Point-Based Methodology to Monitor and Control Gene Regulatory Networks via Noisy Measurements

Mahdi Imani, *Student Member, IEEE* and Ulisses M. Braga-Neto, *Senior Member, IEEE*

***Abstract*—This paper proposes a methodology to monitor and control gene regulatory networks (GRNs) via noisy measurements in an infinite observation space. Towards this end, we employ the partially-observed Boolean dynamical systems (POBDS) signal model. The proposed methodology consists of offline and online steps. In the offline step, a family of point-based methods is applied to the POBDS model to gather the necessary control policy prior to the online (execution) step. This is accomplished by developing efficient backup and belief expansion processes to make the computation scale with the log of the number of states, as opposed to the complexity of existing point-based methods, which grows with the number of states. In the online step, simultaneous monitoring and control is achieved by a one-step look-ahead search procedure using the optimal state estimation algorithm for the POBDS model, known as the Boolean Kalman Filter (BKF), as well as the information gathered in the offline step. The online one-step look-ahead process confers robustness to changes in system dynamics, possibility of starting the execution process before the completion of the offline step. The use of the BKF for simultaneous monitoring and control during the online stage can be key in assessing possible side effects of intervention. The performance of the proposed methodology is investigated through a comprehensive set of numerical experiments using synthetic gene expression data generated from a melanoma gene regulatory network.**

***Index Terms*—Infinite-Horizon Control, Gene Regulatory Networks, Partially-Observed Boolean Dynamical Systems, Point-Based Methods, Boolean Kalman Filter.**

I. Introduction

A key purpose of control of gene regulatory networks (GRNs) is to derive intervention strategies to avoid undesirable states, such as those associated with disease. GRNs play a crucial role in every process of life, including cell differentiation, metabolism, the cell cycle and signal transduction [1]. It is often the case that the transcriptional state of each gene is represented by 0 (OFF) or 1 (ON), and the relationship among genes is described by logical gates updated at discrete time intervals [2], i.e., through a Boolean dynamical system. These models were first introduced as a completely-observable, deterministic model by Kauffman and collaborators [3]. Several other models have been introduced in the literature to mathematically capture the behavior of gene regulatory networks. These models include probabilistic Boolean network (PBN) [4], Bayesian networks [5], and Boolean control networks [6]. Several intervention strategies were also developed for control of GRNs (e.g. [7]–[9]).

All aforementioned methods assume that the Boolean states of the system are directly observable. But, in practice, this is never the case. Modern transcriptional studies are based on technologies that produce noisy indirect measurements of gene activity, such as cDNA microarrays [10], RNA-seq [11], and cell imaging-based assays [12]. Thus, the goal of this paper is to obtain intervention strategies for gene regulatory networks observed through noisy gene expression measurements to beneficially alter network dynamics (e.g shifting the steady-state mass from undesirable cell transcriptional states, such as cell proliferation states that may be associated with cancer [7], to desirable states).

Toward this end, we employ the partially-observed Boolean dynamical system (POBDS) signal model [13], [14]. This signal model is capable of addressing noisy observations as well as incomplete measurements (e.g., some of the genes in a pathway or gene network are not monitored). In the POBDS model, there are two layers or processes: the Boolean dynamical system, which is the hidden state process, while the observation layer models the actual data that are available to the researcher. Several tools for POBDSs have been developed in recent years, such as the optimal filter and smoother based on the minimum mean square error (MMSE) criterion — as opposed to the maximum-a-posteriori (MAP) criterion usually employed with finite state spaces — called the Boolean Kalman Filter (BKF) [13], [15] and Boolean Kalman Smoother (BKS) [16], respectively. In addition, particle filtering implementations of these filters, as well as schemes for simultaneous state and parameter estimation for POBDS were introduced in [13], [14].

In [17], [18], a state feedback controller for POBDSs was proposed based on optimal infinite horizon control of the Boolean state process, with the Boolean Kalman filter as state observer. This method, which is called V_BKF in this paper, has similarities to the Q_MDP method introduced in [19] for the general state space model, in which the control policy is not obtained in the belief space. Although this type of controller can be effective in some domains, the obtained policies by these methods do not take informed control action and might perform poorly in domains where repeated information gathering is necessary [20]–[23].

The methodology proposed in this paper is based on point-based techniques [23]. With the advent of powerful computation, point-based methods were developed to find the

M. Imani and U.M. Braga-Neto are with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: m.imani88@tamu.edu,ulisses@ece.tamu.edu)

approximate solution for partially-observed Markov decision processes (POMDPs) [24]. Several variations of these methods have been developed recently [20], [25]–[28]. A basic point-based method for POBDS with finite observation space was introduced in [29]. Most existing point-based techniques are designed to find control strategies for systems with finite observation spaces. Despite several attempts to adapt these techniques to systems with infinite or large observation spaces, the very large computational complexity required effectively prevents their application in real practical problems.

The proposed approach addresses these issues by developing efficient backup and belief expansion processes, two key elements in the family of point-based methods, which exploit the Boolean vectorial structure of the state space and the gene expression measurement model so that computation scales with the log of the number of states (more on this below). While the proposed backup and expansion processes allow adaptation of any point-based method, we adopt two specific point-based methods, PBVI [23] and Perseus [20], to gather necessary information in the belief space before starting execution. Using information from this offline or planning stage, an online controller is proposed, which combines a one-step look-ahead search process and the Boolean Kalman Filter for both monitoring and decision making purposes during the execution process. Our numerical results indicate that the proposed Perseus-POBDS and PBVI-POBDS methods both display a smaller cost per step than the competing Q_MDP and V_BKF methods. In addition, as the belief set size increases, the cost per step achieved by the proposed algorithms approaches the optimal cost of an ideal controller that has direct access to the system states.

The main contributions of this paper can be summarized as follows:

1) The proposed offline controller takes advantage of two main features of the problem at hand, namely, the Boolean vectorial structure of the state process and the conditional independence of the gene-expression measurements, to derive efficient backup and belief expansion processes that scale with the log of the number of states, allowing the application of the proposed method to practical systems. The complexity of existing point-based methods for infinite observation spaces scale with the number of states, making them applicable only to very small systems. For example, with 7 genes, as in the Melanoma GRN example used in the numerical experiments of Section VII, the proposed methodology samples from a 7-dimensional Gaussian distribution, while existing methods would sample from a Gaussian distribution in a space of $2^7 = 128$ dimensions.
2) The proposed online controller method uses the policy computed by the offline controller as well as a one-step look-ahead search method for decision making process. The main advantages of the proposed online controller are the robustness in the presence of slight changes in the system's dynamics, possibility of starting the execution process before the completion of the offline step, improving the accuracy of the offline controller's policy, and ability of prescribing the proper control input in real-time.
3) The proposed method is able to monitor and control simultaneously the GRN during the online stage. This can be crucial in assessing possible side effects of control (e.g. in drug intervention), as well as capturing possible changes in the dynamics of the system. This is achieved by incorporating the optimal MMSE state estimator, namely, the Boolean Kalman filter (BKF), into the online controller process.

