

Systems biology

Intervention in a family of Boolean networks

Ashish Choudhary¹, Aniruddha Datta¹, Michael L. Bittner² and Edward R. Dougherty^{1,2,*}¹Department of Electrical Engineering, Texas A&M University, College Station, TX 77843, USA and²Translational Genomics Research Institute, 400 North Fifth Street, Suite 1600, Phoenix, AZ 85004, USA

Received on September 7, 2005; revised on October 22, 2005; accepted on November 3, 2005

Advance Access publication November 12, 2005

Associate Editor: Steen Knudsen

ABSTRACT

Motivation: Intervention in a gene regulatory network is used to avoid undesirable states, such as those associated with a disease. Several types of intervention have been studied in the framework of a probabilistic Boolean network (PBN), which is a collection of Boolean networks in which the gene state vector transitions according to the rules of one of the constituent networks and where network choice is governed by a selection distribution. The theory of automatic control has been applied to find optimal strategies for manipulating external control variables that affect the transition probabilities to desirably affect dynamic evolution over a finite time horizon. In this paper we treat a case in which we lack the governing probability structure for Boolean network selection, so we simply have a family of Boolean networks, but where these networks possess a common attractor structure. This corresponds to the situation in which network construction is treated as an ill-posed inverse problem in which there are many Boolean networks created from the data under the constraint that they all possess attractor structures matching the data states, which are assumed to arise from sampling the steady state of the real biological network.

Results: Given a family of Boolean networks possessing a common attractor structure composed of singleton attractors, a control algorithm is derived by minimizing a composite finite-horizon cost function that is a weighted average over all the individual networks, the idea being that we desire a control policy that on average suits the networks because these are viewed as equivalent relative to the data. The weighting for each network at any time point is taken to be proportional to the instantaneous estimated probability of that network being the underlying network governing the state transition. The results are applied to a family of Boolean networks derived from gene-expression data collected in a study of metastatic melanoma, the intent being to devise a control strategy that reduces the WNT5A gene's action in affecting biological regulation.

Availability: The software is available on request.

Contact: edward@ee.tamu.edu

Supplementary information: The supplementary Information is available at <http://ee.tamu.edu/~edward/tree>

INTRODUCTION

Numerous gene regulatory models have been proposed. For the most part these have been developed for descriptive purposes, by which we mean that their purpose is to characterize gene interaction. From a translational perspective, a salient objective is to base

diagnosis and treatment for disease upon these models. For treatment, this constitutes the derivation of intervention strategies that affect the network in beneficial ways. To date, the largest effort in this direction has been in the context of probabilistic Boolean networks (PBNs). A PBN is essentially a collection of Boolean networks in which at any discrete time point the gene state vector transitions according to the rules of one of the Boolean networks (Shmulevich *et al.*, 2002a). The earliest effort at intervention involved resetting the state of a PBN to a more desirable initial state and letting the network evolve from there (Shmulevich *et al.*, 2002b). More sophisticated approaches have since been proposed that use the methods of automatic control (Datta *et al.*, 2003, 2004; Pal *et al.*, 2005a). These employ external control variables that can be manipulated in progression to affect the dynamic evolution of a network over a finite time horizon.

Given a dataset consisting of gene-expression measurements, PBN design constitutes an ill-posed inverse problem that is treated using a design algorithm to generate a solution. Inference can be formalized by postulating criteria that constitute a solution space for the inverse problem. The criteria come in two forms: (1) the *constraint criteria* are composed of restrictions on the form of the network and (2) the *operational criteria* are composed of relations that must be satisfied between the model and the data. The solution space consists of all PBNs that satisfy the two sets of criteria. Recognizing that PBNs are composed of Boolean networks, and since it is difficult to infer the probabilistic structure among the constituent Boolean networks from the steady-state data typically used for design, a more general view may be taken in which the inverse problem is restricted to determining a solution space of Boolean networks and then finding networks in that space (Pal *et al.*, 2005b). Without a probabilistic structure between the Boolean networks, we have a family of Boolean networks satisfying both the constraint and operational criteria. If desired, one can then go further and construct a PBN using networks from the family, or one can simply treat the family as a collection of solutions to the Boolean-network inverse problem.

In this paper, we derive a control algorithm that can be applied to a family of Boolean networks. This is accomplished by minimizing a composite cost function that is a weighted average cost over the entire family. Ideally, the weighting for each member of the family at any time point would be proportional to the instantaneous probability of a particular network being the governing network. Although these instantaneous probabilities are not known, we adaptively estimate them from the available data and the estimate is used to implement the control algorithm.

*To whom correspondence should be addressed.

