BIOINFORMATICS

Vol. 18 no. 2 2002
Pages 261–274

Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks

Ilya Shmulevich[1], Edward R. Dougherty[2], Seungchan Kim[2] and Wei Zhang[1]

[1]Cancer Genomics Laboratory, University of Texas M.D. Anderson Cancer Center, 1515 Holcombe Blvd, Box 85, Houston, TX 77030, USA and [2]Department of Electrical Engineering, Texas A&M University, College Station, TX 77843, USA

Received on May 2, 2001; revised on July 13, 2001; accepted on October 5, 2001

ABSTRACT

Motivation: Our goal is to construct a model for genetic regulatory networks such that the model class: (*i*) incorporates rule-based dependencies between genes; (*ii*) allows the systematic study of global network dynamics; (*iii*) is able to cope with uncertainty, both in the data and the model selection; and (*iv*) permits the quantification of the relative influence and sensitivity of genes in their interactions with other genes.

Results: We introduce Probabilistic Boolean Networks (PBN) that share the appealing rule-based properties of Boolean networks, but are robust in the face of uncertainty. We show how the dynamics of these networks can be studied in the probabilistic context of Markov chains, with standard Boolean networks being special cases. Then, we discuss the relationship between PBNs and Bayesian networks—a family of graphical models that explicitly represent probabilistic relationships between variables. We show how probabilistic dependencies between a gene and its parent genes, constituting the basic building blocks of Bayesian networks, can be obtained from PBNs. Finally, we present methods for quantifying the influence of genes on other genes, within the context of PBNs. Examples illustrating the above concepts are presented throughout the paper.

Contact: is@ieee.org

1 INTRODUCTION

To understand the nature of cellular function, it is necessary to study the behavior of genes in a holistic rather than in an individual manner because the expressions and activities of genes are not isolated or independent of each other. Mathematical and computational methods are being developed in order to construct formal models of genetic interactions. This research direction provides insight and a conceptual framework for an integrative view of genetic function and regulation. It paves the way toward understanding the complex relationship between genes within the genome. Research in this area will provide new hypotheses for experimental verification.

There have been a number of attempts to model gene regulatory networks, including linear models (van Someren *et al.*, 2000; D'Haeseleer *et al.*, 1999), Bayesian networks (Murphy and Mian, 1999; Friedman *et al.*, 2000; Hartemink *et al.*, 2001; Moler *et al.*, 2000), neural networks (Weaver *et al.*, 1999), differential equations (Mestl *et al.*, 1995), and models including stochastic components on the molecular level (Arkin *et al.*, 1998; see Smolen *et al.*, 2000 and Hasty *et al.*, 2001 for reviews of general models). In general, gene expression time trajectories can be modeled as random functions of time (Dougherty *et al.*, 1999). The model system that has received, perhaps, the most attention is the *Boolean Network* model originally introduced by Kauffman (Kauffman, 1969; Glass and Kauffman, 1973; Kauffman, 1993) approximately 30 years ago. In this model, gene expression is quantized to only two levels: ON and OFF. The expression level (state) of each gene is functionally related to the expression states of some other genes, using logical rules. Computational models that reveal these logical interrelations have been successfully constructed (Yuh *et al.*, 1998).

Recent research seems to indicate that many other realistic biological questions may be answered within the seemingly simplistic Boolean formalism, which emphasizes fundamental, generic principles rather than quantitative biochemical details (Huang, 1999). This model system has yielded insights into the overall behavior of large genetic networks (Somogyi and Sniegoski, 1996; Szallasi and Liang, 1998; Wuensche, 1998; Thomas *et al.*, 1995) and allows the study of large data sets in a global fashion. For example, the dynamic behavior of such networks can be used to model many biologically meaningful phenomena—for example, cellular state dynamics, possessing switch-like behavior, stability, and hysteresis (Huang, 1999).

Besides the conceptual framework afforded by such models, a number of practical uses, such as the identification of suitable drug targets in cancer therapy, may be reaped by inferring the structure of the genetic models from experimental data, e.g. from gene expression profiles (Huang, 1999). To that end, much recent work has gone into identifying the structure of gene regulatory networks from expression data (Liang *et al.*, 1998; Akutsu *et al.*, 1998, 1999; D'Haeseleer *et al.*, 2000; Akutsu *et al.*, 2000; Shmulevich *et al.*, 2001). Shmulevich *et al.* (2001) showed that the problem of identifying the network structure using the so-called Best-Fit Extension method is polynomial-time solvable, implying its practical applicability to real data analysis. Nevertheless, it remains an open question as to the degree to which the Boolean formalism can explain the complicated genetic network interplay of higher-order eukaryotes, where more uncertainty of the network exists, attributed to increased gene complexity and differentiation-related specification. The work of Akutsu *et al.* (2000) marked one point of departure from the traditional deterministic constraints of Boolean models by proposing so-called noisy Boolean networks together with an identification algorithm, in order to deal with noise present in expression patterns. In that model, they relax the requirement of consistency intrinsically imposed by the Boolean functions.

Perhaps the most salient limitation of standard Boolean networks is their inherent determinism. From a conceptual point of view, it is likely that the regularity of genetic function and interaction known to exist is not due to 'hard-wired' logical rules, but rather to the intrinsic self-organizing stability of the dynamical system, despite the existence of stochastic components in the cell. From an empirical point of view, the assumption of only one logical rule per gene may lead to incorrect conclusions when inferring these rules from gene expression measurements, as the latter are typically noisy and the number of samples is small relative to the number of parameters to be inferred.

Because of these considerations, we introduce a new model class, called Probabilistic Boolean Networks (PBNs), that shares the appealing properties of Boolean networks, but is able to cope with uncertainty, both in the data and the model selection. There are various reasons for utilizing a probabilistic network. A model incorporates only a partial description of a physical system. This means that a Boolean function giving the next state of a variable is likely to be only partially accurate. There will be occasions when different Boolean functions may actually describe the transition, but these are outside the scope of the conventional Boolean model. If, consequently, we are uncertain as to which transition rule should be used, then a PBN involving a set of possible Boolean functions for each variable may be more suitable than a network in which there is only a single function for each variable.

Even if one is fairly confident that a model is sufficiently robust that other variables can be ignored without significant impact, there remains the problem of inferring the Boolean functions from sample data. In the case of gene-expression microarrays, the data are severely limited relative to the number of variables in the system. Should it happen that a particular Boolean function has even a moderately large number of essential variables, then its design from the data is likely to be imprecise because the number of possible input states will be too large for precise estimation. This situation is exacerbated if some essential variables are either unknown or unobservable. As a consequence of the inability to observe sufficient examples to design the transition rule, it is necessary to restrict the number of variables over which a function is defined. For each subset of the full set of essential variables, there may be an optimal function, in the sense that the prediction error is minimized for that function, given the variables in the subset. These optimal functions must be designed from sample data. Owing to inherent imprecision in the design process, it may be prudent to allow a random selection between several functions, with the weight of selection based on a probabilistic measure of worth, such as the coefficient of determination (Dougherty *et al.*, 2000).

A final consideration is that one may wish to model an open system, rather than a closed system. An open system has inputs (stimuli). Depending upon a particular external condition at a given moment of time, the system may transition differently than it would in the absence of that condition. Such effects have been considered in the framework of using the coefficient of determination in the presence of external stresses (Kim *et al.*, 2000a). Under the assumption that the external stimuli occur asynchronously, it is prudent to allow uncertainty among the transition rules and weight their likelihood accordingly. It may be that the probability of applying a Boolean function corresponding to an unlikely condition is low; however, system behavior might be seriously misunderstood if the possibility of such a transition is ignored.

To set the stage, we briefly review standard Boolean network models and their dynamics in the probabilistic context of Markov chains. Then, we introduce PBNs and expose an important connection with Bayesian Networks—nother model class that has been recently used to model gene expression data. The proposed model class represents an interface between the absolute determinism of Boolean networks and the probabilistic nature of Bayesian networks, in that it incorporates rule-based uncertainty. This compromise is important because rule-based dependencies between genes are biologically meaningful, while mechanisms for handling uncertainty are conceptually and empirically necessary.

Finally, we will discuss the notion of *influence* and *sensitivity* of genes. These measures allow us to quantify and ultimately discover which genes have a higher impact on other genes, individually or collectively, and which genes are more likely to be influenced by other genes. These notions correspond to the strength of downstream and upstream effects in genetic control.