The article is organized as follows. In Section II, the POBDS model is reviewed. Then, the infinite-horizon control problem is formulated in Section III. In Section IV, key components of our point-based methods for POBDS are developed, followed by the proposed offline and online controllers in Section V. The simultaneous monitoring and control methodology is discussed in Section VI. Results of a numerical experiment using a melanoma gene regulatory network observed through synthetic gene expression time series are reported and discussed in Section VII. Finally, Section VIII contains concluding remarks.

II. POBDS Model

In this section, the POBDS model is briefly introduced. It consists of a state model that describes the evolution of the Boolean dynamical system, which includes the system input, and an observation model that relates the state to the system output (measurements). More details can be found in [13], [15].

A. POBDS State Model

Assume that the system is described by a *state process* $\{\mathbf{X}_k; k = 0, 1, \ldots\}$, where $\mathbf{X}_k \in \{0,1\}^d$ represents the activation/inactivation state of the genes at time k. The state of genes is affected by a sequence of *control inputs* $\{\mathbf{u}_k; k = 0, 1, \ldots\}$, where $\mathbf{u}_k$ takes values in a finite set $\mathbb{U}$ and represents a purposeful control input into the system state. The states are assumed to be updated at each discrete time through the following nonlinear signal model:

$$\mathbf{X}_k \;=\; \mathbf{f}\left(\mathbf{X}_{k-1}, \mathbf{u}_{k-1}\right) \,\oplus\, \mathbf{n}_k\,, \tag{1}$$

for $k = 1, 2, \ldots$, where $\mathbf{f} : \{0,1\}^d \times \mathbb{U} \to \{0,1\}^d$ is a Boolean function, called the *network function*, $\mathbf{n}_k \in \{0,1\}^d$ is Boolean transition noise, and "$\oplus$" indicates componentwise modulo-2 addition. The process noise $\{\mathbf{n}_k; k = 1, 2, \ldots\}$ is assumed to be "white" in the sense that the noise values at distinct time points are independent. It is also assumed that the noise process is independent from the initial state $\mathbf{X}_0$.

Let $(\mathbf{x}^1, \ldots, \mathbf{x}^{2^d})$ be an arbitrary enumeration of the possible state vectors. The *controlled* transition matrix of the state process $\{\mathbf{X}_k; k = 0, 1, \ldots\}$ will be required later:

$$\begin{aligned}(M_k(\mathbf{u}))_{ij} \;&=\; P(\mathbf{X}_k = \mathbf{x}^i \mid \mathbf{X}_{k-1} = \mathbf{x}^j, \mathbf{u}_{k-1} = \mathbf{u}) \\ &=\; P\left(\mathbf{n}_k \;=\; \mathbf{f}(\mathbf{x}^j, \mathbf{u}) \oplus \mathbf{x}^i\right)\,,\end{aligned} \tag{2}$$

for each $\mathbf{u} \in \mathbb{U}$ and $i, j = 1, \ldots, 2^d$.

B. POBDS Observation Model

In this paper, we assume a POBDS observation model that corresponds to Gaussian gene expression measurements at each time point. This is an appropriate model for many important gene-expression measurement technologies, such as cDNA microarrays [10] and live cell imaging-based assays [12], in which gene expression measurements are continuous and unimodal (within a single population of interest).

Let $\mathbf{Y}_k = (\mathbf{Y}_k(1), \ldots, \mathbf{Y}_k(d))$ be a vector containing the measurement at time k, for $k = 1, 2, \ldots$ The component $\mathbf{Y}_k(j)$ is the abundance measurement corresponding to transcript j, for $j = 1, \ldots, d$. We assume conditional independence of the transcript counts given the state:

$$\begin{aligned} &p(\mathbf{Y}_k = \mathbf{y} \mid \mathbf{X}_k = \mathbf{x}) \\ &\quad = \prod_{j=1}^{d} p(\mathbf{Y}_k(j) = \mathbf{y}(j) \mid \mathbf{X}_k(j) = \mathbf{x}(j)) \,, \end{aligned} \tag{3}$$

and adopt the Gaussian model:

$$\begin{aligned} p(\mathbf{Y}_k(j) =& \mathbf{y}(j) \mid \mathbf{X}_k(j) = \mathbf{x}(j)) \\ &= \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left(-\frac{(\mathbf{y}(j) - \mu_j)^2}{2\,\sigma_j^2}\right) , \end{aligned} \tag{4}$$

where μ_j and $\sigma_j > 0$ are the mean and standard deviation of abundance of transcript j, respectively, for $j = 1, \ldots, d$.

According to the Boolean state model, there are two possible states for the abundance of transcript j: high, if $\mathbf{x}(j) = 1$, and low, if $\mathbf{x}(j) = 0$. We thus model the parameters μ_j and σ_j as:

$$\begin{aligned} \mu_j &= \mu_j^0\,(1 - \mathbf{x}(j)) + \mu_j^1\,\mathbf{x}(j)\,, \\ \sigma_j &= \sigma_j^0\,(1 - \mathbf{x}(j)) + \sigma_j^1\,\mathbf{x}(j)\,, \end{aligned} \tag{5}$$

where the parameters $(\mu_j^0, \sigma_j^0 > 0)$ and $(\mu_j^1, \sigma_j^1 > 0)$ are the means and standard deviations of the abundance of transcript j in the inactivated and activated states, respectively.

Based on equations (4) and (5), the *update matrix* is a diagonal matrix with diagonal elements given by:

$$\begin{aligned} &(T_k(\mathbf{y}))_{ii} = p\left(\mathbf{Y}_k = \mathbf{y} \mid \mathbf{X}_k = \mathbf{x}^i\right) \\ &= \left(\prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\left(\sigma_j^0\,(1 - \mathbf{x}^i(j)) + \sigma_j^1\,\mathbf{x}^i(j)\right)^2}}\right) \\ &\times \exp\left(-\sum\nolimits_{j=1}^{d} \frac{\left(\mathbf{y}(j) - \mu_j^0(1 - \mathbf{x}^i(j)) - \mu_j^1\mathbf{x}^i(j)\right)^2}{2\left(\sigma_j^0(1 - \mathbf{x}^i(j)) + \sigma_j^1\mathbf{x}^i(j)\right)^2}\right), \end{aligned} \tag{6}$$

for $i = 1, \ldots, 2^d$. Typical values for all parameters are given in Section VII. In practice, these values (e.g. the means or standard deviations of the abundance of transcripts) might not be fully-known. In that case, one can apply a maximum-likelihood adaptive filter for POBDS [13], [14] to the available data sequence, prior to application of the proposed method, to estimate these parameters.