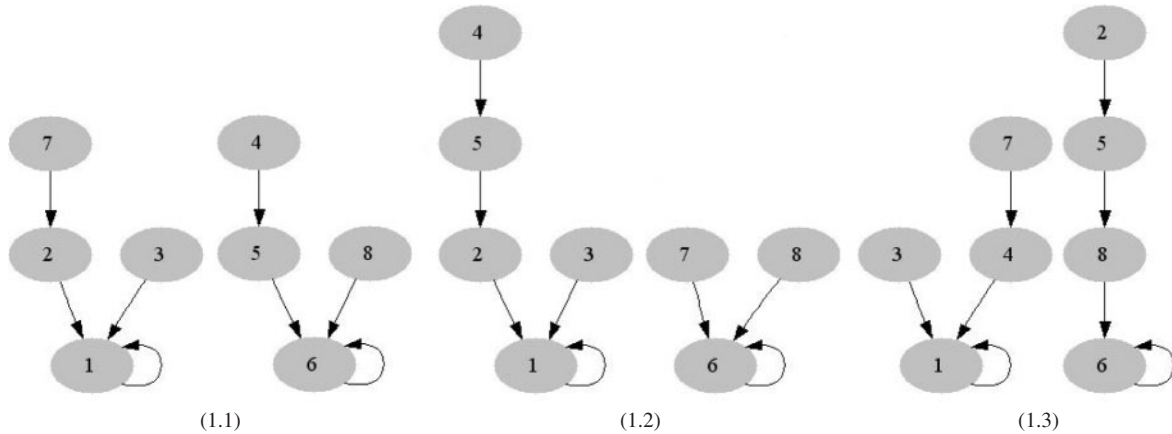


Fig. 1. Distinct autonomous networks N_1 (shown as 1.1), N_2 (shown as 1.2) and N_3 (shown as 1.3) with predictors $\mathcal{P}_1 = \{x_1, x_2, x_3\}$, $\mathcal{P}_2 = \{x_1, x_3\}$, $\mathcal{P}_3 = \{x_1\}$, and attractors 1(000) and 6(101).

SYSTEMS AND METHODS

Boolean networks

A Boolean network (BN) consists of a set of nodes (genes) in which each gene can take on one of two binary values, 0 or 1 (Kauffman, 1969, 1993). Given n genes, the activity level of gene i at time step k is denoted by $x_i(k)$, where $x_i(k) = 0$ indicates that gene i is not expressed and $x_i(k) = 1$ indicates that it is expressed. The overall expression levels of all the genes in the network at time step k is given by the state (row) vector $x(k) = [x_1(k), x_2(k), \dots, x_n(k)]$, also called the gene activity profile (GAP) of the network at time k . Gene i evolves from time k to $k + 1$ according to the Boolean function $f_i(x_1(k), x_2(k), \dots, x_n(k))$. Usually the value of f_i does not depend on the entire set $\{x_1, x_2, \dots, x_n\}$ of n gene values but only on a finite subset \mathcal{P}_i of it. This set \mathcal{P}_i is called the predictor set for the i -th gene. Specifying the truth table for the functions f_1, f_2, \dots, f_n along with the associated predictor sets $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_n$ supplies all the information necessary to determine the time evolution of the states of the BN.

The binary n -digit state vector $x(k)$ can be mapped to positive integers $z(k)$ so that as $x(k)$ ranges from $00 \dots 0$ to $11 \dots 1$, $z(k)$ goes from 1 to 2^n . Here we employ the decimal representation $z(k)$ and the set $\mathcal{S} = \{1, 2, \dots, 2^n\}$ constitutes the state space for the Boolean network.

Attractors play a key role in Boolean networks. Given a starting state, within a finite number of steps, the network will transition into a cycle of states, called an attractor cycle, and will continue to cycle thereafter. Non-attractor states are transient and are visited at most once on any network trajectory. The level of a state is the number of transitions required for the network to transition from the state into an attractor cycle. Attractors are often identified with phenotypes (Kauffman, 1993). Real biological systems are typically assumed to have short attractor cycles. Singleton attractors are a key interest since these are associated with phenomena such as cell proliferation and apoptosis (Huang, 1999). We focus on Boolean networks possessing only singleton attractors.

In most cases we lack time-course gene-expression measurements corresponding to the temporal evolution of the network, and our assumption is that the measurements (or almost all of them) are taken in the steady state (Zhou *et al.*, 2004; Pal *et al.*, 2005a, b)—see Pal *et al.*, 2005b for a discussion of the biological considerations concerning the steady-state assumption. Under this assumption data states are, with probability near one, attractor states. Thus, we would like them to be attractors in the model and their inclusion or lack of inclusion in a designed network can be used to support or not support network validity. If we take the view that there is no reason to believe that data states are not attractor states, then we may wish to require that the attractor states of a designed network exactly match the data states.