2 BOOLEAN NETWORKS AS MODELS OF GENE REGULATORY NETWORKS

One of the main objectives of Boolean network modeling is to study generic coarse-grained properties of large genetic networks and the logical interactions of genes, without knowing specific quantitative details (Huang, 1999). The biological basis for the development of Boolean networks as models of genetic regulatory networks lies in the fact that during regulation of functional states, the cell exhibits switch-like behavior, which is important for cells to move from one state to another in a normal cell growth process or in situations when cells need to respond to external signals, many of which are detrimental. Let us use cell cycle regulation as an example. Cells grow and divide. This process is highly regulated; failure to do so results in unregulated cell growth in diseases such as cancer. In order for cells to move from the G1 phase to the S phase, when the genetic material, DNA, is replicated for the daughter cells, a series of molecules such as cyclin E and Cyclin Dependent Kinase 2 (cdk2) work together to phosphorylate the Retinoblastoma (Rb) protein and inactivate it, thus releasing cells into the S phase. Cdk2/cyclin E is regulated by two switches: the positive switch complex called Cdk Activating Kinase (CAK) and the negative switch p21/WAF1. The CAK complex can be composed of two gene products: cyclin H and cdk7. When cyclin H and cdk7 are present, the complex can activate cdk2/cyclin E. A negative regulator of cdk2/cyclin E is p21/WAF1, which in turn can be activated by p53. When p21/WAF1 binds to cdk2/cyclin E, the kinase complex is turned off (Gartel and Tyner, 1999). Further, p53 can inhibit cyclin H, a positive regulator of cyclin E/cdk2 (Schneider *et al.*, 1998). This negative regulation is an important defensive system in the cells. For example, when cells are exposed to mutagen, DNA damage occurs. It is to the benefit of cells to repair the damage before DNA replication so that the damaged genetic materials do not pass onto the next generation. An extensive amount of work has demonstrated that DNA damage triggers switches that turn on p53, which then turns on p21/WAF1. p21/WAF1 then inhibits cdk2/cyclin E, thus Rb becomes activated and DNA synthesis stops. As an extra measure, p53 also inhibits cyclin H, thus turning off the switch that turns on cdk2/cyclin E. Such delicate genetic switch networks in the cells is the basis for cellular homeostasis.

For purposes of illustration, let us consider a simplified diagram, shown in Figure 1, illustrating the effects of cdk7/cyclin H, cdk2/cyclin E, and p21/WAF1 on Rb. Thus, p53 and other known regulatory factors are not considered.

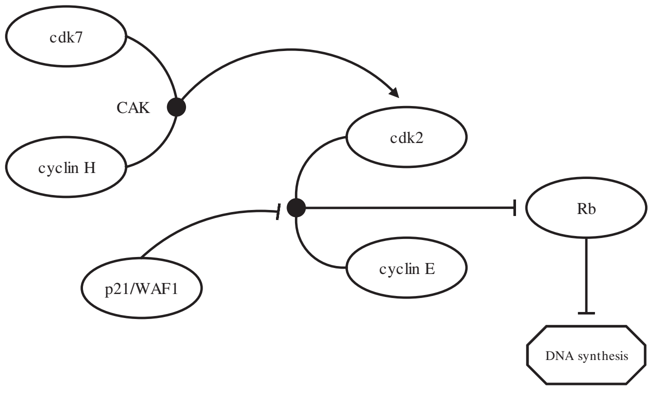

Fig. 1. A diagram illustrating the cell cycle regulation example. Arrowed lines represent activation and lines with bars at the end represent inhibition.

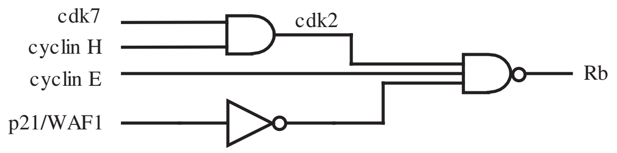

Fig. 2. The logic diagram describing the activity of Rb protein in terms of 4 inputs: cdk7, cyclin H, cyclin E, and p21. The gate with inputs cdk7 and cyclin H is an AND gate, the gate with input p21/WAF1 is a NOT gate, and the gate whose output is Rb is a NAND (negated AND) gate.

While this diagram represents the above relationships from a pathway perspective, we may also wish to represent the activity of Rb in terms of the other variables in a logic-based fashion. Figure 2 contains a logic circuit diagram of the activity of Rb ('on' or 'off') as a Boolean function of four input variables: cdk7, cyclin H, cyclin E, and p21/WAF1. Note that cdk2 is shown to be completely determined by the values of cdk7 and cyclin H using the AND operation and thus, cdk2 is not an independent input variable. Also, in Figure 1, p21/WAF1 is shown to have an inhibitive effect on the cdk2/cyclin E complex, which in turn regulates Rb, while in Figure 2, we see that from a logic-based perspective, the value of p21/WAF1 works together with cdk2 and cyclin E to determine the value of Rb. Such dual representations in biological literature were pointed out by Rzhetsky *et al.* (2000). We now proceed with the more general description of Boolean networks.

For consistency of notation with other related work, we will be using the same notation as in Akutsu *et al.* (1999). A Boolean network $G(V, F)$ is defined by a set of nodes $V = \{x_1, \ldots, x_n\}$ and a list of Boolean functions $F = (f_1, \ldots, f_n)$. A Boolean function $f_i(x_{i_1}, \ldots, x_{i_k})$ with k specified input nodes is assigned to node x_i. In general, k could be varying as a function of i, but without loss of generality, we may define it to be a constant equal

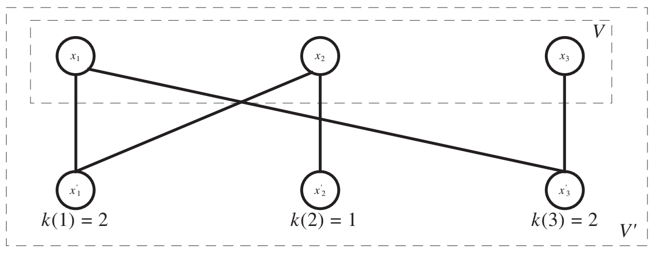

Fig. 3. An example of a wiring diagram for $n = 3$.

to n and allowing the unnecessary variables (nodes) in each function to be *fictitious*. For a function f, the variable x_i is fictitious if

$$f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n),$$

for all $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$. A variable that is not fictitious is called *essential*. Each node x_i represents the state (expression) of gene i, where $x_i = 1$ means that gene i is expressed and $x_i = 0$ means it is not expressed. The list of Boolean functions F represents the rules of regulatory interactions between genes. That is, any given gene transforms its inputs (regulatory factors that bind to it) into an output, which is the state or expression of the gene itself. All genes (nodes) are updated synchronously in accordance with the functions assigned to them and this process is then repeated. The artificial synchrony simplifies computation while preserving the qualitative, generic properties of global network dynamics (Huang, 1999; Kauffman, 1993; Wuensche, 1998).

To capture the dynamic nature of these networks, it is useful to consider a 'wiring diagram' $G'(V', F')$ (Akutsu *et al.*, 1999). Let $k(i)$ be the number of essential variables of function f_i in F. We then construct n additional nodes $x'_1, \ldots, x'_n$ and for each $i = 1, \ldots, n$, we draw an edge from x_{i_j} to x'_i, for each $1 \leqslant j \leqslant k(i)$. Then, $V' = \{x_1, \ldots, x_n, x'_1, \ldots, x'_n\}$ and the list F' is actually the same as F, but containing the same functions being assigned to nodes $x'_1, \ldots, x'_n$ (with inputs from V) while the functions assigned to $x_1, \ldots, x_n$ are just the trivial identity functions, e.g. $f(x_i) = x_i$. In other words, $x'_i = f_i(x_{i_1}, \ldots, x_{i_{k(i)}})$ and thus, the expression pattern $(x_1, \ldots, x_n)$ corresponds to the states of the genes at time t (INPUT) and the pattern $(x'_1, \ldots, x'_n)$ corresponds to the states of the genes at time $t + 1$ (OUTPUT). An example of a wiring diagram for $n = 3$ is shown in Figure 3. Collectively, the states of individual genes in the genome form a *Gene Activity Profile* (GAP; Huang, 1999).

Consider the state space of a Boolean network with n genes. Then, the number of possible GAPs is equal to 2^n. For every GAP, there is another successor GAP into which the system transitions in accordance with its structural rules as defined by the Boolean functions. Thus, there is a directionality that is intrinsic to the dynamics of such systems. Consequently, the system ultimately transitions into so-called *attractor* states. The states of the system that flow into the same attractor state make up a *basin of attraction* of that attractor (Wuensche, 1998). Sometimes, the system periodically cycles between several *limit-cycle* attractors. It is interesting to note that such behavior even exists for some infinite networks (networks with an infinite number of nodes; Moran, 1995), such as those in which every Boolean function is the majority function.

Although the large number of possible GAPs would seem to preclude computer-based analysis, simulations show that for networks in which most Boolean functions have few essential variables, only a small number of GAPs actually correspond to attractors (Kauffman, 1993). Since other GAPs are unstable, the system is normally not found in those states unless perturbed.

2.1 Dynamics of Boolean Networks

Since Boolean networks are completely deterministic, so are their dynamics. The only randomness that may exist lies entirely in the selection of the initial starting state of the network. This can be captured by considering the joint probability distribution of all the genes. In order for us to have a useful probabilistic description of the dynamics of such systems, it is necessary to consider joint probabilities of all Boolean functions corresponding to all the nodes because even after one step of the network, the nodes become dependent, regardless of assumptions on the initial distribution.