III. Infinite-Horizon Control

In this section, the infinite-horizon control problem for the POBDS model is formulated. Our goal is to select the appropriate external input $\mathbf{u}_k \in \mathbb{U}$ at each time k to make the network spend the least amount of time, on average, in undesirable states (e.g., states corresponding to cell proliferation, which may be associated with cancer [33]).

We will assume that the system prediction matrix $M_k(\mathbf{u})$ and update matrix $T_k(\mathbf{y})$ can only depend on time through the control input $\mathbf{u} \in \mathbb{U}$ and measurement $\mathbf{y} \in \mathbb{R}^d$, respectively. We will thus drop the index k and write simply $M(\mathbf{u})$ and $T(\mathbf{y})$.

Since the state of the system is not observed directly, all that is available for decision making at each time step are the observations up to the current time $\mathbf{y}_{1:k} = (\mathbf{y}_1, \ldots, \mathbf{y}_k)$, and the control inputs applied to the system at the previous times $\mathbf{u}_{0:k-1} = (\mathbf{u}_1, \ldots, \mathbf{u}_{k-1})$. Rather than storing the history of observations and control inputs, we record the probability of the state given that information at each time step. This probability distribution is known as the *belief state* at time k,

$$\mathbf{b}_k(i) = P(\mathbf{X}_k = \mathbf{x}^i \mid \mathbf{y}_{1:k}, \mathbf{u}_{0:k-1})\,, \tag{7}$$

for $i = 1, \ldots, 2^d$. The initial belief state is simply the initial state distribution, $\mathbf{b}_0(i) = P(\mathbf{X}_0 = \mathbf{x}^i)$, for $i = 1, \ldots, 2^d$. Since $0 \le \mathbf{b}(i) \le 1$ and $\sum_{i=1}^{2^d} \mathbf{b}(i) = 1$, a belief vector $\mathbf{b}_k$ is a point in a $(2^d - 1)$-dimensional simplex $\mathbb{B}$, called the *belief space*.

If $\mathbf{b}$ is the current belief state of the system, a control input $\mathbf{u}$ is applied, and observation $\mathbf{y}$ is made, the new belief can be obtained by using Bayes' rule as:

$$\mathbf{b}^{\mathbf{u},\mathbf{y}} = \frac{T(\mathbf{y})\,M(\mathbf{u})\,\mathbf{b}}{\|T(\mathbf{y})\,M(\mathbf{u})\,\mathbf{b}\|_1}\,, \tag{8}$$

where $\|\cdot\|_1$ denotes the L_1-norm of a vector. Thus, by using the concept of belief state, a POBDS can be transformed into a Markov decision process (MDP) with state transition probability given by:

$$p(\mathbf{b}' \mid \mathbf{b}, \mathbf{u}) = \int_{\mathbf{y} \in \mathbb{R}^d} \|T(\mathbf{y})\,M(\mathbf{u})\,\mathbf{b}\|_1\, I_{\mathbf{b}' = \mathbf{b}^{\mathbf{u},\mathbf{y}}}\, d\mathbf{y}, \tag{9}$$

where $I_{\mathbf{b}' = \mathbf{b}^{\mathbf{u},\mathbf{y}}}$ equals 1 if $\mathbf{b}' = \mathbf{b}^{\mathbf{u},\mathbf{y}}$ and 0 otherwise.

Now, let $c(\mathbf{x}^i, \mathbf{u})$ be a bounded cost of control for state $\mathbf{x}^i$ and control input $\mathbf{u}$, for $i = 1, \ldots, 2^d$ and $\mathbf{u} \in \mathbb{U}$. Collect all these costs in a vector $\mathbf{g}(\mathbf{u}) = [c(\mathbf{x}^1, \mathbf{u}) \ \ldots \ c(\mathbf{x}^{2^d}, \mathbf{u})]^T$ of size 2^d. The costs can be transformed to belief space as follows:

$$g(\mathbf{b}, \mathbf{u}) = \sum_{i=1}^{2^d} c(\mathbf{x}^i, \mathbf{u})\,\mathbf{b}(i) = \mathbf{g}(\mathbf{u})^T\,\mathbf{b}\,. \tag{10}$$

The goal of infinite-horizon control is to minimize

$$J_\infty = E\left[\sum_{k=1}^{\infty} \gamma^k\, g(\mathbf{b}_k, \mathbf{u}_k) \,\middle|\, \mathbf{b}_0\right], \tag{11}$$

where $\mathbf{b}_0$ is the known initial belief state, and the discount factor γ places a premium on minimizing the costs of early interventions as opposed to later ones, which is sensible from a medical perspective [8]. The classical results proved in [34] for MDPs can be used here. For an infinite-horizon control

problem with discount factor γ, the Bellman operator for the belief space $\mathbb{B}$ can be written as follows:

$$T[J](\mathbf{b}) = \min_{\mathbf{u}\in\mathbb{U}} \left[g(\mathbf{b},\mathbf{u}) + \gamma \int_{\mathbf{b}'\in\mathbb{B}} p(\mathbf{b}' \mid \mathbf{b},\mathbf{u})\, J(\mathbf{b}')\, d\mathbf{b}' \right]$$
$$= \min_{\mathbf{u}\in\mathbb{U}} \left[\mathbf{g}(\mathbf{u})^T\mathbf{b} + \gamma \int_{\mathbf{y}\in\mathbb{R}^d} \|T(\mathbf{y})M(\mathbf{u})\mathbf{b}\|_1\, J(\mathbf{b}^{\mathbf{u},\mathbf{y}}) d\mathbf{y} \right]. \tag{12}$$

However, since the belief $\mathbf{b}$ is in the (2^d-1)-dimensional simplex $\mathbb{B}$, computing the Bellman operator in (12) for all belief points is not possible.