To achieve this end, an algorithm has been developed to generate Boolean networks with a prescribed attractor structure (Pal *et al.*, 2005b). To look for biologically meaningful networks in the space of desired networks, the number of predictors for a gene and the number of levels for a transient state are bounded. To avoid the inclusion of non-regulating genes, each gene must occur in the predictor set of at least one other gene. The algorithm can function in two modes. In one, it begins with genes, attractor states, predictor sets and a maximum number of levels, and generates all possible networks having these; in a more general mode, it begins with genes, attractor states, a maximum predictor-set size and a maximum level, and generates networks having these. For instance, in the first mode suppose there are three genes x_1, x_2, x_3 , attractor states 000 and 101, predictor sets $\mathcal{P}_1 = \{x_1, x_2, x_3\}$, $\mathcal{P}_2 = \{x_1, x_3\}$ and $\mathcal{P}_3 = \{x_1\}$, and maximum level 3. Three 3-gene networks with the given singleton attractors, predictor sets and maximum level are shown in Figure 1 using decimal representation.

Control for a probabilistic Boolean network

A PBN consists of a finite collection of BNs over a fixed set of genes, where each BN is defined by a fixed network function. At any given moment of discrete time there is a probability p of randomly switching the state of the PBN, so that each constituent BN is a BN with random perturbation. Moreover, at each moment of time there is a probability q of switching to a different constituent BN, where, given a switch, each BN composing the network has a probability of being selected. If $q = 1$, then a new network function is randomly selected at each time point; if $q < 1$, then the PBN remains in a given constituent BN until the random binary variable governed by q calls for a network switch. If $q = 1$, the PBN is said to be *instantaneously random*, the idea being to model uncertainty in model selection; if $q < 1$, it is said to be *context-sensitive*, the idea being to model the situation where the model is affected by latent variables outside the model. To motivate and to set the background for the control policy proposed in this paper, we briefly review the results of the original analysis for an instantaneously random PBN (Datta *et al.*, 2003)—see Pal *et al.*, 2005a for control in a context-sensitive PBN.

Consider the problem of external control in an instantaneously random PBN with n genes and m control inputs, u_1, u_2, \dots, u_m , each of which can take on only the binary values 0 or 1. At any time k , the row vector $u(k) \triangleq [u_1(k), u_2(k), \dots, u_m(k)]$ describes the complete status of all the control inputs. $u(k)$ can take on all binary values from $00 \dots 0$ to $11 \dots 1$. One can equivalently represent the control input status using a decimal number $v(k)$ ranging from 0 to $2^m - 1$, so that $\mathcal{A} = \{0, 1, \dots, 2^m - 1\}$ is the set of possible control actions. This set could be a function of the state, because not all control alternatives may be available from all states.

As shown in Datta *et al.* (2003), the one-step evolution of the probability distribution vector in the case of a PBN containing 2^n states with control inputs takes place according to the equation

$$w(k+1) = w(k)A(v(k)), \quad (1)$$

where $w(k)$ is the 2^n -dimensional state probability distribution vector and $A(v(k))$ is the $2^n \times 2^n$ control-dependent transition probability matrix (TPM). Since the TPM is a function of the control input $v(k)$, the evolution of the probability distribution vector of the PBN with control now depends not only on the initial distribution vector but also on the values of the control input at different time steps. Intuitively, it appears possible to make the states of the network evolve in a desirable fashion by appropriately choosing the control input at each time step.

We define the following quantities:

- $a_{ij}(v)$ is the i -th row, j -th column entry of the stochastic matrix $A(v)$, $v \in \mathcal{A}$.
- M represents the treatment/intervention window; control actions are taken at steps $0, 1, \dots, M-1$.
- For any $i \in \mathcal{S}$, $C_k(i, v)$ is the cost of applying the control v in state i at the k -th time step.
- For any $i \in \mathcal{S}$, $C_M(i)$ is the terminal cost associated with the state i , i.e. the cost of ending up in state i at the M -th time step, when no more control steps are remaining.

For a given initial state $z(0)$, the *optimal cost* for the finite horizon optimal control problem is given by $J_0(z(0))$, where for $k = 0, 1, 2, \dots, M-1$, $J_k(i)$ represents the *cost-to-go* function at the k -th time step starting from state i (Datta *et al.*, 2003). The $J_k \mathcal{S}$ can be found using the following dynamic programming algorithm (Bellman, 1957):

$$J_k(i) = \min_{v \in \mathcal{A}} \left[C_k(i, v) + \sum_{j \in \mathcal{S}} a_{ij}(v) \cdot J_{k+1}(j) \right], \quad k = M-1, M-2, \dots, 0. \quad (2)$$

$$J_M(i) = C_M(i). \quad (3)$$

Equation (2) states that the optimal cost to go from state i at the k -th time step is the sum of the cost of the optimal control action at state i and the expected value of the cost to go at the $(k+1)$ -th time step. Since there is no control action in the terminal time step, (3) simply formalizes the fact that the cost to go at the terminal time step equals the penalty associated with the terminal state.

The optimal control obtained as a solution to (2), (3) can be represented as a table $\mathcal{S} \times \mathcal{T} \rightarrow \mathcal{A}$, where \mathcal{T} is the discrete time variable. To set up such a table, we first tabulate $J_M(i)$ for any $i \in \mathcal{S}$ using (3). $J_{M-1}(i)$ and the corresponding minimizing control v can be calculated and stored for all $i \in \mathcal{S}$ using (2) and making use of the $J_M(j)$ values tabulated earlier. By repeating these steps we can fill up the table for $k = M-2, \dots, 0$.

