approach.

- Ideker *et al.* [15].
  This paper introduces two methods to infer the topology of a gene regulatory network from gene expression measurements. The first "network inference" step consists of the estimation of a set of Boolean networks consistent with an observed set of steady-state gene expression profiles, each generated from a different perturbation to the genetic network studied. Next, an "optimization step" involves the use of an entropy-based approach to select an additional perturbation experiment in order to perform a model selection from the set of predicted Boolean networks. In order to compute the sparsest network that interpolates the data, Ideker *et al.* rely on the "minimum set cover" problem. An approximate solution for the hitting set problem is obtained by means of a branch and bound technique [25]. Assessment is performed "*in Numero*": The proposed method is evaluated on simulated networks with varying number of genes and numbers of interactions per gene.
- Jarrah *et al.* [13]
  This paper introduces a method for the inference of the network topology from gene expression data, from which one extracts state transition measurements of wild-type and perturbation data. The goal of this reverse-engineering algorithm is to output one or more most likely network topologies for a collection $x_1, \ldots, x_n$ of molecular species (genes, proteins, *etc.*), which we will refer to as variables. The state of a molecular species can represent its levels of activation. That is, each variable x_i takes values in the set $X = \{0, 1, 2, \ldots\}$ and the interactions among species indicate causal relationships among molecular species. The inference algorithm takes as input one or more time courses of observational data. The output is a most likely network structure for the interactions among $x_1, \ldots, x_n$ that is consistent with the observational data: The notion of consistency with observational data makes the assumption that the regulatory network for $x_1, \ldots, x_n$ can be viewed as a dynamical system

that is described by a function $f : X_n \to X_n$, which transforms an input state $(s_1, \ldots, s_n)$, $s_i \in X$, of the network into an output state $(t_1, \ldots, t_n)$ at the next time step. A directed edge $x_i \to x_j$ in the graph of the network topology of this dynamical system f indicates that the value of x_j under application of f depends on the value of x_i. Hence a directed graph is consistent with a given time course $s_1, \ldots, s_r$ of states in X_n, if it is the network topology of a function $f : X_n \to X_n$ that reproduces the time course; that is, $f(s_i) = s_{i+1}$ for all i.

One possible drawback of reverse-engineering approaches lies in the fact that they construct the "sparest" possible network consistent with the given data. However, real biological networks are known to be not minimal [29]. Although accurate measures of deviation from sparsity are difficult to estimate, nonetheless it seems reasonable to allow additional edges in the network in a "controlled" manner that is consistent with the given data. As already commented in [15], it is possible to add redundancies to the reverse-engineering construction. The basic hitting set approach provides only a minimal set of connections, whereas real biological networks are known to contain redundancies (*e.g.*, see [22]). To account for this, one can modify the hitting set approach to add redundancies systematically by allowing additional parameters to control the extra connections. Theoretically, in terms of the algorithm this corresponds to a standard generalization of the set-cover problem, known as the set-multicover problem, which is well studied in the literature, and for which approximation algorithms are known [4].

The search for the topologies that interpolate the input data involves directly the hitting set problem, which is solved analytically with the use of a computational algebra tools.

The algorithms presented in [5, 17] also make use of hitting set algorithms, but we will restrict our attention to the comparison of the two methods described above.

40.3.1 Benchmarking RE Combinatorial-Based Methods

40.3.1.1 In Silico Gene Regulatory Networks. We use data from two different regulatory networks. These contain some features that are common in real regulatory networks, such as time delays and the need for a measurement data presented into discrete states $(0, 1, 2, \ldots)$.

IN SILICO NETWORK 40.1: GENE REGULATORY NETWORK WITH EXTERNAL PERTURBATIONS. This network was originally introduced in [6]. It was generated using software package given in [23], the interactions between genes in this regulatory network are phenomenological, and represent the net effect of transcription, translation, and post-translation modifications on the regulation of the genes in the network. The model is implemented as a system of ODEs in *Copasi* [18].