It has been shown in [35] that, under minimal regularity conditions, the cost function can be modeled by the lower envelope of a finite set of linear functions. These linear functions are described by $\boldsymbol{\alpha}$-vectors (row vectors). Given the set Λ of all $\boldsymbol{\alpha}$-vectors, the cost function for a given belief point $\mathbf{b}$ can be obtained as:

$$J(\mathbf{b}) \;=\; \min_{\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}\,\mathbf{b}\,, \tag{13}$$

where the multiplication of each $\boldsymbol{\alpha}$-vector (row vector) and $\mathbf{b}$ (vertical vector) results in a scalar which is the expected cost for a given belief point.

Fig. 1 represents the partitions created by three $\boldsymbol{\alpha}$-vectors over continuous belief space for a system with only two states ($d=1$). The solid line shows the approximate cost over belief space in this case. The regions over which each $\boldsymbol{\alpha}$-vector dominates are also specified in Fig. 1.

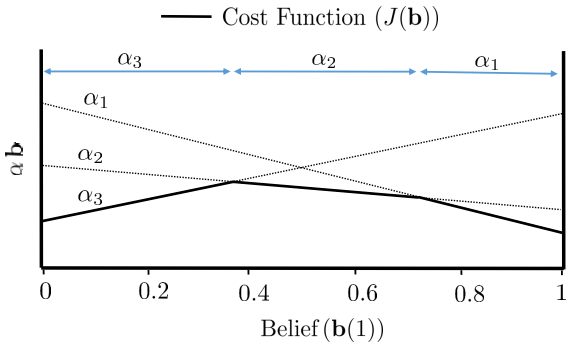

Fig. 1: $\boldsymbol{\alpha}$-vectors in belief space of a system with two states ($d=1$).

The idea is to perform "backup" over the $\boldsymbol{\alpha}$-vectors instead of the Bellman operator in (12) for computation of the optimal policy. *Exact value iteration* [36] achieves this exactly. However, exponential growth of the $\boldsymbol{\alpha}$-vectors limits application of this method to only toy examples with few states and measurements. In the next section, an approximate solution for computing the Bellman operator in equation (12) using the point-based technique will be discussed.

IV. Point-based Technique for POBDS

Point-based techniques were developed to approximate the optimal cost function based on a finite number of points in an infinite belief space [20], [23], [25]–[27]. Consider the following simplification of equation (12):

$$\begin{aligned}
&T[J](\mathbf{b}) \\
&= \min_{\mathbf{u}\in\mathbb{U}} \Bigg[\mathbf{g}(\mathbf{u})^T\mathbf{b} \\
&+ \gamma \int_{\mathbf{y}\in\mathbb{R}^d} \|T(\mathbf{y})M(\mathbf{u})\mathbf{b}\|_1 \left(\min_{\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}\,\mathbf{b}^{\mathbf{u},\mathbf{y}} \right) d\mathbf{y} \Bigg] \\
&= \min_{\mathbf{u}\in\mathbb{U}} \Bigg[\mathbf{g}(\mathbf{u})^T\,\mathbf{b} \\
&+ \gamma \int_{\mathbf{y}\in\mathbb{R}^d} \|T(\mathbf{y})M(\mathbf{u})\mathbf{b}\|_1 \left(\min_{\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha} \frac{T(\mathbf{y})\,M(\mathbf{u})\mathbf{b}}{\|T(\mathbf{y})M(\mathbf{u})\mathbf{b}\|_1} \right) d\mathbf{y} \Bigg] \\
&= \min_{\mathbf{u}\in\mathbb{U}} \left[\mathbf{g}(\mathbf{u})^T\,\mathbf{b} + \gamma \int_{\mathbf{y}\in\mathbb{R}^d} \min_{\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}\, T(\mathbf{y})\, M(\mathbf{u})\,\mathbf{b}\, d\mathbf{y} \right] \\
&= \min_{\mathbf{u}\in\mathbb{U}} \Bigg[\mathbf{g}(\mathbf{u})^T \\
&+ \gamma \int_{\mathbf{y}\in\mathbb{R}^d} \operatorname*{argmin}_{\boldsymbol{\alpha}T(\mathbf{y})M(\mathbf{u}):\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}T(\mathbf{y})M(\mathbf{u})\mathbf{b}\, d\mathbf{y} \Bigg] \mathbf{b}.
\end{aligned} \tag{14}$$

One can now write a compact backup operation that generates a new $\boldsymbol{\alpha}$-vector for a specific belief $\mathbf{b}$ as follows:

$$\text{backup}(\Lambda,\mathbf{b}) \;=\; \operatorname*{argmin}_{\boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}}:\,\mathbf{u}\in\mathbb{U}} \boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}}\,\mathbf{b}\,, \tag{15}$$

where

$$\begin{aligned}
\boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}} \;&=\; \mathbf{g}(\mathbf{u})^T \\
&+ \gamma \int_{\mathbf{y}\in\mathbb{R}^d} \operatorname*{argmin}_{\boldsymbol{\alpha}\,T(\mathbf{y})M(\mathbf{u}):\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}T(\mathbf{y})M(\mathbf{u})\mathbf{b}\, d\mathbf{y}.
\end{aligned} \tag{16}$$

The idea behind most point-based methods is to select a finite set of beliefs and approximate the optimal cost function by performing the backup operator on this finite set. In the following sections, two key ingredients of all point-based methods, which are the computation of backup operator in (15)–(16) and the selection of a finite subset of belief points from set $\mathbb{B}$, are developed for the POBDS model.

A. Backup Operator

Point-based methods were mostly designed to deal with finite observation spaces; however, we are dealing with measurements in an uncountably infinite space ($\mathbf{y}\in\mathbb{R}^d$). The usual approach to address that is to quantize the observation space into a finite set. However, that may introduce a large degree of error in the decision making task. Instead, we employ a sampling-based dynamical partitioning of the observation space for backup computation, which is described next.

From (16), one can see that for given $\mathbf{b}\in\mathbb{B}$ and $\mathbf{u}\in\mathbb{U}$, the integrand can only assume $|\Lambda|$ values over the observation space. Hence, the observation space can be partitioned at most into $|\Lambda|$ equivalent parts, and one can replace the integral in equation (16) by a summation:

$$\boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}} \;=\; g(\mathbf{u})^T + \gamma \sum_{\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}\, F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}})\, M(\mathbf{u})\,, \tag{17}$$

where

$$\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}} = \left\{ \mathbf{y}\in\mathbb{R}^d \;\middle|\; \operatorname*{argmin}_{\boldsymbol{\alpha}'\in\Lambda} \boldsymbol{\alpha}'\,T(\mathbf{y})\,M(\mathbf{u})\,\mathbf{b} = \boldsymbol{\alpha} \right\}, \tag{18}$$

and $F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}})$ is a diagonal matrix defined by:

$$\big(F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}})\big)_{ii} \;=\; \int_{\mathbf{y}\in\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}}} (T(\mathbf{y}))_{ii}\, d\mathbf{y}\,, \tag{19}$$

for $i = 1, \ldots, 2^d$.