The optimal control solution of (2) and (3) is from a very general setting; however, for genetic regulatory networks, the class of allowable controls is severely constrained since the control action consists of forcibly altering the expression status of only one, or perhaps, two genes. This limited control objective is dictated primarily by limitations on the kind of interventions that appear to be within the current realm of biological possibility.

ALGORITHM

If a family of BNs is designed whose attractors match the data, assuming the family is not too small we have the expectation that the underlying biological phenomena are closely modeled by at least some of the BNs in the family. In the absence of perfect knowledge as to which BNs are capable of better representing the underlying phenomena, we develop a control policy that optimizes a composite cost function over the entire family of BNs.

Let \mathcal{N} be a set of L Boolean networks N_1, N_2, \dots, N_L possessing identical sets of singleton attractors, all sharing the same state space \mathcal{S} and the same control space \mathcal{A} . Associated with each network is an initial probability of it representing the underlying phenomenon. Since this information is not available, we will adaptively estimate these probabilities as more transitions are observed. For each network N_l , $l = 1, 2, \dots, L$ define

- $a_{ij}^l(v)$ to be the i -th row, j -th column entry of the matrix $A^l(v)$ of the network N_l ;
- $C_k^l(i, v)$ to be the cost of applying the control v at the k -th time step in state i in network N_l ;
- $C_M^l(i)$ to be the terminal cost associated with state i in network N_l .

We define the *belief vector* $\pi_k = [\pi_k^1, \pi_k^2, \dots, \pi_k^L]$, where π_k^l is the probability of network N_l being the underlying network at the k -th time step. π_k is the probability distribution vector for the family of networks at the k -th time step. Since π_k is unknown, we will make an initial guess for it and update it as more information becomes available. The use of this vector is inspired by the *information vector* in Smallwood *et al.* (1973).

Suppose i is the current state at step k , π is the current estimate of the belief vector, and upon application of control v we observe state j at the next time step. Then the new belief vector is $\pi' = T(\pi, i | j, v)$, where the transformation T can be obtained by use of Bayes' theorem and the theorem of total probability,

$$\pi' = \left[\dots, \frac{a_{ij}^l(v) \cdot \pi_k^l}{\sum_{s \in \mathcal{N}} a_{ij}^s(v) \cdot \pi_k^s}, \dots \right]. \quad (4)$$

Dynamic programming over a family of networks

Suppose we are given an initial belief vector π_0 and an initial state $z(0)$. The initial belief vector is based on our prior knowledge of the system. It could be a function of likelihood or Bayesian scores of networks, or it could be uniform to reflect no prior knowledge. Our objective is to find controls $v(0), v(1), \dots, v(k), \dots, v(M-1)$ to minimize the expectation of the cost-to-go function over all networks in \mathcal{N} . The cost to go at the k -th time step ($0 \leq k < M$) is a function of the current state $z(k)$ and the updated belief vector π_k . Motivated by (2) for the single PBN case, we define the average optimal cost-to-go function by

$$J_k(\pi_k, i) = \min_{v \in \mathcal{A}} \left[\sum_{l \in \mathcal{N}} \pi_k^l \left\{ C_k^l(i, v) + \sum_{j \in \mathcal{S}} a_{ij}^l(v) \cdot J_{k+1}(T(\pi_k, i | j, v), j) \right\} \right] \quad (5)$$

The inner summation is the expectation over all $j \in \mathcal{S}$ of the cost to go at the $(k+1)$ -th step in the l -th network on observing j . We then add to it the cost of control at the k -th step and average over all the networks in the family. Finally we take the minimum over all control actions in \mathcal{A} to obtain the optimal policy and the cost to go at the k -th step.

The terminal cost for a state i is trivially defined to be the average terminal cost over the entire family:

$$J_M(\pi_M, i) = \sum_{l \in \mathcal{N}} \pi_M^l \cdot C_M^l(i). \quad (6)$$

In Datta *et al.* (2003) terminal penalties were assigned to states based on the expression level of certain key genes; however,

as discussed in Choudhary *et al.* (2005), it may be more reasonable to assign terminal penalties based on the long-term prospective behavior of the system in the absence of control. For any Markov chain with closed classes the procedure is summarized by the following points:

- Partition the states of the Markov chain into transient and persistent states.
- For singleton attractors the penalty C_M^l is set according to the status of the penalty gene(s). A *penalty gene* is a gene for which certain expression statuses are known to be undesirable.
- For a closed class the penalty is based on the fraction of time spent in states having a penalty gene in an undesirable profile.
- For a transient state j , the terminal penalty

$$C_M^l(j) = \sum_i \text{Prob}(z(\infty) = i | z(t) = j, \text{Network} = N_l) \cdot C_M^l(i), \quad (7)$$

where the summation variable i corresponds to a closed class or a singleton attractor. $\text{Prob}(z(\infty) = i | z(t) = j, \text{Network} = N_l)$ is the long-term probability of getting absorbed in i starting from state j , in network N_l .