To this end, suppose we are given a Boolean network $G(V, F)$ containing n nodes (genes) $x_1, \ldots, x_n$ and an initial joint probability distribution $D(x), x \in \{0, 1\}^n$, over the n-dimensional hypercube. We are interested in computing the joint probability of every node after one step of the network. It is easy to see that

$$\Pr\{f_1(x) = i_1, f_2(x) = i_2, \ldots, f_n(x) = i_n\} = \sum_{x \in \{0,1\}^n : f_k(x) = i_k, k=1,\ldots,n} D(x), \tag{1}$$

where $i_k \in \{0, 1\}$. Equation (1) can then be used in an iterative fashion, following the dynamic nature of the system. In other words, the computed joint distribution $\Pr\{f_k(x) = i_k, k = 1, \ldots, n\}$ can be used in place of $D(x)$ to compute the joint distribution at the next time point. This defines the iterative system

$$D^{t+1} = \Psi(D^t), \tag{2}$$

where the mapping $\Psi : [0, 1]^{2^n} \to [0, 1]^{2^n}$ is implicitly defined by (1). In fact, Ψ is an affine mapping. To see

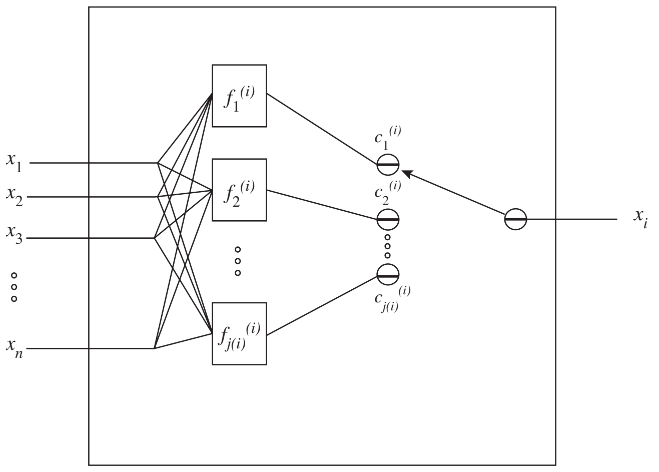

Fig. 4. A basic building block of a PBN.

this, let D^t and D^{t+1} be represented by 1×2^n vectors containing the joint probabilities and let the matrix A be a $2^n \times 2^n$ binary matrix defined as

$$A_{ij} = \left\{ \begin{array}{ll} 1, & \text{if } \exists x \in \{0,1\}^n, C(f_1(x), \ldots, f_n(x)) \\ & \qquad = j, C(x_1, \ldots, x_n) = i \\ 0, & \text{otherwise} \end{array} \right\},$$

where $C(i_1, \ldots, i_n) = 1 + \sum_{j=1}^{n} 2^{n-j} \cdot i_j$, and each $i_j \in \{0,1\}$. Thus, i and j are simply indices such that i is the integer representation of the binary vector $(x_1, \ldots, x_n)$ while j encodes the binary vector $(f_1(x), \ldots, f_n(x))$. This representation is efficient because the matrix A contains exactly one non-zero entry in each row. Thus, (2) can be written as

$$\begin{aligned} D^{t+1} &= D^t \cdot A \\ &= D^0 \cdot A^{t+1}, \end{aligned} \tag{3}$$

where $D^0 = D(x)$ is the starting (prior) joint distribution. Equation (3) is the familiar Markov chain representation, where the state-transition matrix A is binary. This is to be expected, since the state transitions are completely specified by the Boolean functions and the probability of transition can be either 0 or 1.

3 PROBABILISTIC BOOLEAN NETWORKS

As mentioned in the Section **Introduction**, given several ‘good’ competing functions for a given gene, we have little reason to put all our faith in one of them. To overcome the deterministic rigidity of Boolean networks, we extend the Boolean network concept to a probabilistic setting.

The basic idea is to extend the Boolean network to accommodate more than one possible function for each node. Thus, to every node x_i, there corresponds a set

$$F_i = \{f_j^{(i)}\}_{j=1,\ldots,l(i)}, \tag{4}$$

where each $f_j^{(i)}$ is a possible function determining the value of gene x_i and $l(i)$ is the number of possible functions for gene x_i. We will also refer to the functions $f_j^{(i)}$ as *predictors*, since the process of inferring these functions from measurements or equivalently, of producing a minimum-error estimate of the value of a gene at the next time point, is known as prediction in estimation theory. This will be discussed further in Section 3.1. A realization of the PBN at a given instant of time is determined by a vector of Boolean functions. If there are N possible realizations, then there are N vector functions, $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N$ of the form $\mathbf{f}_k = (f_{k_1}^{(1)}, f_{k_2}^{(2)}, \ldots, f_{k_n}^{(n)})$, for $k = 1, 2, \ldots, N$, $1 \leqslant k_i \leqslant l(i)$ and where $f_{k_i}^{(i)} \in F_i$ $(i = 1, \ldots, n)$. In other words, the vector function $\mathbf{f}_k : \{0,1\}^n \to \{0,1\}^n$ acts as a transition function (mapping) representing a possible realization of the entire PBN. Thus, given the values of all genes $(x_1, \ldots, x_n)$, $\mathbf{f}_k(x_1, \ldots, x_n) = (x'_1, \ldots, x'_n)$ gives us the state of the genes after one step of the network given by the realization $\mathbf{f}_k$.

Now, let $\mathbf{f} = (f^{(1)}, \ldots, f^{(n)})$ be a random vector taking values in $F_1 \times \cdots \times F_n$. That is, $\mathbf{f}$ can take on all possible realizations of the PBN. Then, the probability that predictor $f_j^{(i)}$ is used to predict gene i $(1 \leqslant j \leqslant l(i))$ is equal to

$$c_j^{(i)} = \Pr\{f^{(i)} = f_j^{(i)}\} = \sum_{k : f_{k_i}^{(i)} = f_j^{(i)}} \Pr\{\mathbf{f} = \mathbf{f}_k\}. \tag{5}$$

Since the $c_j^{(i)}$ are probabilities, they must satisfy

$$\sum_{j=1}^{l(i)} c_j^{(i)} = 1. \tag{6}$$

It is not necessary that the selection of the Boolean functions composing a specific network be independent. This means that it is not necessarily the case that

$$\begin{aligned} &\Pr\{f^{(i)} = f_j^{(i)}, f^{(l)} = f_k^{(l)}\} \\ &\qquad = \Pr\{f^{(i)} = f_j^{(i)}\} \cdot \Pr\{f^{(l)} = f_k^{(l)}\}. \end{aligned}$$

A PBN is said to be *independent* if the random variables $f^{(1)}, f^{(2)}, \ldots, f^{(n)}$ are independent. In the dependent case, product expansions such as the one given in the preceding equation, as well as ones involving more functions, require conditional probabilities. Henceforth, unless otherwise mentioned, we will assume independent PBNs. Figure 4 illustrates the basic building block of a PBN. A number of predictors share common inputs while their outputs are synthesized, in this case by random selection, into a single output. This type of structure is known as a *synthesis filter bank* in digital signal processing literature. The wiring diagram for the entire PBN would consist of n such building blocks.

Thus, a PBN $G(V, F)$ is defined by a set of nodes $V = \{x_1, \dots, x_n\}$ and the list $F = (F_1, \dots, F_n)$, where the latter is defined in (4). Assuming independence, $N = \prod_{i=1}^{n} l(i)$ is the number of possible PBN realizations. If $l(i) = 1$ for all $i = 1, \dots, n$, then $N = 1$ and the PBN reduces to the standard Boolean network.

The dynamics of the PBN are essentially the same as for Boolean networks, but at any given point in time, the value of each node is determined by one of the possible predictors, chosen according to its corresponding probability. This can be interpreted by saying that at any point in time, we have one out of N possible networks. In order for us to parallel the discussion of Section 2.1 on the dynamics of PBNs, we will first need to calculate the probability that a particular network is selected. Let us define the matrix

$$K = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & 1 & \cdots & 1 & 2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \cdots & 1 & l(n) \\ 1 & 1 & \cdots & 2 & 1 \\ 1 & 1 & \cdots & 2 & 2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \cdots & 2 & l(n) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l(1) & l(2) & \cdots & l(n-1) & l(n) \end{bmatrix},$$

containing lexicographically ordered rows, each one corresponding to a possible network configuration. That is, row m corresponds to network m and the entry j in the ith column specifies that predictor $f_j^{(i)}$ should be used for gene x_i. It is easy to see that K is an $N \times n$ matrix. Using this matrix, the probability that network i is selected is

$$P_i = \Pr\{\text{Network } i \text{ is selected}\} = \prod_{j=1}^{n} c_{K_{ij}}^{(j)}, \qquad (7)$$

where K_{ij} is the ijth entry in matrix K. It can easily be checked that $\sum_{i=1}^{N} P_i = 1$ by noting that

$$\sum_{i=1}^{N} P_i = \sum_{i=1}^{N} \prod_{j=1}^{n} c_{K_{ij}}^{(j)} = \prod_{j=1}^{n} \sum_{i=1}^{l(j)} c_i^{(j)} = \prod_{j=1}^{n} 1 = 1, \qquad (8)$$

where we have used (6).