Partitioning of measurement space in (17) can be obtained by computing the partition boundaries:

$$\left\{\mathbf{y}\in\mathbb{R}^d \mid \boldsymbol{\alpha}_m\, T(\mathbf{y})\, M(\mathbf{u})\, \mathbf{b} - \boldsymbol{\alpha}_n\, T(\mathbf{y})\, M(\mathbf{u})\, \mathbf{b} = 0\right\}, \tag{20}$$

for all distinct $\boldsymbol{\alpha}_m, \boldsymbol{\alpha}_n \in \Lambda$. However, finding a closed form analytical solution for (20) is not feasible.

In this paper, we use Monte Carlo sampling to approximate the computation in (17)–(19). This approximation is performed by drawing from a proposal distribution $P(\mathbf{y} \mid \mathbf{b}, \mathbf{u})$ and approximating $\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}}$ and $F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}})$, for $\boldsymbol{\alpha} \in \Lambda$.

Assuming Gaussian proposal distributions, let

$$\begin{aligned}\bar{\mathbf{y}}_{0,j} &\sim \mathcal{N}\left([\mu_1^0, \ldots, \mu_d^0]^T, \mathrm{Diag}[(\sigma_1^0)^2, \ldots, (\sigma_d^0)^2]^T\right),\\ \bar{\mathbf{y}}_{1,j} &\sim \mathcal{N}\left([\mu_1^1, \ldots, \mu_d^1]^T, \mathrm{Diag}[(\sigma_1^1)^2, \ldots, (\sigma_d^1)^2]^T\right),\end{aligned} \tag{21}$$

for $j = 1, \ldots, N_s$, where the parameters $(\mu_j^0, \sigma_j^0 > 0)$ and $(\mu_j^1, \sigma_j^1 > 0)$ are the means and standard deviations of the abundance of transcript j in the inactivated and activated states, respectively, which are assumed to be known (c.f. (5)).

Given an input $\mathbf{u} \in \mathcal{U}$, we define

$$\tilde{\mathbf{y}}_j^{\mathbf{u}} \;=\; \bar{\mathbf{y}}_{0,j} \bullet (\mathbf{1}_d - A\, M(\mathbf{u})\, \mathbf{b}) \;+\; \bar{\mathbf{y}}_{1,j} \bullet (A\, M(\mathbf{u})\, \mathbf{b})\,, \tag{22}$$

for $j = 1, \ldots, N_s$, where $\mathbf{1}_d$ is the vector of size d with all element equal to 1, "$\bullet$" denotes componentwise multiplication of vectors, and $A = [\mathbf{x}^1, \ldots, \mathbf{x}^{2^d}]$ is a $d \times 2^d$ matrix containing all Boolean states of the system.

Next the set $\{\tilde{\mathbf{y}}_j^{\mathbf{u}}\}_{j=1}^{N_s}$ is partitioned into $|\Lambda|$ subsets

$$\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}} \;=\; \left\{\tilde{\mathbf{y}}_j \;\middle|\; \operatorname*{argmin}_{\boldsymbol{\alpha}'\in\Lambda} \boldsymbol{\alpha}'\, T(\tilde{\mathbf{y}}_j^{\mathbf{u}})\, M(\mathbf{u})\, \mathbf{b} = \boldsymbol{\alpha}\right\}, \tag{23}$$

for $\boldsymbol{\alpha} \in \Lambda$. Then one need to approximate $F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}})$ in equation (19), for $\boldsymbol{\alpha} \in \Lambda$. To do so, we define 2^d target distributions $P(\mathbf{y} \mid \mathbf{x}^i), i = 1, \ldots, 2^d$ and compute the weights resulting from the ratio of target over proposal as follows:

$$w_i^{\mathbf{u},j} \;=\; \frac{p(\tilde{\mathbf{y}}_j^{\mathbf{u}} \mid \mathbf{x}^i)}{p(\tilde{\mathbf{y}}_j^{\mathbf{u}} \mid \mathbf{b}, \mathbf{u})} \;=\; \frac{\big(T(\tilde{\mathbf{y}}_j^{\mathbf{u}})\big)_{ii}}{||T(\tilde{\mathbf{y}}_j^{\mathbf{u}})\, M(\mathbf{u})\, \mathbf{b}||_1}\,, \tag{24}$$

for $j = 1, \ldots, N_s$, $i = 1, \ldots, 2^d$. Here $w_i^{\mathbf{u},j}$ specifies the weight associated with the jth particle $\tilde{\mathbf{y}}_j^{\mathbf{u}}$ and ith target. Then, the ith diagonal element of $F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}})$ can be approximated using the weights as follows:

$$(F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}}))_{ii} \;\approx\; \frac{1}{\sum_{l=1}^{N_s} w_i^{\mathbf{u},l}} \sum_{\{j:\, \tilde{\mathbf{y}}_j^{\mathbf{u}} \in \mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}}\}} w_i^{\mathbf{u},j}\,, \tag{25}$$

for $i = 1, \ldots, 2^d$. As the Monte-Carlo sample size N_s grows to infinity, the previous approximation is guaranteed to converge to the true value; see for example [37].

The backup procedure for given $\mathbf{b}$ and Λ is summarized in algorithm 1. To appreciate the efficiency of the proposed sampling process, let us assume a GRN with 2^d states. Direct application of a standard point-based technique requires drawing enough Monte Carlo (MC) samples to represent a 2^d-dimensional Gaussian distribution, while the proposed sampling process requires a much smaller number of MC samples to represent a d-dimensional Gaussian distribution, so that the algorithm complexity scales up with the log of the number of states. This is achieved by exploiting the Boolean vectorial structure of the state of genes as well as the gene-expression measurement model. These computational savings allow the application of the proposed method to practical problems involving several genes.