Since the attractors are shared by each network in the family, the attractor states will have the same penalty across the different networks; however, penalties for non-attractor states will differ across networks, depending on the particular attractor in whose basin that non-attractor state may happen to lie in.

IMPLEMENTATION

The solution to the dynamic programming problem (5), (6) will now be presented as a policy tree that is optimal specific to a particular initial state and an initial belief vector. An M -step policy tree has an optimal action as its root with branches for each possible observation (in our case states) followed by $M - 1$ -step policy trees. A detailed exposition on construction and pruning of such trees can be found in Kaelbling *et al.* (1998). Here we state an algorithm that is close to exhaustive enumeration and subsequent pruning. We use a data structure node with five components *STATE*, *BELIEF-VECTOR*, *OPTIMAL-COST*, *OPTIMAL-CONTROL* and *DEPTH*. The algorithm involves the following steps:

- (1) Compute the M step control and the corresponding $J_0(i)$, $J_1(i), \dots, J_M(i) \forall i \in S$ for each of the networks N_1, \dots, N_L as a table. Table 1 is such a table of controls for the example to be presented later in the next section.
- (2) Initialize the tree's root node with the first observed state *STATE* = $z(0)$, *BELIEF-VECTOR* = π_0 and *DEPTH* = 0.
- (3) Expand the root node and all the subsequently generated nodes; while *BELIEF-VECTOR* $\neq e_i$ (i.e. the network N_l is not uniquely identified to be the underlying network) and *DEPTH* $\leq M$. To expand a particular node in the tree with *STATE* = i , *BELIEF-VECTOR* = π_k and *DEPTH* = k , we consider all possible states that could be observed next. In other words, a child node is created for any j , such that $a_{ij}^l(v) > 0$ with $\pi_k^l > 0$. Such a node has *STATE* = j , *BELIEF-VECTOR* = $T(\pi_k, i | j, v)$ and *DEPTH* = $k + 1$.

Table 1. Optimal control policies obtained from different networks

Network	Time step (k)	State							
		1	2	3	4	5	6	7	8
N_1	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	1	1	0	0
N_2	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	1	1	1
N_3	0	0	0	0	0	0	1	0	1
	1	0	0	0	0	1	0	0	1
N_{sw}	0	0	0	0	0	1	1	1	1
	1	0	0	0	0	1	1	1	1

N_{sw} is obtained for $\pi = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$.

- (4) Now consider all the leaf nodes. For the nodes with *DEPTH* = M we use (6) to obtain *OPTIMAL-COST*. For leaf nodes with *DEPTH* = $k \neq M$ and *BELIEF-VECTOR* = e_i and some *STATE* = i , *OPTIMAL-COST* is set to $J_k(i)$ from the table for network N_l .
- (5) Now use (5) for all nodes with *DEPTH* = $M - 1, \dots, 0$ (in that order) to obtain *OPTIMAL-COST* and the minimizing v as the *OPTIMAL-CONTROL*.
- (6) Prune the subtrees generated with non-optimal actions to obtain Pol^{TR} , the policy tree. The optimal policy follows the table for network N_l onwards from a node which has *BELIEF-VECTOR* = e_i .

At first glance, generating the tree may seem to be a formidable task due to the potentially large branching factor which can be as high as $|\mathcal{S}| \times |\mathcal{A}|$. However, in the case of a family of BNs this is a much more manageable task due to the following mitigating factors: (1) the branching factor is small since not all states would be observed following an action due to similarities in the different BN transitions—for instance, in Figure 1 state 3 goes to state 1 in all the three networks; (2) from a given node, if more than one node is generated for some v , then the *BELIEF-VECTOR*s for the children would be more sparse than the parent, and in some cases it would become a leaf node with *BELIEF-VECTOR* = e_i ; and (3) the set of possible control actions \mathcal{A} is not large owing to the limited number of genes for intervention.

DISCUSSION

Illustrative example

To illustrate the algorithmic details, we consider the 3-gene network introduced previously. Suppose state 1 (000) is a desirable attractor state with terminal penalty 0 and state 6 (101) is an undesirable state with terminal penalty +5. The terminal cost of any other non-attractor state is the cost of the attractor whose basin it is in. For instance for the network in Figure 1 (1.2), the non-attractor states 2, 3, 4 and 5 have terminal penalty 0, while states 7 and 8 have terminal penalty +5. Let x_1 be the control gene and suppose that the control action is to forcibly flip this gene: for $v(k) = 1$, flip gene x_1 at the k -th time step and for $v(k) = 0$ leave it as is. Let the cost of control $C_k^l(i, v) = C(v) = v$. Transitions take place according to the network transition rule—e.g. in the network in Figure 1 (1.2), if $z(k) = 6$ (101) and $v(k) = 1$, then $z(k+1) = 1$, corresponding to a jump from