Now, let us consider the state space of the PBN, also consisting of 2^n states. Similarly to the discussion in Section 2.1, we are interested in establishing the state transition matrix A. In this case, however, the network may transition from a state to a number of other possible states, hence defining a random process. The probability of transitioning from state $(x_1, \dots, x_n)$ to state $(x_1', \dots, x_n')$ can be obtained as

$$\begin{aligned} &\Pr\{(x_1, \dots, x_n) \to (x_1', \dots, x_n')\} \\ &= \sum_{i : f_{K_{i1}}^{(1)}(x_1,\dots,x_n)=x_1', f_{K_{i2}}^{(2)}(x_1,\dots,x_n)=x_2', \dots, f_{K_{in}}^{(n)}(x_1,\dots,x_n)=x_n'} P_i \\ &= \sum_{i=1}^{N} P_i \underbrace{\left[\prod_{j=1}^{n} (1 - |f_{K_{ij}}^{(j)}(x_1, \dots, x_n) - x_j'|) \right]}_{\in\{0,1\}}, \end{aligned} \qquad (9)$$

where in the last expression, binary values are treated as real values. Since $P_i = \Pr\{\text{Network } i \text{ is selected}\}$, (9) can be interpreted as

$$\begin{aligned} \Pr\{(x_1, \dots, x_n) \to (x_1', \dots, x_n')\} = \sum_{i=1}^{N} \Pr\{(x_1, \dots, x_n) \\ \to (x_1', \dots, x_n') \mid \text{Network } i \text{ is selected}\} \cdot P_i \end{aligned}$$

and $\Pr\{(x_1, \dots, x_n) \to (x_1', \dots, x_n') \mid \text{Network } i \text{ is selected}\} \in \{0, 1\}$, as is the case for standard Boolean networks, described in Section 2.1. By using (8) and (9), and the fact that for any $(x_1, \dots, x_n)$ there always exists an $(x_1', \dots, x_n')$ and i such that $\prod_{j=1}^{n} (1 - |f_{K_{ij}}^{(j)}(x_1, \dots, x_n) - x_j'|) = 1$, we have that

$$\sum_{j=1}^{2^n} A_{ij} = 1$$

for any $i = 1, \dots, 2^n$. Thus, A is also a Markov matrix and the PBN is a homogeneous Markov process, meaning that the transition probabilities do not change with time. It also follows that A has at most $N \cdot 2^n$ non-zero entries and reduces to the binary state-transition matrix when $N = 1$, as described in Section 2.1. Let us consider an example PBN illustrating the above constructions.

Example 1. *Suppose we are given a PBN consisting of three genes* $V = (x_1, x_2, x_3)$ *and the function sets* $F = (F_1, F_2, F_3)$, *where* $F_1 = \{f_1^{(1)}, f_2^{(1)}\}$, $F_2 = \{f_1^{(2)}\}$, *and* $F_3 = \{f_1^{(3)}, f_2^{(3)}\}$. *Let the functions be given by the following truth tables.*

$x_1x_2x_3$	$f_1^{(1)}$	$f_2^{(1)}$	$f_1^{(2)}$	$f_1^{(3)}$	$f_2^{(3)}$
000	*0*	*0*	*0*	*0*	*0*
001	*1*	*1*	*1*	*0*	*0*
010	*1*	*1*	*1*	*0*	*0*
011	*1*	*0*	*0*	*1*	*0*
100	*0*	*0*	*1*	*0*	*0*
101	*1*	*1*	*1*	*1*	*0*
110	*1*	*1*	*0*	*1*	*0*
111	*1*	*1*	*1*	*1*	*1*
$c_j^{(i)}$	*0.6*	*0.4*	*1*	*0.5*	*0.5*

Since there are 2 *functions for node* x_1, *1 function for node* x_2, *and* 2 *functions for node* x_3, *there are* $N = 4$ *possible networks and matrix K is equal to*

$$K = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 2 & 1 & 1 \\ 2 & 1 & 2 \end{bmatrix}.$$

For example, the second row of K containing $(1, 1, 2)$ *means that the predictors* $(f_1^{(1)}, f_1^{(2)}, f_2^{(3)})$ *will be used. Finally, by using (9), the state transition matrix A is given by*

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ P_4 & P_3 & 0 & 0 & P_2 & P_1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & P_2 + P_4 & P_1 + P_3 \\ 0 & 0 & 0 & 0 & P_2 + P_4 & P_1 + P_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Let us consider one of the entries in matrix A to clarify its construction. Suppose we wish to compute the transition probability $\Pr\{(1, 1, 0) \to (1, 0, 0)\}$, *which corresponds to the entry* $A_{7,5}$ *(the indexing starts with* 1*). To do this, we need to use the row corresponding to* $(x_1, x_2, x_3) = (1, 1, 0)$ *in the network truth table given above. Then, we look for possible combinations of the predictors for each of the three genes that will give us the values* $(1, 0, 0)$. *By direct inspection, we can see that either* $(f_1^{(1)}, f_1^{(2)}, f_2^{(3)})$ *or* $(f_2^{(1)}, f_1^{(2)}, f_2^{(3)})$ *result in* $(1, 0, 0)$. *The two possible combinations correspond to the second and fourth rows of matrix K. That is why this transition probability is equal to* $P_2 + P_4$. *All other entries in A are computed similarly. The state transition diagram corresponding to this matrix is shown in Figure* 5. *For example, the seventh row of matrix A corresponds to* $(1, 1, 0)$ *and it can be seen that the only possible transitions are to* $(1, 0, 0)$ *or* $(1, 0, 1)$, *corresponding to columns* 5 *and* 6, *respectively.*

3.1 Inference of probabilistic Boolean networks

A natural way to select a set of predictors for a given gene is to employ the *Coefficient Of Determination* (COD; Dougherty *et al.*, 2000; Kim *et al.*, 2000a,b). The COD measures the degree to which the transcriptional levels of an observed gene set can be used to improve the prediction of the transcriptional level of a target gene relative to the best possible prediction in the absence of observations. Let X_i be the target gene; $\mathbf{X}_1^{(i)}, \mathbf{X}_2^{(i)}, \ldots, \mathbf{X}_{l(i)}^{(i)}$ be sets of genes; and $f_1^{(i)}, f_2^{(i)}, \ldots, f_{l(i)}^{(i)}$ be function rules such that $f_1^{(i)}(\mathbf{X}_1^{(i)}), \ldots, f_{l(i)}^{(i)}(\mathbf{X}_{l(i)}^{(i)})$ are optimal predictors of X_i relative to some probabilistic error measure $\varepsilon(X_i, f_k^{(i)}(\mathbf{X}_k^{(i)}))$, keeping in mind that X_i and $f_k^{(i)}(\mathbf{X}_k^{(i)})$ are random variables, for which we use upper case letters.

For each k, the coefficient of determination for X_i relative to the conditioning set $\mathbf{X}_k^{(i)}$ is defined by

$$\theta_k^i = \frac{\varepsilon_i - \varepsilon(X_i, f_k^{(i)}(\mathbf{X}_k^{(i)}))}{\varepsilon_i},$$

where ε_i is the error of the best (constant) estimate of X_i in the absence of any conditional variables (Dougherty *et al.*, 2000). The COD is between 0 and 1 and measures the relative decrease in error from estimating X_i via $f_k^{(i)}(\mathbf{X}_k^{(i)})$ rather than by just the best constant estimate. For instance, in the case of minimum mean-square error estimation, ε_i is the error of the mean of X_i, which is the best constant estimate, and $f_k^{(i)}(\mathbf{X}_k^{(i)})$ is the conditional expectation of X_i given $\mathbf{X}_k^{(i)}$, that is, $f_k^{(i)}(\mathbf{X}_k^{(i)}) = \mathrm{E}[X_i|\mathbf{X}_k^{(i)}]$. In practice, the COD must be estimated from training data with designed approximations being used in place of $f_1^{(i)}, f_2^{(i)}, \ldots, f_{l(i)}^{(i)}$. Consequently, the complexity of the functions $f_1^{(i)}, f_2^{(i)}, \ldots, f_{l(i)}^{(i)}$ and the amount of training data become an issue. For the microarray-based analysis of Kim *et al.* (2000a), the number of genes in each predictor was kept to a maximum of three. For a detailed analysis of design complexity, we refer to Dougherty *et al.* (2000). As mentioned in the Section **Introduction**, the framework afforded by the PBN is well suited for dealing with design imprecision due to limited sample size, where the domain of each predictor may need to be constrained due to lack of training data, but several predictors, with possibly different domains, are collectively employed in a synthesis filter bank fashion.