Algorithm 1 BACKUP (Λ,$\mathbf{b}$)

Initial Sampling:
Draw $\{\bar{\mathbf{y}}_{0,j}, \bar{\mathbf{y}}_{1,j}; j = 1, \ldots, N_s\}$ from (21).
Dynamical Partitioning:
for $\mathbf{u} \in \mathbb{U}$ **do**
  $\tilde{\mathbf{y}}_j^{\mathbf{u}} = \bar{\mathbf{y}}_{0,j} \bullet (\mathbf{1}_d - AM(\mathbf{u})\mathbf{b}) + \bar{\mathbf{y}}_{1,j} \bullet (AM(\mathbf{u})\mathbf{b}), j = 1, \ldots, N_s$
  for $\boldsymbol{\alpha} \in \Lambda$ **do**
    $\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}} = \{\tilde{\mathbf{y}}_j^{\mathbf{u}} \mid \operatorname{argmax}_{\boldsymbol{\alpha}'\in\Lambda} \boldsymbol{\alpha}' T(\tilde{\mathbf{y}}_j^{\mathbf{u}}) M(\mathbf{u})\mathbf{b} = \boldsymbol{\alpha}\}$
    for $i = 1, \ldots, 2^d$ **do**
      $w_i^{\mathbf{u},j} = \frac{\left(T(\tilde{\mathbf{y}}_j^{\mathbf{u}})\right)_{ii}}{||T(\tilde{\mathbf{y}}_j^{\mathbf{u}})\, M(\mathbf{u})\, \mathbf{b}||_1}, j = 1, \ldots, N_s$
      $(F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}}))_{ii} = \frac{1}{\sum_{l=1}^{N_s} w_i^{\mathbf{u},l}} \sum_{\{j:\, \tilde{\mathbf{y}}_j^{\mathbf{u}} \in \mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}}\}} w_i^{\mathbf{u},j}$
    end for
    $\boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}} = \mathbf{g}(\mathbf{u})^T + \gamma \sum_{\boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}\, F(\mathbf{Y}_{\boldsymbol{\alpha}}^{\mathbf{b},\mathbf{u}})\, M(\mathbf{u})$.
  end for
end for
Backup Process:
$\boldsymbol{\alpha}^{\mathbf{b}} = \operatorname{argmin}_{\boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}}:\, \mathbf{u}\in\mathbb{U}, \boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}}\, \mathbf{b}$
$\mathbf{u}^{\mathbf{b}} = \operatorname{argmin}_{\mathbf{u}:\, \mathbf{u}\in\mathbb{U}, \boldsymbol{\alpha}\in\Lambda} \boldsymbol{\alpha}_{\mathbf{u},\mathbf{b}}\, \mathbf{b}$.
return $(\boldsymbol{\alpha}^{\mathbf{b}}, \mathbf{u}^{\mathbf{b}})$

B. Belief Expansion

Point-based methods were designed to bound the size of the cost function by computing the cost only over a finite belief set and optimizing the cost function using the point-based procedure. Choosing the belief set has a major effect on the performance of the policy obtained by these methods. Several belief expansion processes have been developed for adaptation of point-based techniques to infinite observation spaces [21]. These methods are mostly based on two main facts:

Expansion based on reachability: The basic idea is that the belief set should be expanded into areas in which the system has more chance of observing them during the execution stage.

Expansion based on coverage: As it was mentioned before, the belief space is a $(2^d - 1)$-dimensional simplex $\mathbb{B}$, and in order to achieve good results, the belief set should be expanded throughout the continuous belief space. This fact will become clear in Section V, when the upper bound error by the PBVI method is computed.

Next, we develop our expansion procedure for the POBDS model, which addresses both aforementioned criteria. Assuming that B is the set of current belief, for each $\mathbf{b} \in B$, first we generate $\{\bar{\mathbf{y}}_{0,j}, \bar{\mathbf{y}}_{1,j}\}$, from (21), and $\tilde{\mathbf{y}}_j^{\mathbf{u}}$ from (22), for all $\mathbf{u} \in \mathbb{U}$. Notice that these samples are from $P(\mathbf{y} \mid \mathbf{b}, \mathbf{u})$, and

as a result they meet the reachability criterion. One should compute successor beliefs as follows:

$$\mathbf{b}_i^{\mathbf{u}} \;=\; \frac{T(\tilde{\mathbf{y}}_i^{\mathbf{u}})\, M(\mathbf{u})\, \mathbf{b}}{\|T(\tilde{\mathbf{y}}_i^{\mathbf{u}})\, M(\mathbf{u})\, \mathbf{b}\|_1}\,, \tag{26}$$

for $i = 1, \ldots, N_e$ and $\mathbf{u} \in \mathbb{U}$. Among all $N_e \times |\mathbb{U}|$ successor beliefs, we chose the one with the largest minimum distance from set B, so as to increase the coverage and diversity of the belief set. The belief expansion procedure is summarized in Algorithm 2. It should be emphasized that similar to the proposed backup process, the belief expansion process is built on the efficient sampling process derived in (22) and thus achieves similar computational savings, i.e., its complexity scales with the log of the number of states.

Algorithm 2 BELIEF_EXPANSION (B)

Initialization:
$B' \leftarrow B$
$B_{\text{new}} \leftarrow \emptyset$
Initial Sampling:
Draw $\{\bar{\mathbf{y}}_{0,j}, \bar{\mathbf{y}}_{1,j}, j = 1, \ldots, N_e\}$ from (21).
Belief Expansion:
for $\mathbf{b} \in B$ **do**
  for $\mathbf{u} \in \mathbb{U}$ and $i = 1, \ldots, N_e$ **do**
    $\tilde{\mathbf{y}}_i^{\mathbf{u}} = \bar{\mathbf{y}}_{0,i} \bullet (\mathbf{1}_d - AM(\mathbf{u})\mathbf{b}) + \bar{\mathbf{y}}_{1,i} \bullet (AM(\mathbf{u})\mathbf{b})$
    $\mathbf{b}_i^{\mathbf{u}} = \frac{T(\tilde{\mathbf{y}}_i^{\mathbf{u}})\, M(\mathbf{u})\, \mathbf{b}}{\|T(\tilde{\mathbf{y}}_i^{\mathbf{u}})\, M(\mathbf{u})\, \mathbf{b}\|_1}$
  end for
  $\mathbf{b}^* = \text{argmax}_{\mathbf{b}_i^{\mathbf{u}}:\, i=1,\ldots,N_e, \mathbf{u}\in\mathbb{U}} \, \min_{\mathbf{b}'\in B'} \|\mathbf{b}_i^{\mathbf{u}} - \mathbf{b}'\|_1$
  $B_{\text{new}} \leftarrow B_{\text{new}} \cup \mathbf{b}^*$
  $B' \leftarrow B' \cup \mathbf{b}^*$
end for
return B_{new}

V. Proposed Offline and Online Controllers

In this section, we describe the offline and online steps of our proposed point-based methodology, which are based on the backup and belief expansion procedures introduced in the previous section.

A. Offline Controller

Our offline controllers are based on two popular point-based methods, PBVI [23] and Perseus [20].