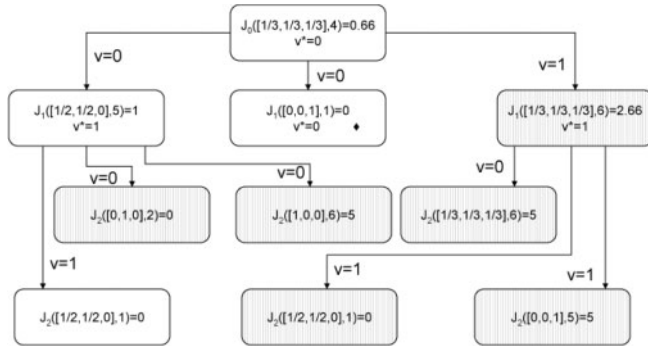


Fig. 2. Tree calculation for initial belief vector $\pi_0 = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$, initial state $z(0) = 4$. The shaded region is pruned. Closed diamond is a leaf node at $DEPTH < M$.

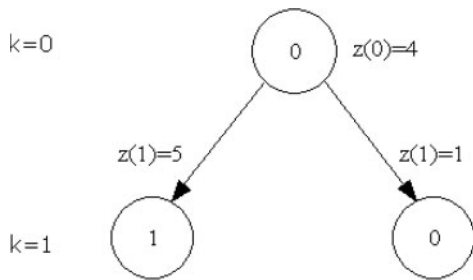


Fig. 3. Pruned policy tree. The number inside the circle is the optimal control action. The arc corresponds to the next observation, which leads to the next optimal control action.

state 6 (101) to state 2 (001) and then evolution to the state 1 (000). We show the evaluation of the policy tree Pol^{TR} starting from an initial state $z(0) = 4$ and $\pi_0 = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ in Figure 2. Figure 3 shows the corresponding policy tree obtained after pruning.

We now proceed to compare the performance of an $M = 2$ step policy Pol^{TR} and policies obtained using two other methods to control this family of networks.

Single network. We calculate the optimal policy Pol^i for each network N_i in the family. We obtain the control policy as a table with M rows and $|\mathcal{S}|$ columns. Each element $v(m, i)$ is the control alternative to be used when the state is i at the m -th time step. Since a single-network policy does not apply to the entire family, to implement a policy tree we follow one of the possible state observations after each action. It may happen that some of the possible states observed may not be listed as an option in a single-network policy tree. For a single BN the policy tree is a tree with a branching factor of 1, i.e. a path. Single network optimal policies for each network are listed in Table 1.

Context switching. The context-sensitive PBN design of Pal et al. (2005a) is more general than the method proposed here because there is no requirement that the constituent BNs possess identical attractor structures; however, it is more constrained in the sense that it assumes knowledge of the PBN switching structure. If, as is assumed here, we lack knowledge of the probabilistic structure governing BN selection so that we do not view the family of BNs as composing a single PBN and if the attractor structures are identical, as with a design strategy in which the attractors of each BN match the data states, then it may well be that the policy proposed here

Table 2. Performance of control with $\pi_0 = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$, $M = 2$

Policy	$J_0^{Pol}(1)$	$J_0^{Pol}(2)$	$J_0^{Pol}(3)$	$J_0^{Pol}(4)$	$J_0^{Pol}(5)$	$J_0^{Pol}(6)$	$J_0^{Pol}(7)$	$J_0^{Pol}(8)$
Pol^1	0.00	0.33	0.00	0.66	2.00	2.66	0.33	2.66
Pol^2	0.00	1.66	0.00	1.66	0.66	2.66	0.33	2.66
Pol^3	0.00	0.33	0.00	0.66	2.00	1.33	1.66	1.66
Pol^{SW}	0.00	0.33	0.00	0.66	1.00	1.33	1.00	1.66
Pol^{TR}	0.00	0.33	0.00	0.66	0.66	1.33	0.33	1.66

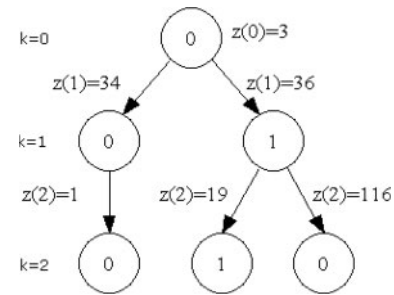


Fig. 4. Policy tree for $M = 3$, initial state $z(0) = 3$ and initial belief vector $\pi_0 = [\frac{1}{4}, \frac{1}{4}, \frac{1}{4}]$.

could outperform the context-sensitive-PBN method. If, for the present 3-gene example, we assume that the BNs compose a PBN in which each has equal probability of being selected, then the method of Pal et al. (2005a) yields the optimal control policy, Pol^{SW} , presented in the last row of Table 1.