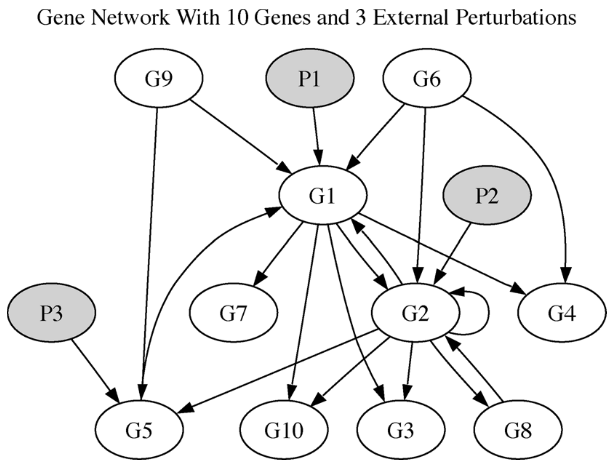

Figure 40.3 Network 1: 10 genes and 3 environmental perturbations. In this network, the 3 environmental perturbations P1, P2, and P3 directly affect the expression rate of genes G1, G2, and G5, respectively.

This network, shown in Figure 40.3, consists of 13 species: ten genes plus three different environmental perturbations. The perturbations affect the transcription rate of the gene on which they act directly (through inhibition or activation) and their effect is propagated throughout the network by the interactions between the genes.

Network 40.2: segment polarity genes network in *drosophila melanogaster*. The network of segment polarity genes is responsible for pattern formation in the *D. melanogaster* embryo (Figure 40.4). Albert and Othmer [1] proposed and analyzed a Boolean model based on the binary ON/OFF representation of mRNA and protein levels of five segment polarity genes. This model was constructed based on the known topology, and it was validated using published gene and expression data. We generated time courses from this model, from which we will attempt to reverse-engineer the network in order to benchmark the performance of the reverse-engineering algorithms being evaluated.

The network of the segment polarity genes represents the last step in the hierarchical cascade of gene families initiating the segmented body of the fruit fly. The genes of this network include engrailed (*en*), wingless (*wg*), hedgehog (*hh*), patched (*ptc*), cubitus interruptus (*ci*), and sloppy paired (*slp*), coding for the corresponding proteins, which are represented by capital letters (*EN, WG, HH, PTC, CI*, and *SLP*). Two additional proteins, resulting from transformations of the protein *CI*, also play important roles: *CI* may be converted into a transcriptional activator, *CIA*, or may be cleaved to form a transcriptional repressor *CIR*. The expression of the segment polarity genes occurs in stripes that encircle the embryo. These key features of these patterns can be represented in one dimension by a line of 12 interconnected cells, grouped into three parasegment primordia, in which the genes are expressed every fourth cell. In Albert and Othmer [1], parasegments are assumed to be identical, and thus only one parasegment of four cells is considered. Therefore, in the model, the