Let us now assume that a class of gene sets $\mathbf{X}_1^{(i)}, \mathbf{X}_2^{(i)}, \ldots, \mathbf{X}_{l(i)}^{(i)}$ possessing high CODs has been selected. We can take the designed approximations of the optimal function rules $f_1^{(i)}, f_2^{(i)}, \ldots, f_{l(i)}^{(i)}$ as the rule set for gene X_i with the probability of $f_j^{(i)}$ being chosen (see (5)) given by

$$c_k^{(i)} = \frac{\theta_k^i}{\sum_{j=1}^{l(i)} \theta_j^i},$$

where the CODs are the estimates formed from the training data. According to the above expression, those functions corresponding to the highest CODs will be selected more often in the probabilistic network. The number of chosen predictors, $l(i)$, can be a user-selectable parameter and determines the amount of uncertainty that the model can handle.

3.2 Dynamics of probabilistic Boolean networks

In the previous section, we showed that the dynamic behavior of PBNs can be represented as a Markov chain where the state transition matrix A is completely specified by all of the possible Boolean functions and their probabilities. As such, the theory of the limiting behavior of Markov chains is directly applicable to the study of the dynamics of PBNs. For instance, suppose that in Example 1, the probabilities of the predictors are given by $c_1^{(1)} = 0.6$, $c_2^{(1)} = 0.4$, $c_1^{(2)} = 1$, $c_1^{(3)} = 0.5$, $c_2^{(3)} = 0.5$, as shown in the bottom row of the network truth table. Then, the probabilities of the four networks can be computed via (7). For example, to compute P_2, we use row 2 of matrix K and get

$$P_2 = c_1^{(1)} c_1^{(2)} c_2^{(3)} = 0.6 \times 1 \times 0.5 = 0.3.$$

Thus, the probabilities of the four networks are: $P_1 = 0.3$, $P_2 = 0.3$, $P_3 = 0.2$, and $P_4 = 0.2$. Substituting those values into the matrix A and iterating (3), supposing that the starting joint distribution of all the genes is uniform, that is, $D^0 = [\frac{1}{8}, \ldots, \frac{1}{8}]$, we find that the limiting probabilities are

$$\pi = [0.15, 0, 0, 0, 0, 0, 0, 0.85].$$

This tells us that in the long run, all three genes will either be OFF (000), with probability 0.15, or all three genes will be ON (111), with probability 0.85. These two states are called *absorbing*. This can be easily seen by looking at the state transition diagram corresponding to matrix A, shown in Figure 5. Once the process moves into states (000) or (111), it never exits them. This notion corresponds to the concept of attractors in Boolean networks. Similarly, the concept of limit cycle attractors corresponds to the irreducible sets of states in the Markov chain, or in other words, those sets of states such that no state outside them can be reached from any state in them. One can think of the network as being 'trapped' in those sets of states. Finally, the transient states in the Markov chain that lead to either absorbing states or irreducible sets of states correspond to the basins of attraction in Boolean networks. Thus, PBNs qualitatively exhibit the same dynamical properties as Boolean networks, but are inherently probabilistic and thus can cope with uncertainty.

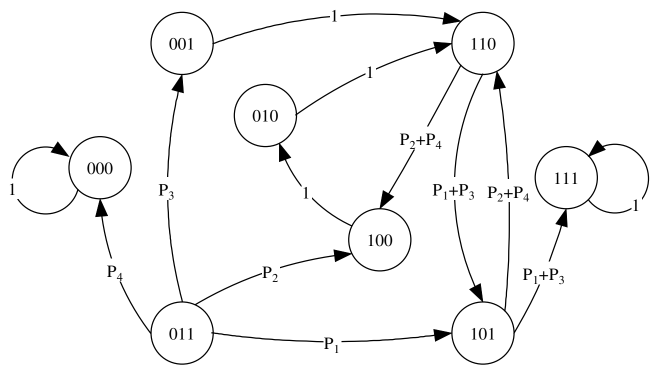

Fig. 5. State transition diagram from Example 1.

An important consideration for PBNs, which is not an issue in Boolean networks, is whether or not there exists a steady-state distribution. For instance, in the above example, the probabilities of ending up in states (000) or (111) would be different if the starting distribution D^0 was also different. To illustrate this, suppose the starting distribution is $D^0 = [1, 0, 0, 0, 0, 0, 0, 0]$, that is, the network begins in state (000) with probability 1. Then, it is clear that it will never escape that state and the limiting probabilities would also be $\pi = D^0$. Therefore, we cannot ask the question about where the PBN will end up in the long run without specifying where it started. Since this is an important issue, let us describe it in more detail.

3.2.1 The existence of steady-state distributions. When considering the long-run behavior of a Markov chain, it is useful to consider equivalence classes within the set of states. This is especially true for genomic systems in which the state space can be extremely large, and it may be partitioned according to various subsystems. If an equivalence class is closed, meaning that no state outside the class is accessible from a state within the class, then for long-run analysis that class can be treated as an irreducible Markov chain in its own right: once inside the class, the system cannot leave it. Hence, we will consider long-run dynamics in terms of a single irreducible finite Markov chain.

A key property in the characterization of long-run behavior is periodicity. Since periodicity is a class property, an irreducible Markov chain can be considered to be or not to be aperiodic. A homogeneous Markov chain with finite state space $S = \{1, 2, \ldots, M\}$ is said to possess a *stationary distribution* (or *invariant distribution*) if there exists a probability distribution $\pi = (\pi_1, \pi_2, \ldots, \pi_M)$ such that, for any $j \in S$ and for any number r of time steps,

$$\pi_j = \sum_{i=1}^{M} \pi_i P_{ij}^r,$$

where P_{ij}^r is the r-step transition probability. Hence, if the initial distribution is $\pi = (\pi_1, \pi_2, \ldots, \pi_M)$, then the probability of being in state i at time r is equal to π_i for all r and the Markov chain is a strictly stationary random process. The Markov chain is said to possess a *steady-state (limiting) distribution* if there exists a probability distribution $\pi = (\pi_1, \pi_2, \ldots, \pi_M)$ such that, for all states $i, j \in S$,

$$\lim_{r\to\infty} P_{ij}^r = \pi_j.$$

If there exists a steady-state distribution, then, regardless of the starting state, the probability of the Markov chain being in state i in the long run is π_i. In particular, for any initial distribution $D^0 = (D_1^0, D_2^0, \ldots, D_M^0)$, the state probability D_i^k approaches π_i as $k \to \infty$. Relative to the probability vector π, the vector D^k satisfies $\lim_{k\to\infty} D^k = \pi$. Every irreducible, finite-state, homogeneous Markov chain possesses a unique probability vector π, with $0 < \pi_i < 1$, providing the stationary distribution. If the chain is also aperiodic, then π also provides the steady-state distribution. Should the chain only be irreducible, and not necessarily aperiodic, then it may not possess a steady-state distribution. A more detailed treatment of the above concepts can be found in most textbooks on stochastic processes, such as Çınlar (1997).

If the chain has a steady-state distribution, we can answer the following question: in the long run, what is the probability that the chain is in state i? The answer does not depend on the initial state. Suppose the states are divided into two classes, C_1 and C_2. Then we can answer the following question without concern for the initial state: in the long run, what is the probability that the chain is in class C_1 (or C_2)? Such a question need not be answerable if there does not exist a steady state (if the chain is not aperiodic).

To illustrate lack of a steady-state distribution, let us consider a three-variable independent PBN. Since we are not concerned with the probabilities, but only the possible Boolean functions, we can use a simplified notation to list the possible functions. We use a table consisting of eight rows corresponding to the eight states, and three columns corresponding to the possible values the Boolean functions can have for the three variables given the state determining the row. The entry * in the table means that the value of the predictor for that gene given the values of the genes in that row can be either 0 or 1. Consider the following function table:

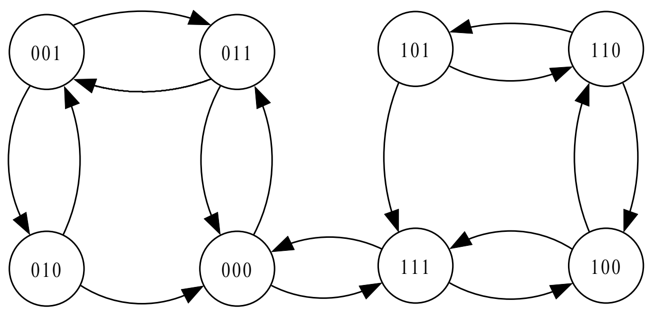

Fig. 6. State transition diagram of an independent PBN without a steady-state distribution.

$x_1x_2x_3$	$f^{(1)}$	$f^{(2)}$	$f^{(3)}$
000	*	1	1
001	0	1	*
010	0	0	*
011	0	0	*
100	1	1	*
101	1	1	*
110	1	0	*
111	*	0	0

So, for example, there are four possible predictors $f_1^{(1)}, f_2^{(1)}, f_3^{(1)}, f_4^{(1)}$ for the first gene. Similarly, there are 256 possible vector functions (network realizations) of the form $\mathbf{f} = (f^{(1)}, f^{(2)}, f^{(3)})$. The corresponding Markov diagram is given in Figure 6. Every state has period 2, and therefore the PBN is not aperiodic. Thus, there does not exist a steady-state distribution. The requirement for a PBN to possess a steady-state distribution may be imposed so that the associated long-run questions may be posed. If so, then this imposes a constraint on the collections of Boolean functions. Certain sets of Boolean functions, such as the one just considered, are not permissible.