To assess the performance of a particular policy Pol , we apply it to all the networks in the family starting from each initial state i and obtain $J_0^{i, Pol}(i)$. We then compute

$$J_0^{Pol}(i) = \sum_{l \in \mathcal{N}} J_0^{l, Pol}(i) \cdot \pi^l \quad (8)$$

by averaging over all the networks. Assuming that $\pi_0 = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ Table 2 shows the results of applying various policies for all possible initial states. As measured by the value of the optimal cost, the policy Pol^{TR} of this work is superior. More examples appear on the companion website.

Melanoma application

We now apply the methodology of this paper to derive an optimal intervention strategy for a family of gene regulatory networks. These networks were developed from data collected in a study of metastatic melanoma in which the abundance of messenger RNA for the gene WNT5A was found to be highly discriminating between cells with properties typically associated with high metastatic competence versus those with low metastatic competence (Bittner et al., 2000). These findings were validated and expanded in a second study in which experimentally increasing the levels of the Wnt5a protein secreted by a melanoma cell line via genetic engineering methods directly altered the metastatic competence of that cell as measured by the standard *in vitro* assays for metastasis (Weeraratna et al., 2000). A further finding was that an intervention that blocked the Wnt5a protein from activating its receptor by the use of an antibody that binds the Wnt5a protein could substantially reduce Wnt5a’s ability to induce a metastatic phenotype.

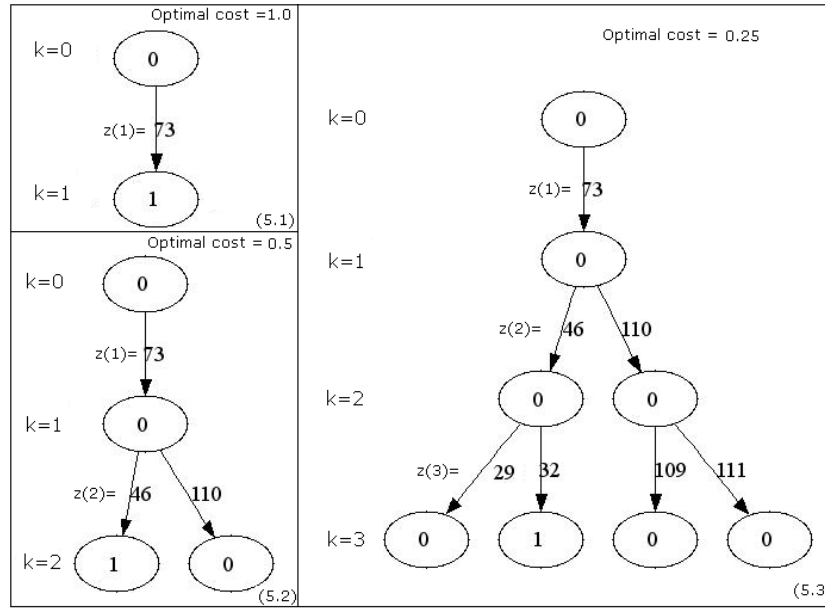


Fig. 5. Policy trees and optimal costs, for initial state $z(0) = 93$, $\pi_0 = [\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}]$, $M = 2$ (5.1), $M = 3$ (5.2) and $M = 4$ (5.3).

Table 3. Cluster centers as attractors for the WNT5A network

	z	Gene activity profile x						
		PIRIN	S100P	RET1	MART1	HADHB	STC2	WNT5A
BAD	4	0	0	0	0	0	1	1
	32	0	0	1	1	1	1	1
	82	1	0	1	0	0	0	1
GOOD	33	0	1	0	0	0	0	0
	57	0	1	1	1	0	0	0
	95	1	0	1	1	1	1	0
	109	1	1	0	1	1	0	0

The good attractors are the ones with the profile of WNT5A gene downregulated. PIRIN is the most significant bit(MSB) and WNT5A is the least significant bit(LSB).

This suggests a study of control based on interventions that alter the contribution of the WNT5A gene’s action to biological regulation, since disruption of this influence could reduce the chance of a melanoma metastasizing a desirable outcome.

The original study used 587 genes with 31 gene expression patterns with varying stress conditions. A network with 587 genes would be intractable to design and subsequently control. Consequently the network was reduced to seven genes believed to be the most closely knit: PIRIN, S100P, RET1, MART1, HADHB, STC2 and WNT5A (Datta *et al.*, 2004). Since all 31 data points correspond to steady-state behavior, they should be considered as attractors in the networks. However, out of the 31 samples only 18 were distinct. To reduce the number of attractors, we formed seven clusters from the data points and treated the cluster centers as attractors. These attractors are shown in Table 3. The first column is used to classify them into two categories, GOOD and BAD, depending on the status of the WNT5A gene.

Using the procedure of Pal *et al.* (2005b), we obtained four distinct BNs (N_1, N_2, N_3, N_4) with the same set of seven attractors.