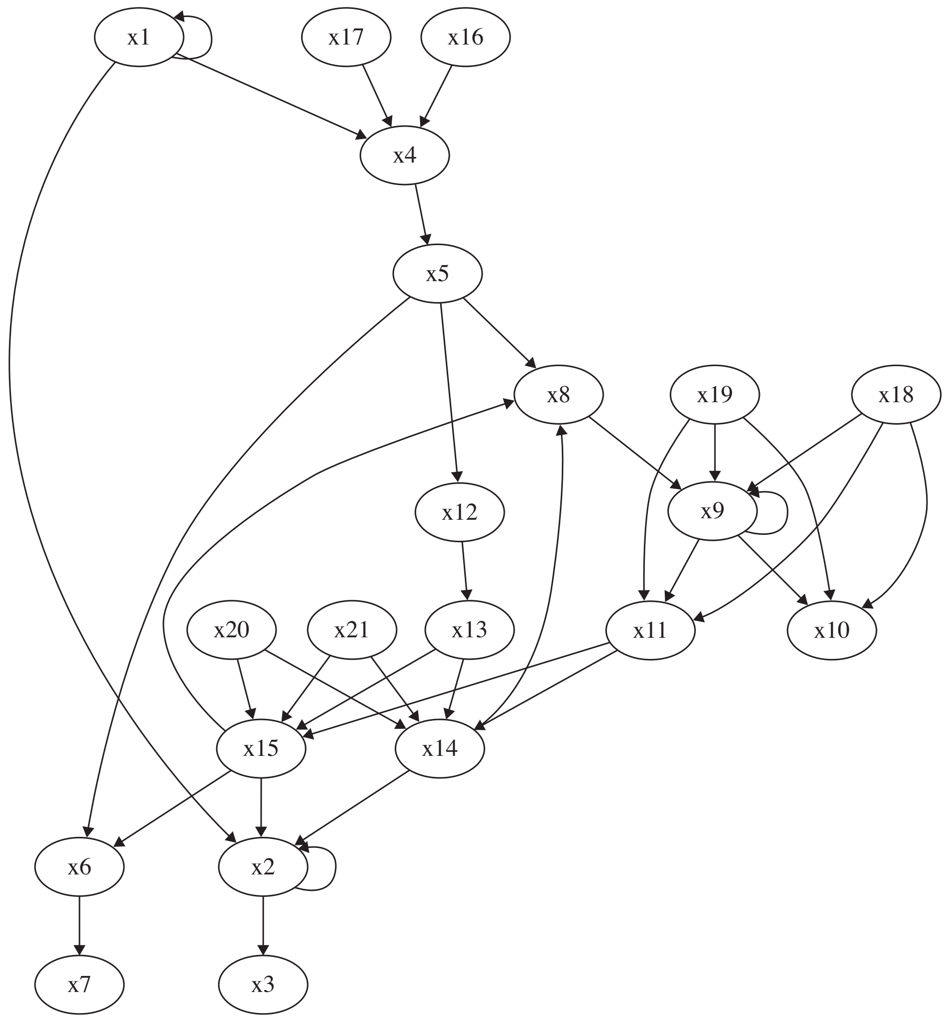

Figure 40.4 Segment polarity genes network on the *D. melanogaster*. This network consists of the interaction of 60 molecular species: genes and proteins.

variables are the expression levels of the segment polarity genes and proteins (listed above) in each of the four cells, and the network can be seen as a $15 \times 4 = 60$ node network. Using the wild-type pattern from [1], we consider one wild-type time series of length 23.

40.3.1.2 Results of Comparison. In this section we compare the results obtained after running Jarrah *et al.*'s and Ideker *et al.*'s methods on each of the above networks. Computations were made on Mac OS X, Processor 2GHz Intel Core 2 Duo.

As we mentioned in Section 40.3, for Jarrah *et al.*'s method, the input data must be discrete. Hence, in order to apply this reverse-engineering method to network 1, we discretize the input data, considering then different discretizations as our running parameter to test Jarrah *et al.*'s method in the ROC space. We specifically use

Table 40.1 Comparison of RE methods

	TP	FP	TN	FN	TPR	FPR	ACC	PPV
Network1								
Jarrah Exact Sol D7	12	34	113	10	.5454	.231	.7396	.2608
Jarrah Exact Sol Q5	9	49	98	13	.4090	.3334	.6331	.1551
Jarrah Exat Sol I5	7	46	101	15	.3181	.313	.6390	.1320
Karp Greedy Approx R1	9	49	98	13	.4090	.666	.016	.084
Karp Greedy Approx R2	11	63	84	11	.5	.571	.020	.115
Karp LP Approx. R1	7	46	101	15	.318	.687	.014	.064
Karp LP Approx. R2	9	59	88	13	.409	.598	.018	.092
Network 2								
Jarrah Exact Sol	–	–	–	–	–	–	–	–
Karp Greedy Approx R1	4	3321	91	124	.031	.026	.923	.042
Karp Greedy Approx R2	15	3254	218	113	.117	.062	.908	.064
Karp LP Approx. R1	3	3279	93	125	.023	.026	.939	.031
Karp LP Approx. R2	9	3285	187	119	.070	054	.915	.045

three discretization methods: a graph-theoretic based approached "D" (see [8]), as well as quantile "Q" (discretization method on which each variable state receives an equal number of data values) and interval "I" discretization (discretization method on which we select thresholds for the different discrete values).

For Ideker *et al.*'s method we have considered both greedy and linear programming approximations to the hitting set problem as well as redundancy values (how many extra edges one allows) of $R = 1$ or 2.