The previous example considered an independent PBN. The steady-state requirement can be even more constraining for dependent PBNs. Consider the following very simple PBN with only two network functions (realizations), $\mathbf{f}_1(x_1, x_2) = (f_1^{(1)}, f_1^{(2)}) = (\bar{x}_1, x_2)$ and $\mathbf{f}_2(x_1, x_2) = (f_2^{(1)}, f_2^{(2)}) = (x_1, \bar{x}_2)$. Since the PBN is dependent, the selection of the predictor for the first gene may not be viewed independently of the selection of the predictor for the second gene. The above two possible network realizations imply, for instance, that if $f_1^{(1)} = \bar{x}_1$ is selected for the first gene, then $f_2^{(2)} = \bar{x}_2$ cannot be simultaneously selected for the second gene. That is, the probability that the network $\mathbf{f}$ takes on any realization other than the two given above, say $(f_1^{(1)}, f_2^{(2)})$, is zero.

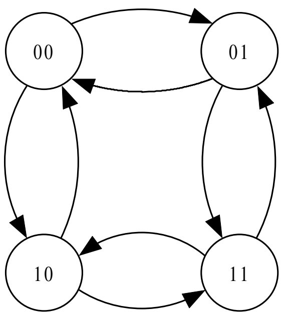

Fig. 7. Example of a dependent PBN not containing a steady-state distribution.

The corresponding Markov diagram is given in Figure 7. This PBN is not aperiodic and does not possess a steady-state distribution. Note that the addition of the network function $\mathbf{f}_3(x_1, x_2) = (f_3^{(1)}, f_3^{(2)}) = (\bar{x}_1, \bar{x}_2)$ makes the PBN aperiodic.

3.3 Relationship to Bayesian networks

Bayesian networks are graphical models that explicitly represent probabilistic relationships between variables (Pearl, 1988; Jensen, 1996; Charniak, 1991). The model structure embeds conditional dependencies and independencies and efficiently specifies the joint probability distribution of all the variables. Recently, Bayesian network models have been used to analyze gene expression data (Murphy and Mian, 1999; Friedman *et al.*, 2000; Hartemink *et al.*, 2001; Moler *et al.*, 2000). For that purpose, the variables in a Bayesian network typically represent expression levels of genes, but can also describe experimental conditions or unobserved variables, such as protein concentrations. The relationships between these variables are represented by a directed graph in which vertices correspond to variables and directed edges between vertices represent their dependencies. One attractive property of Bayesian networks is that they naturally allow one to select a model, from a set of competing models, that best explains the observed expression data.

On the other hand, rule-based models are appealing to biologists, since the types of dependencies between genes constitute important biological information. To partially deal with this problem, Hartemink *et al.* (2001) suggested annotating the edges in Bayesian networks, thus describing simple positive/negative relationships between a variable and its parent in the graph, also in the context of binary variables. A serious drawback of applying Bayesian networks to gene expression data is the computational complexity of learning the network structure. In many formulations, this is an NP-hard problem (Chickering, 1996). The problem is further confounded due to the existence of equivalent networks, in the sense that they represent equivalent independence statements. Finally, standard Bayesian networks are inherently static, as they can have no directed cycles, and in order to model dynamic processes, the static model must be ‘unrolled’ in time, resulting in a so-called Dynamic Bayesian Network (Murphy and Mian, 1999), further complicating the process of learning the model structure and parameters.

Conversely, Boolean models encode rules of genetic regulation, are inherently dynamic, and lend themselves to tractable inference (Shmulevich *et al.*, 2001; Liang *et al.*, 1998; Akutsu *et al.*, 1998, 1999). The proposed PBN models retain these appealing properties, while furnishing the means to handle uncertainty. Let us take a closer look at the basic building blocks of PBNs and Bayesian networks, illustrating their relationships.

In a Bayesian network, the conditional probability of a random variable X_i given all its predecessors in the graph is equal to the conditional probability of X_i given only the *Markovian parents* of X_i, denoted by $\mathrm{Pa}(X_i)$. In other words, $\Pr\{x_i|x_1, \ldots, x_{i-1}\} = \Pr\{x_i|\mathrm{Pa}(x_i)\}$, where lowercase letters denote realizations of the respective random variables. Using the chain rule of probability, we can express the joint probability as a product of conditional probabilities as

$$\Pr\{x_1, \ldots, x_n\} = \prod_i \Pr\{x_i|\mathrm{Pa}(x_i)\}.$$

This fact facilitates economical representation of joint distributions in Bayesian networks, since the entire joint distribution table does not need to be stored. Let us consider these conditional probability building blocks of Bayesian networks in the context of PBNs.

Recall that for a node (gene) x_i, there corresponds a set F_i of possible predictor functions along with their probabilities $c_1^{(i)}, \ldots, c_{l(i)}^{(i)}$. Now, we are interested in computing the conditional probability of gene X_i given its ‘parents’, namely, the set of all genes involved in predicting it. To make this more formal, let the set $\mathbf{X}_j^{(i)} \subseteq \{x_1, \ldots, x_n\}$, $j = 1, \ldots, l(i)$, denote the set of essential variables used by predictor $f_j^{(i)}$ for gene x_i. Then, the set

$$\mathrm{Pa}(x_i) = \bigcup_{j=1}^{l(i)} \mathbf{X}_j^{(i)}$$

is the set of all variables used to predict gene x_i, or simply, the *parents* of x_i. For convenience of notation, let us expand the domains of all predictors by adding fictitious variables so that all are functions of the variables in $\mathrm{Pa}(x_i)$. Computationally, this step is not necessary. Then, the conditional probability that $X_i = 1$, given the fact that predictor $f_j^{(i)}$ is used, is equal to

$$\Pr\{X_i = 1 | f_j^{(i)} \text{is used}\} = \sum_{x \in \{0,1\}^{|\mathrm{Pa}(x_i)|}} D_i(x) f_j^{(i)}(x), \tag{10}$$

where $D_i(x)$ is the joint distribution over the variables in $\mathrm{Pa}(x_i)$. Because we are in the binary setting, (10) can also be interpreted as the conditional expectation of X_i given the predictor $f_j^{(i)}$. Computationally, we only need to consider the essential variables of $f_j^{(i)}$ and their joint distribution, which can be obtained from the entire joint distribution $D(x)$ by 'integrating out' the fictitious variables. Moreover, (10) can be computed as a vector multiplication of the joint distribution and the truth table of the predictor.

Ultimately, we are interested in obtaining $\Pr\{X_i = 1\}$. Recall that $c_j^{(i)} = \Pr\{f_j^{(i)}$ is used$\}$. Then,

$$\begin{aligned}\Pr\{X_i = 1\} &= \sum_{j=1}^{l(i)} \Pr\{X_i = 1 | f_j^{(i)} \text{is used}\} \cdot c_j^{(i)} \\ &= \sum_{j=1}^{l(i)} c_j^{(i)} \cdot \sum_{x \in \{0,1\}^{|\mathrm{Pa}(x_i)|}} D_i(x) f_j^{(i)}(x). \end{aligned} \tag{11}$$

Equation (11) provides the expression of the probability of the target gene X_i in terms of the probabilities of the predictors and the joint distribution of the target's parent genes.

4 INFLUENCES OF GENES IN PROBABILISTIC BOOLEAN NETWORKS

Given a gene x_i and a predictor $f_j^{(i)}$ for that gene, along with the genes used to make the prediction, it is important to distinguish those genes that have a major impact on the predictor from those that have only a minor impact. In other words, some (parent) genes are more 'important' than others in determining the value of a target gene. Many examples of such biased regulation of gene expression are known to biologists. For example, the cell cycle regulator gene p21/WAF1/cip1 can be transcriptionally activated by a series of genes p53, smad4, AP2, BRCA1, etc. (Gartel and Tyner, 1999). Among those genes, p53 has the most potent effect.

Let us illustrate this with a simple example. Suppose the Boolean function $f(x_1, x_2, x_3, x_4) = x_1 \vee x_2x_3x_4$ is used as a predictor of some gene, where the symbol $\vee$ is disjunction and $\cdot$ is conjunction. It is easy to see that x_1 is a more 'important' variable because setting it to 1 forces the function to be equal to 1 regardless of the other variables. However, if $x_1 = 0$, the other three variables must 'cooperate' in order to determine the value of the function. The influences of variables on Boolean functions can quantify this notion of importance, while taking into account the joint distribution of the variables.