These networks are available on the companion website. We assigned a penalty of 5 to all states in the basin of the undesirable attractors (WNT5A = 1) and 0 to all the other states. We used PIRIN as the control gene. A forcible alteration in the expression level of PIRIN is associated with $v = 1$ while $v = 0$ represents no control. In a reasoning similar to our previous work (Datta *et al.*, 2003, 2004), a terminal penalty of 5 for bad states versus 0 for good states and a control cost of 1 for intervention versus 0 for no intervention is our attempt to capture the intuitive notions of the relative costs of ending up in desirable versus undesirable state and the cost of intervention.

A pruned policy tree for $M = 3$ with initial belief vector $\pi_0 = [\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}]$ and initial state $z(0) = 3$ is shown in Figure 4. The expected cost is 0.75 when we control using Pol^{TR} , 1.5 when using Pol^{SW} , and 1.75, 2.5, 1.5 and 1.75 when using Pol^1 , Pol^2 , Pol^3 and Pol^4 , respectively. The expected uncontrolled cost is 2.5. For all horizons M and all initial states $z(0) = i \in \mathcal{S}$ the method of this paper is superior to those discussed in the earlier section. Out of the 128 states in the network, 89 states needed to be controlled in at least one of the 4 networks. In particular for $M = 5$, starting from such states Pol^{TR} was more effective than Pol^{SW} in reducing the cost by 0.1152 on average. In terms of absolute probabilities Pol^{TR} was able to take the system to a desirable attractor starting from all initial states and all networks with a probability 1.0, except for states 4, 36, 68 and 100 in network N_2 , which are uncontrollable from PIRIN. For Pol^{SW} , states 4, 8, 24, 36, 68 and 100 are not taken to a desirable attractor in N_2 . In the event of N_2 being the underlying network, starting from states 4, 36, 68, 100, Pol^{TR} recognizes this and gives up promptly, while Pol^{SW} keeps on applying control, incurring extra costs, without any extra benefit.

Policy trees for initial state $z(0) = 93$, $\pi_0 = [\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}]$ and $M = 2, 3$ and 4 are shown in Figure 5. The expected cost with $M = 2$ is 1.0, which can be further reduced to 0.25 if $M \geq 4$. This is reasonable because the algorithm has more time steps to identify and control the system. For $M = 4$, the policy computation took 0.28 s on

a 2.4 GHz, P4 processor system. More examples appear on the website.

CONCLUDING REMARKS

In conclusion, we have developed a method to optimally control a family of BNs that share a common attractor structure. Such a family arises naturally from the steady-state data obtained from gene expression microarrays. The control algorithm is presented as a policy tree depending on an initial belief vector that is updated in an adaptive fashion. At every stage of the evolution, the estimated belief vector is used to appropriately weight the individual networks in the construction of the composite cost function to be minimized.

ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation under grants ECS-0355227 and CCF-0514644, and by the Translational Genomics Research Institute.

Conflict of Interest: none declared.

REFERENCES

Bittner, M. et al. (2000) Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature*, **406**, 536–540.

- Bellman, R. (1957) *Dynamic Programming*. Princeton University Press, Princeton, NJ.
- Choudhary, A. et al. (2005) Assignment of terminal penalties in gene regulatory networks. In *Proceedings of the American Control Conference*, Portland, 417–422.
- Datta, A. et al. (2003) External control in Markovian genetic regulatory networks. *Mach. Learning*, **52**, 169–191.
- Datta, A. et al. (2004) External control in Markovian genetic regulatory networks: the imperfect information case. *Bioinformatics*, **20**, 924–930.
- Huang, S. (1999) Gene expression profiling, genetic networks, and cellular states: an integrating concept for tumorigenesis and drug discovery. *Mol. Med.*, **77**, 469–480.
- Kaelbling, L. et al. (1998) Planning and acting in partially observable stochastic domains. *Artif. Intell.*, **101**, 99–134.
- Kauffman, S. (1969) Metabolic stability and epigenesis in randomly constructed genetic nets. *Theor. Biol.*, **22**, 437–467.
- Kauffman, S. (1993) *The Origins of Order: Self-organization and Selection in Evolution*. Oxford University Press, NY.
- Pal, R. et al. (2005a) Intervention in context-sensitive probabilistic Boolean networks. *Bioinformatics*, **21**, 1211–1218.
- Pal, R. et al. (2005b) Generating Boolean networks with a prescribed attractor structure. *Bioinformatics*, **21**, 4021–4025.
- Smallwood, R.D. and Sondik, E.J. (1973) Optimal control of partially observable Markov processes over a finite horizon. *Operations Res.*, **21**, 1071–1088.
- Shmulevich, I. et al. (2002a) Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, **18**, 261–274.
- Shmulevich, I. et al. (2002b) Gene perturbation and intervention in probabilistic Boolean networks. *Bioinformatics*, **18**, 1319–1331.
- Weeraratna, A. et al. (2002) Wnt5a signaling directly affects cell motility and invasion of metastatic melanoma. *Cancer Cell*, **1**, 279–288.
- Zhou, X. et al. (2004) A Bayesian connectivity-based approach to constructing probabilistic gene regulatory networks. *Bioinformatics*, **20**, 2918–2927.