1) Point-Based Value Iteration for POBDS: The point-based value iteration (PBVI) was first developed in [23] to find the optimal control policy of partially-observed markov decision processes (POMDPs) with finite observation spaces. In this section, this method is adapted for POBDS control with an infinite observation space, by using the backup and belief expansion processes presented in Section IV.

Our proposed method, called PBVI-POBDS, is presented in algorithm 3 and contains two main steps:

PBVI update step: The method starts with the initial belief set $B_0 = \{\mathbf{b}_0\}$ and $\Lambda_0 = \{\boldsymbol{\alpha}_0\}$ (a common choice for initial $\boldsymbol{\alpha}_0$ is $\boldsymbol{\alpha}_0(j) = \frac{1}{(1-\gamma)} \max_{i=1,\ldots,2^d, \mathbf{u}\in\mathbb{U}} c(\mathbf{x}^i, \mathbf{u})$, $j = 1, \ldots, d$). In the mth iteration, the backup operator presented in algorithm 1 is applied to all beliefs $\mathbf{b} \in B_{m-1}$ to create the Λ_m set. The process continues until the difference between the costs for all beliefs in B_{m-1} in two consecutive iterations gets smaller than a prespecified threshold. Notice that the order of performing the backup operation on the belief points in B_{m-1} is arbitrary.

Belief expansion step: The current belief set B_{m-1} is expanded using the expansion process presented in algorithm 2 for the next step of the PBVI Update step. With this procedure, the size of the belief set will double after each iteration. The stopping criterion for this method is the size of the belief set being greater than a prespecified number N^{PBVI}.

Algorithm 3 PBVI-POBDS

Initialization:
$B_0 \leftarrow \{\mathbf{b}_0\}$.
$\Lambda_0 \leftarrow \{\boldsymbol{\alpha}_0\}$.
$m \leftarrow 0$.
while $|B_m| \leq N^{\text{PBVI}}$ **do**
Backup Process:
  $m \leftarrow m + 1$.
  $\Upsilon_0 \leftarrow \Lambda_{m-1}$.
  $J'_0(\mathbf{b}) \leftarrow \min_{\boldsymbol{\alpha}\in\Upsilon_0} \boldsymbol{\alpha}\,\mathbf{b}$, for all $\mathbf{b} \in B_{m-1}$.
  $e \leftarrow 0$.
  repeat
    $e \leftarrow e + 1$.
    $\Upsilon_e \leftarrow \emptyset$.
    for $\mathbf{b} \in B_{m-1}$ **do**
      $\boldsymbol{\alpha}^{\mathbf{b}} \leftarrow$ BACKUP($\Upsilon_{e-1}, \mathbf{b}$).
      $\Upsilon_e \leftarrow \Upsilon_e \cup \{\boldsymbol{\alpha}^{\mathbf{b}}\}$.
      $J'_e(\mathbf{b}) \leftarrow \boldsymbol{\alpha}^{\mathbf{b}}\,\mathbf{b}$.
    end for
  until $\max_{\mathbf{b}\in B_{m-1}} |J'_e(\mathbf{b}) - J'_{e-1}(\mathbf{b})| > \epsilon$.
Belief Expansion Process:
  $B_m \leftarrow B_{m-1} \cup$ BELIEF_EXPANSION (B_{m-1}).
  $\Lambda_m \leftarrow \Upsilon_e$.
end while
return (Λ_m)

Assuming Λ_m is the set of vectors obtained by the PBVI-POBDS method in mth iteration, the cost function for any belief point $\mathbf{b}$ in the belief space $\mathbb{B}$ is defined as:

$$J_m(\mathbf{b}) \;=\; \underset{\boldsymbol{\alpha}\in\Lambda_m}{\text{argmin}}\, \boldsymbol{\alpha}\,\mathbf{b}. \tag{27}$$

In the rest of paper, J_m represents the cost function approximated by the PBVI-POBDS in the mth iteration over the whole belief space $\mathbb{B}$. Similarly, J^* denotes the optimal cost function over the whole simplex $\mathbb{B}$. In addition, we use the term $||J_m - J^*||_\infty$ to refer to the maximum absolute difference between the cost function J_m and J^* over the whole belief space $\mathbb{B}$.

In order to study the accuracy of the proposed PBVI offline controller, let ϵ_{B_m} be the maximum distance between any belief point in the $(2^d - 1)$-dimensional simplex $\mathbb{B}$ and the belief set B_m, and let $g_{\max}$ and $g_{\min}$ be the maximum and minimum possible costs, respectively, for any state $\mathbf{x}$ and

control input $\mathbf{u} \in \mathbb{U}$:

$$\begin{aligned} \epsilon_{B_m} &= \max_{\mathbf{b}' \in \mathbb{B}} \min_{\mathbf{b} \in B_m} \|\mathbf{b} - \mathbf{b}'\|_1 \,, \\ g_{\max} &= \max_{\mathbf{u} \in \mathbb{U}, i \in \{1,\ldots,2^d\}} c(\mathbf{x}^i, \mathbf{u}) \,, \\ g_{\min} &= \min_{\mathbf{u} \in \mathbb{U}, i \in \{1,\ldots,2^d\}} c(\mathbf{x}^i, \mathbf{u}) \,. \end{aligned} \tag{28}$$

The following results, which are adapted from [23], are proved in the Appendix.

Lemma 1: If T and $\tilde{T}$ are the optimal transformation and transformation applied by Algorithm 3, respectively, then

$$\|\tilde{T}[J_{m-1}] - T[J_{m-1}]\|_\infty \leq \frac{(g_{\max} - g_{\min})}{1-\gamma} \, \epsilon_{B_m} \,. \tag{29}$$

Theorem 1: The error of Algorithm 3 for any belief set B_m and any horizon m is bounded by

$$\|J_m - J_m^*\|_\infty \leq \frac{(g_{\max} - g_{\min})}{(1-\gamma)^2} \, \epsilon_{B_m} \,. \tag{30}$$

In addition, for a sufficient large horizon m, the distance between J_m and the optimal infinite horizon cost J^* is bounded by

$$\|J_m - J^*\|_\infty \leq \frac{(g_{\max} - g_{\min})}{(1-\gamma)^2} \, \epsilon_{B_m} + \gamma^m \, ||J_0^* - J^*||_\infty \,. \tag{31}$$

The previous results show, in particular, that the difference between J_m and J^* becomes small for large m. The analysis does not include the error introduced in the backup procedure in Algorithm 1 due to Monte-Carlo sampling; however, this error can be made as small as wanted by simply increasing the Monte-Carlo sample size N_s.