To define the notion of influence, first consider the partial derivative of a Boolean function with respect to variable x_j $(1 \leqslant j \leqslant n)$:

$$\frac{\partial f(x)}{\partial x_j} = |f(x) - f(x^{(j)})|,$$

where $x^{(j)}$ is the same as x except that the jth component is toggled (from 0 to 1, or from 1 to 0). While this definition uses the real-valued equivalent of a Boolean function and resembles the definition of a real-valued derivative, another standard definition of the partial derivative of a Boolean function f is given by

$$\frac{\partial f(x)}{\partial x_j} = f(x^{(j,0)}) \oplus f(x^{(j,1)}),$$

where $\oplus$ is addition modulo 2 (exclusive OR) and

$$x^{(j,k)} = (x_1, \ldots, x_{j-1}, k, x_{j+1}, \ldots, x_n),$$

for $k = 0, 1$. The second definition inherently indicates that the partial derivative is itself a Boolean function. The physical meaning behind the partial derivative of a Boolean function with respect to the ith variable is that, defined on the $n-1$ dimensional projection of the n-cube, it acts as an indicator of whether or not the function differs along the ith dimension. The partial derivative is 0 if toggling the value of variable x_j does not change the value of the function, and it is 1 otherwise.

The *influence* of the variable x_j on the function f is the expectation of the partial derivative with respect to the distribution $D(x)$:

$$\begin{aligned} I_j(f) &= E_D\left[\frac{\partial f(x)}{\partial x_j}\right] \\ &= \Pr\left\{\frac{\partial f(x)}{\partial x_j} = 1\right\} = \Pr\{f(x) \neq f(x^{(j)})\}. \end{aligned} \tag{12}$$

The last expression gives the influence as the probability that a toggle of the jth variable changes the value of the function (Kahn *et al.*, 1998).

Recall that in (standard) Boolean networks, a predictor function f_i is assigned to node x_i. Thus, instead of thinking of a variable as influencing the function, we will think of it as influencing the target node or gene. Furthermore, the partial derivative itself is a Boolean function and thus (12) can be interpreted as a conditional probability that the value of node x_i is equal to 1, given that the partial derivative was used as a predictor of the value of node x_i:

$$I_j(x_i) = \Pr\left\{X_i = 1 \,\middle|\, \frac{\partial f_i(x)}{\partial x_j} \text{ is used as a predictor}\right\}, \tag{13}$$

where the notation $I_j(x_i)$ represents the influence of gene x_j on gene x_i, given f_i as the predictor (cf. 10). The point of this observation is to show that the same methods and framework that are used for gene predictors in Boolean networks can be used for influences as well, simply by replacing predictors by their partial derivatives.

Let us go one step further and consider PBNs. In this case, we have a number of predictors for each gene, along with their probabilities. As before, let F_i be the set of predictors for gene x_i with corresponding probabilities $c_1^{(i)}, \ldots, c_{l(i)}^{(i)}$. Let $I_k(f_j^{(i)})$ be the influence of variable x_k on the predictor $f_j^{(i)}$. Since many possible predictors can be used for gene x_i, we would like to determine the overall influence of gene x_k on gene x_i. Thus, the same idea as in (11) can be used, simply by unconditioning on all the partial derivatives of the predictors. Specifically,

$$I_k(x_i) = \sum_{j=1}^{l(i)} I_k(f_j^{(i)}) \cdot c_j^{(i)}.$$

This calculation can be performed between all pairs of variables and a $n \times n$ matrix Γ of influences can be constructed. That is, $\Gamma_{ij} = I_i(x_j)$. We will call Γ the *influence matrix*. Let us consider an example illustrating the computation of the influence of one variable on another, in the context of PBNs.

EXAMPLE 2. *Consider the same PBN as in Example* 1. *Again, let* $c_1^{(1)} = 0.6$, $c_2^{(1)} = 0.4$, $c_1^{(2)} = 1$, $c_1^{(3)} = 0.5$, $c_2^{(3)} = 0.5$. *Suppose we would like to compute the influence of variable* x_2 *on variable* x_1. *Therefore, we will need to use both of the predictors* $f_1^{(1)}$ *and* $f_2^{(1)}$ *given in the table of Example* 1. *Further, suppose D is the uniform distribution, that is,* $D(x) = 1/8$ *for all* $x \in \{0,1\}^3$. *First, we get*

$$I_2(f_1^{(1)}) = E_D\left[\frac{\partial f_1^{(1)}(x)}{\partial x_2}\right] = 0.5$$

$$I_2(f_2^{(1)}) = E_D\left[\frac{\partial f_2^{(1)}(x)}{\partial x_2}\right] = 0.75.$$

Putting these two influences together, we obtain

$$I_2(x_1) = 0.5 \cdot 0.6 + 0.75 \cdot 0.4 = 0.6.$$

If we repeat these calculations between all pairs of variables, we will obtain the influence matrix

$$\Gamma = \begin{bmatrix} 0.1 & 0.75 & 0.375 \\ 0.6 & 0.75 & 0.375 \\ 0.6 & 0.75 & 0.375 \end{bmatrix}.$$

It is not surprising that Γ is not symmetric, since the influence of variable x_i on variable x_j may be stronger than the influence of variable x_j on variable x_i or vice versa.

Another useful measure is the *average sensitivity* of a function. For example, consider the *sensitivity* of f at vector x:

$$s_x(f) = \sum_{j=1}^{n} |f(x) - f(x^{(j)})|.$$

The average sensitivity of f (with respect to distribution D) is then

$$\begin{aligned} s(f) = E_D[s_x(f)] &= \sum_{j=1}^{n} E_D[|f(x) - f(x^{(j)})|] \\ &= \sum_{j=1}^{n} I_j(f). \end{aligned}$$

Thus, one definition of average sensitivity of function f is the sum of the influences of all variables on f.

An interpretation of $s(f)$ is how much, on the average, the function f changes between Hamming neighbors (i.e. those vectors differing in one coordinate). Since for PBNs, we have several predictors for each gene, we will again use the notion of influence of a gene on another gene. In other words, the average sensitivity of gene x_i can be expressed as $s(x_i) = \sum_{j=1}^{n} I_j(x_i)$. Given the influence matrix, this can be computed as

$$s(x_i) = \sum_{k=1}^{n} \Gamma_{ki}.$$

Biologically, the sensitivity of a gene represents the stability or, in some sense, the autonomy of a gene. If the sensitivity of a gene is low, this implies that other genes have little affect on it. The so-called 'house-keeping' genes that encode structural protein in the cells fall into this category. In Example 2, gene x_2 is the most sensitive.

Since we're dealing with PBNs, we always have at least as many predictors as we do genes. Hence, another informative measure would be the collective effect of a gene on all the other genes. This could simply be called the *influence* of gene x_i, denoted by $r(x_i)$, and can also be obtained from the influence matrix as

$$r(x_i) = \sum_{k=1}^{n} \Gamma_{ik}.$$

Biologically, a gene with a high influence factor has a high collective impact on the other genes. It is precisely these genes that have the potential to regulate the dynamics of the network, as their perturbation can lead to significant 'downstream' effects, possibly forcing the system to transition to a different basin of attraction. Many transcriptional factor genes fall into this category. In Example 2, genes x_2 and x_3 are equally more important than gene x_1.

5 DISCUSSION

We have introduced a new class of models for genetic regulatory networks. This new class constitutes a probabilistic generalization of the well-known Boolean network models and offers a more flexible and powerful modeling framework. As is typically the case, the added flexibility brings added design complexity; however, the precise manner in which this complexity is manifested is itself flexible and in the hands of the designer. The maximum number of Boolean functions to predict a target gene may be increased or decreased depending on the amount of training data available. For instance, rather than a single Boolean function of four variables, which would make the model deterministic and limit the regulatory modeling of the target gene to four variables, one could use a PBN with two Boolean functions of three variables each. This would make the model non-deterministic and allow the effects of up to six variables on the target gene. The greater flexibility of the PBN, and therefore its ability to model richer systems by fitting richer data sets, can be balanced with increased design complexity relative to both the number of functions per gene and the number of variables upon which those functions operate. The PBN can be constructed so as to involve many simple, but decent predictors rather than one complex, but poor one. This idea is in accordance with the reasoning behind splines and multiresolution analysis: rather than fitting one overly complex model to the data, one can fit many simple models and use them in a concerted manner.

Our method of inference, based on the coefficient of determination as described in Section 3.1, produces a number of good candidate predictors for each target gene. Since the COD itself is estimated from the data, we have little reason to put all our faith into just one possibly good predictor. Thus, our approach has been to probabilistically 'synthesize' the good predictors such that each predictor's contribution is proportional to its determinative potential. As previously remarked, for small sample sizes, the complexity of each predictor can be limited. As new data make themselves available, the model class naturally allows us to narrow down as needed, effectively reducing the uncertainty for predicting each target gene.