We have displayed some our results on Table 40.1. We observe that for network 1, Jarrah *et al.*'s method obtains better results than Ideker *et al.*'s method when considering these values in the ROC space, although both fare very poorly. On the other hand, we observe that Ideker *et al.*'s method achieves a performance no better than random guessing on this network. In contrast, for network 2, Jarrah *et al.*'s method could not obtain any results after running their method for more than 12 hours, but Ideker *et al.*'s method was able to compute results for such a network in less than 1 minute. Also Ideker *et al.*'s method improved slightly its results when the redundancy number is increased; this might indicate the shortcoming of inferring sparser networks when they are of a larger size containing redundancies.

40.3.2 Software Availability

The implementation of Jarrah *et al.*'s algorithm [13] is available online through the web interface provided at http://polymath.vbi.vt.edu/polynome/. The implementation of Ideker *et al.*'s algorithm [15] is available online through the web interface provided at http://sts.bioengr.uic.edu/causal/.

40.4 CONCLUDING REMARKS

In this chapter, we first provided a brief discussion of the biological reverse-engineering problem, which is a central problem in systems biology. As a case study, we then focused on two methods that rely on the solution of the "hitting set problem" but that differ in their approach to solve this problem, thus leading to different performance.

In terms of network inference power, we hypothesize that, for the smaller network, the poor quality of the results when using Jarrah's approach might be ascribed to the type of data used: In [13], it is claimed that the method performs better if perturbation data are added. The algorithm has the ability of considering both wild-type and mutant data to infer the network, and probably results would improve if using such additional data. In the case of Ideker *et al.*'s method, in both networks we think that it is possible that the low quality of results could be due to the lack of ability of using more than one time series at a time, as well as the fact that the implementation of the method does not include self-loops (self-loops are edges connecting a node to itself that may, for example, represent degradation terms in biochemical systems). We believe that this feature is fundamental for a good performance of the algorithm.

When comparing the computational efficiency of the approaches, one should keep in mind that there will always be a difference between exact solutions and approximate solutions based on greedy algorithms or linear programming relaxations. However, since the size of the networks was fairly small, it is possible that the reason Jarrah's method did not find a solution within a reasonable time might lie in encoding issues rather than in the intrinsic computational complexity of the problem.

ACKNOWLEDGMENTS

The authors would like to thank Joe Dundas for the implementation and maintenance of the web tool for the Ideker *et al.* method. We would like to thank as well Dr. Brandilyn Stigler for useful discussions on different aspects of this book chapter. This work was supported in part by grants AFOSR FA9550-08, NIH 1R01GM086881, and NSF grants DMS-0614371, DBI-0543365, IIS-0612044, IIS-0346973, and the DIMACS special focus on Computational and Mathematical Epidemiology.

REFERENCES

1. R. Albert and H. Othmer. The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in Drosophila melanogaster. *J Theor Biol*, 223:1–18, 2003.
2. T. Akutsu, S. Miyano, and S. Kuhara. Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, 16(8):727–734, 2000.
3. M.J. Beal and F. Falciani. A Bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics*, 21(3):349–356, 2005.