We plan to apply these inference algorithms to a set of gene expression data. Because of the limited number of samples and the relative simplicity of the potential predictor functions, we prefer to perform a full search for each optimal predictor, rather than rely on heuristic suboptimal solutions the error of which cannot be reliably estimated. Consequently, the combinatorial nature of the search space and the relative simplicity of individual predictors is naturally well suited for distributed computing. For example, for any three genes, there are only 256 Boolean functions to search through to obtain the optimal estimate while there are many three-gene combinations. The implementation of this prediction algorithm on a massively parallel supercomputer is described in Suh *et al.* (2001).

In addition to the advantages mentioned above, the framework of PBNs retains the appealing properties of Boolean networks, such as rule-based dependencies between genes as well as the amenability to global analysis of dynamics. The rich and mature theory of Markov chains provides many useful tools for the latter. For example, future work should concentrate on the rule-based structure of PBNs characterizing the existence of steady-state distributions. Another advantage of PBNs is that they naturally allow one to incorporate prior biological knowledge, if necessary. Thus, if certain regulatory relationships are known to exist, the class of the functions for the genes in question can be constrained such that they reflect this prior knowledge. For instance, if it is known that for a certain collection of genes, the activation of genes cannot cause inhibitory effects on the target gene, then we can restrict our attention to the class of monotone Boolean functions, effectively reducing the search space. Or, for example, if it is believed that for certain classes of genes, canalizing functions provide the proper class of rules, then this constraint can be easily introduced into the inference algorithm.

Finally, the ability of the PBNs to allow quantification of the influence of genes on other genes is important for discovering potential drug targets. Intervention of genes that have greater downstream impact are more likely to result in global network changes. This should be studied from the point of view of optimal control, where the goal is to determine which inputs are most likely to drive the system to a desired state.

ACKNOWLEDGEMENTS

The authors are grateful to the Referees for their careful reading of the manuscript and many helpful suggestions. This work was partially supported by the Tobacco Settlement Funds as appropriated by the Texas State Legislature, by a generous donation from the Michael and Betty Kadoorie Foundation, and by a grant from the RGK Foundation.

REFERENCES

Akutsu,T., Kuhara,S., Maruyama,O. and Miyano,S. (1998) Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'98)*. pp. 695–702.

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.*, **4**, 17–28.

Akutsu,T., Miyano,S. and Kuhara,S. (2000) Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, **16**, 727–734.

Arkin,A., Ross,J. and McAdams,H.H. (1998) Stochastic kinetic analysis of developmental pathway bifurcation in phage-infected *Escherichia coli* cells. *Genetics*, **149**, 1633–1648.

Charniak,E. (1991) Bayesian networks without tears. *AI Mag.*, **12**, 50–63.

Chickering,D. (1996) Learning Bayesian networks is NP-complete. In Fisher,D. and Lenz,H. (eds), *Learning from Data*. Springer, Berlin, pp. 121–130.

Çınlar,E. (1997) *Introduction to Stochastic Processes*. Prentice Hall, Eglewood Cliffs, NJ.

D'Haeseleer,P., Wen,X., Fuhrman,S. and Somogyi,R. (1999) Linear modeling of mRNA expression levels during CNS development and injury. *Pac. Symp. Biocomput.*, **4**, 41–52.

D'Haeseleer,P., Liang,S. and Somogyi,R. (2000) Genetic network inference: from co-expression clustering to reverse engineering. *Bioinformatics*, **16**, 707–726.

Dougherty,E.R., Bittner,M.L., Chen,Y., Kim,S., Sivakumar,K., Barrera,J., Meltzer,P. and Trent,J.M. (1999) Nonlinear filters in genomic control. In *Proceedings of IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing*. Antalya, Turkey.

Dougherty,E.R., Kim,S. and Chen,Y. (2000) Coefficient of determination in nonlinear signal processing. *Signal Process.*, **80**, 2219–2235.

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

Gartel,A.L. and Tyner,A.L. (1999) Transcriptional regulation of the p21(WAF1/CIP1) gene. *Exp. Cell Res.*, **246**, 280–289.

Glass,K. and Kauffman,S.A. (1973) The logical analysis of continuous, non-linear biochemical control netoworks. *J. Theor. Biol.*, **39**, 103–129.

Hartemink,A.J., Gifford,D.K., Jaakkola,T.S. and Young,R.A. (2001) Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In *Pacific Symposium on Biocomputing*. Hawaii.

Hasty,J., McMillen,D., Isaacs,F. and Collins,J.J. (2001) Computational studies of gene regulatory networks: *in numero* molecular biology. *Nature Rev. Genet.*, **2**, 268–279.

Huang,S. (1999) Gene expression profiling, genetic networks, and cellular states: an integrating concept for tumorigenesis and drug discovery. *J. Mol. Med.*, **77**, 469–480.

Jensen,F. (1996) *Introduction to Bayesian Networks*. Springer, Berlin.

Kahn,J., Kalai,G. and Linial,N. (1988) The influence of variables on Boolean functions. In *29th Annual Symposium on Foundations of Computer Science*. IEEE Washington, DC, USA, pp. 68–80.

Kauffman,S.A. (1969) Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.*, **22**, 437–467.

Kauffman,S.A. (1993) *The Origins of Order: Self-organization and Selection in Evolution*. Oxford University Press, New York.

Kim,S., Dougherty,E.R., Chen,Y., Sivakumar,K., Meltzer,P., Trent,J.M. and Bittner,M. (2000a) Multivariate measurement of gene expression relationships. *Genomics*, **67**, 201–209.

Kim,S., Dougherty,E.R., Bittner,M.L., Chen,Y., Sivakumar,K., Meltzer,P. and Trent,J.M. (2000b) General nonlinear framework for the analysis of gene interaction via multivariate expression arrays. *J. Biomed. Opt.*, **5**, 411–424.

Liang,S., Fuhrman,S. and Somogyi,R. (1998) REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. *Pac. Symp. Biocomput.*, **3**, 18–29.

Mestl,T., Plahte,E. and Omholt,S.W. (1995) A mathematical framework for describing and analysing gene regulatory networks. *J. Theor. Biol.*, **176**, 291–300.

Moler,E.J., Radisky,D.C. and Mian,I.S. (2000) Integrating naive Bayes models and external knowledge to examine copper and iron homeostasis in *S. cerevisiae*. *Physiol. Genom.*, **4**, 127–135.

Moran,G. (1995) On the period-two-property of the majority operator in infinite graphs. *Trans. Am. Math. Soc.*, **347**, 1649–1667.

Murphy,K. and Mian,S. (1999) Modelling gene expression data using dynamic Bayesian networks. *Technical Report*. Berkeley.

Pearl,J. (1988) *Probabilistic Reasoning in Intelligent Systems*. Morgan Kauffman, San Francisco.

Rzhetsky,A., Koike,T., Kalachikov,S., Gomez,S.M., Krauthammer,M., Kaplan,S.H., Kra,P., Russo,J.J. and Friedman,C. (2000) A knowledge model for analysis and simulation of regulatory networks. *Bioinformatics*, **16**, 1120–1128.

Schneider,E., Montenarh,M. and Wagner,P. (1998) Regulation of CAK kinase activity by p53. *Oncogene*, **17**, 2733–2741.

Shmulevich,I., Yli-Harja,O. and Astola,J. (2001) Inference of genetic regulatory networks under the best-fit extension paradigm. In *Proceedings of the IEEE—EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP-01)*, June 3–6, Maryland, Baltimore.

Smolen,P., Baxter,D. and Byrne,J. (2000) Mathematical modeling of gene networks. *Neuron*, **26**, 567–580.

Somogyi,R. and Sniegoski,C. (1996) Modeling the complexity of gene networks: understanding multigenic and pleiotropic regulation. *Complexity*, **1**, 45–63.

Suh,E.B., Dougherty,E.R., Kim,S., Russ,D.E. and Martino,R.R. (2001) Parallel computing methods for analyzing gene expression relationships. In *Proceedings of the SPIE Microarrays: Optical Technologies and Informatics*. San Jose, CA.

Szallasi,Z. and Liang,S. (1998) Modeling the normal and neoplastic cell cycle with realistic Boolean genetic networks: their application for understanding carcinogenesis and assessing therapeutic strategies. *Pac. Symp. Biocomput.*, **3**, 66–76.

Thomas,R., Thieffry,D. and Kaufman,M. (1995) Dynamical behaviour of biological regulatory networks—I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state. *Bull. Math. Biol.*, **57**, 247–276.

van Someren,E.P., Wessels,L.F.A. and Reinders,M.J.T. (2000) Linear modeling of genetic networks from experimental data. In *Intelligent Systems for Molecular Biology (ISMB 2000)*, August 19–23, San Diego, CA.

Weaver,D.C., Workman,C.T. and Stormo,G.D. (1999) Modeling regulatory networks with weight matrices. *Pac. Symp. Biocomput.*, **4**, 112–123.

Wuensche,A. (1998) Genomic regulation modeled as a network with basins of attraction. *Pac. Symp. Biocomput.*, **3**, 89–102.

Yuh,C.-H., Bolouri,H. and Davidson,E.H. (1998) Genomic *cis*-regulatory logic: experimental and computational analysis of a sea urchin gene. *Science*, **279**, 1896–1902.