4. P. Berman, B. DasGupta, and E. Sontag. Randomized approximation algorithms for set multicover problems with applications to reverse-engineering of protein and gene networks. *Discrete Appl Math*, 155(6–7):733–749, 2007.
5. P. Berman, B. DasGupta, and E. Sontag. Algorithmic issues in reverse-engineering of protein and gene networks via the modular response analysis method. *Ann NY Acad Sci*, 1115:132–141, 2007.
6. D. Camacho, P. Vera-Licona, P. Mendes, and R. Laubenbacher. Comparison of reverse-engineering methods using an in silico network. *Proc NY Acad Sci*, 1115(1):73–89, 2007.
7. W.B. Cannon. The wisdom of the body. Norton, New York, 1993.
8. E. Dimitrova, L. Garcia-Puente, A.S. Jarrah, R. Laubenbacher, B. Stigler, M. Stillman, and P. Vera-Licona. Parameter estimation for Boolean models of biological networks. Submitted for publication.
9. N. Dojer, A. Gambin, A. Mizera, B. Wilczynski, and J. Tiuryn. Applying dynamic Bayesian networks to perturbed gene expression data. *BMC Bioinformatics*, 7(1):249, 2006.
10. N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using bayesian networks to analyze expression data. *J Comput Biol*, 7(3–4):601–620, 2000.
11. A. de la Fuente, N. Bing, I. Hoeschele, and P. Mendes. Discovery of meaningful associations in genomic data using partial correlation coefficients. *Bioinformatics*, 20:3565–3574, 2004.
12. T.S. Gardner, D. di Bernardo, D. Lorez, and J.J. Collins. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301(5629):102–105, 2003.
13. A.S. Jarrah, R. Laubenbacher, B. Stigler, and M. Stillman. Reverse-engineering polynomial dynamical systems. *Adv Appl Math*, 39(4):477–489, 2007.
14. R.M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*. Plenum Press, New York, 1972.
15. T.E. Ideker, V. Thorsson, and R.M. Karp. Discovery of regulatory interactions through perturbation: Inference and experimental design. *Pacific Symposium on Biocomputing*, 2000, pp. 305–316.
16. J. Kim, D. Bates, I. Postlethwaite, P. Heslop-Harrison, and K.H. Cho. Least-squares methods for identifying biochemical regulatory networks from noisy measurements. *BMC Bioinformatics*, 8(1):8, 2007.
17. B. Krupa. On the number of experiments required to find the causal structure of complex systems. *J Theor Biol*, 219(2):257–267, 2002.
18. S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer. COPASI—a COmplex PAthway SImulator. *Bioinformatics*, 22:3067–3074, 2006.
19. R. Laubenbacher and B. Stigler. A computational algebra approach to the reverse-engineering of gene regulatory networks. *J Theor Biol*, 229:523–537, 2004.
20. S. Liang, S. Fuhrman, and R. Somogyi. Reveal, a general reverse-engineering algorithm for inference of genetic network architectures. *Pacific Symposium on Biocomputing*, 1998, pp. 18–29.
21. S. Martin, Z. Zhang, A. Martino, and J.L. Faulon. Boolean dynamics of genetic regulatory networks inferred from microarray time series data. *Bioinformatics*, 23(7): 866–874, 2007.

22. S. Mehra, W.S. Hu, and G. Karypisb. A Boolean algorithm for reconstructing the structure of regulatory networks. *Metabolic Eng*, 6(4):326, 2004.

23. P. Mendes. Biochemistry by numbers: Simulation of biochemical pathways with Gepasi 3. *Trends Biochem Sci*, 22:361–363, 1997.

24. N. Nariai, Y. Tamada, S. Imoto, and S. Miyano. Estimating gene regulatory networks and protein-protein interactions of Saccharomyces cerevisiae from multiple genome-wide data. *Bioinformatics*, 21(suppl 2):ii206–ii212, 2005.

25. G.L. Nemhauser. *Integer and Combinatorial Optimization*. Wiley, New York, 1988.

26. I. Pournara and L. Wernisch. Reconstruction of gene networks using Bayesian learning and manipulation experiments. *Bioinformatics*, 20(17):2934–2942, 2004.

27. J.J. Rice, Y. Tu, and G. Stolovitzky. Reconstructing biological networks using conditional correlation analysis. *Bioinformatics*, 21(6):765–773, 2005.

28. E. Sontag, A. Kiyatkin, and B.N. Kholodenko. Network reconstruction based on steady-state data. *Essays Biochem*, 45:161–176, 2008.

29. G. Tononi, O. Sporns, and G.H. Edelman. Measures of degeneracy and redundancy in biological networks. *PNAS*, 96(6):3257–3262, 1999.

30. L. von Bertalanffy. *General System Theory*. Braziler, New York, 1968.

31. N. Wiener. *Cybernetics or Control and Communication in the Animal and the Machine*. The MIT Press, Cambridge, MA, 1948.

32. J. Yu, V. Smith, P. Wang, A. Hartemink, and E. Jarvis. Advances to Bayesian network inference for generating causal networks from observational biological data. *Bioinformatics*, 20:3594–3603, 2004.

33. M. Zou and S.D. Conzen. A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. *Bioinformatics*, 21(1):71–79, 2005.