2) Randomized Point-Based Value Iteration (Perseus) for POBDS: Another successful offline controller for general POMDPs with finite observation spaces is called *Perseus* [20]. Unlike the PBVI algorithm, which doubles the size of belief set and also computes backup for all belief points at each time step, the Perseus algorithm considers a fixed belief set in all iterations and the backup is performed only for few beliefs at each iteration. The idea behind the Perseus method is to perform a randomized backup that decreases or at least does not increase the cost of all belief points at each time step. As in the case of the PBVI method, the Perseus method is adapted for POBDS control with an infinite observation space, by using the backup and belief expansion processes presented in Section IV.

Our proposed method, called Perseus-POBDS, is presented in Algorithm 4 and contains two main steps:

Belief Generation: The algorithm starts by searching through belief space and finding N^{Pers} reachable belief points with good coverage in belief space $\mathbb{B}$, based on the belief expansion procedure in Algorithm 2.

Perseus Backup: In the second step of algorithm, the Perseus update process starts by initializing $\Lambda_0 = \{\boldsymbol{\alpha}_0\}$, where $\boldsymbol{\alpha}_0(j) = \frac{1}{(1-\gamma)} \arg\max_{\mathbf{u} \in \mathbb{U} \, i=1,\ldots,2^d} c(\mathbf{x}^i, \mathbf{u})$, for $j = 1, \ldots, 2^d$, to guarantee the reduction of cost at each iteration of Algorithm 4. The initial cost of belief points are $J_0(\mathbf{b}) = \boldsymbol{\alpha}_0 \, \mathbf{b}$, for all $\mathbf{b} \in B$. In the mth iteration of the method, one belief point $\mathbf{b}$ is selected randomly from the current set B. The backup $\boldsymbol{\alpha}^{\mathbf{b}}$ for this belief point is computed using Algorithm 1. One of the two following cases can occur: 1) $\boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b} < J_{m-1}(\mathbf{b})$: In this case all belief points $\mathbf{b}' \in B$ such that $\boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b}' < J_{m-1}(\mathbf{b}')$, i.e., all belief points whose their costs reduce under backup vector $\boldsymbol{\alpha}^{\mathbf{b}}$, are removed from B. In addition, $\boldsymbol{\alpha}^{\mathbf{b}}$ is added to set Λ_m for next set of $\boldsymbol{\alpha}$-vectors. 2) $\boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b} \geq J_{m-1}(\mathbf{b})$: the belief point $\mathbf{b}$ is removed from B, and the $\boldsymbol{\alpha}$-vector in the set Λ_{m-1} which minimizes the cost for this belief point is added to Λ_m. Continuing the above process until B is empty completes one step of the Perseus update. Next, one should check the change in the cost functions over two consecutive iterations of the method for different beliefs. If the change is smaller than a prespecified threshold for all belief points, the algorithm stops and returns Λ_m; otherwise, one restarts the process with the entire set B again.

In contrast to PBVI, for which the size of Λ is close to the size of the belief set B, the size of Λ obtained by the Perseus method is usually much smaller than the size of B. This fact reduces the computational costs of the backup procedure in Algorithm 1, which is directly affected by the size of Λ set. In addition, the computational cost of the Perseus method is not significantly affected by the size of belief set B, so that the latter can be chosen to be very large in comparison to a belief set used in the PBVI method. Further comparative analysis between the PBVI and Perseus methods are given in Section VII.

Algorithm 4 Perseus-POBDS

Initialization:
$B \leftarrow \{\mathbf{b}_0\}$
Belief Expansion Process:
while $|B| < N^{\text{Pers}}$ **do**
  $\bar{B} \leftarrow$ BELIEF_EXPANSION (B).
  $B \leftarrow B \cup \bar{B}$.
end while
Backup Process:
$\boldsymbol{\alpha}_0(j) \leftarrow \frac{1}{1-\gamma} \max_{\mathbf{u} \in \mathbb{U}, i=1,\ldots,2^d} c(\mathbf{x}^i, \mathbf{u})$, for $j = 1, \ldots, 2^d$.
$\Lambda_0 \leftarrow \{\boldsymbol{\alpha}_0\}$.
$J_0(\mathbf{b}) \leftarrow \boldsymbol{\alpha}_0 \, \mathbf{b}$, for all $\mathbf{b} \in B$.
$m \leftarrow 0$.
repeat
  $m \leftarrow m + 1$.
  $\Lambda_m \leftarrow \emptyset$.
  $\bar{B} \leftarrow B$.
  while $\bar{B} \neq \emptyset$ **do**
    Choose a random belief point $\mathbf{b} \in \bar{B}$.
    $\boldsymbol{\alpha}^{\mathbf{b}} \leftarrow$ BACKUP$(\Lambda_{m-1}, \mathbf{b})$.
    if $\boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b} < J_{m-1}(\mathbf{b})$ **then**
      $J_m(\mathbf{b}') = \boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b}'$, for $\mathbf{b}' \in \bar{B} : \boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b}' < J_{m-1}(\mathbf{b}')$
      $\bar{B} \leftarrow \{\mathbf{b}' \in \bar{B} : \boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b}' \geq J_{m-1}(\mathbf{b}')\}$.
    else
      $\bar{B} \leftarrow \bar{B} - \{\mathbf{b}\}$.
      $\boldsymbol{\alpha}^{\mathbf{b}} = \arg\min_{\boldsymbol{\alpha}' \in \Lambda_{m-1}} \boldsymbol{\alpha}' \, \mathbf{b}$.
      $J_m(\mathbf{b}) = \boldsymbol{\alpha}^{\mathbf{b}} \, \mathbf{b}$.
    end if
    $\Lambda_m \leftarrow \Lambda_m \cup \{\boldsymbol{\alpha}^{\mathbf{b}}\}$.
  end while
until $\max_{\mathbf{b} \in B} |J_m(\mathbf{b}) - J_{m-1}(\mathbf{b})| < \varepsilon$.
return (Λ_m)

B. Online Controller

Both offline controllers introduced in the previous section are applied before starting execution to obtain the set Λ, which approximates the optimal cost over the whole continuous belief space $\mathbb{B}$. While the policy obtained by the offline controller can be used for control of the GRN, we propose an efficient online controller that uses the information derived in the offline stage, and combines a one-step look-ahead process and the Boolean Kalman Filter, which is the optimal MMSE state estimator. The online controller introduces robustness against (small) changes in the dynamics during the execution process; allows starting the execution process before the completion of the offline step; improves the accuracy of the offline controller's policy; and is capable of prescribing the proper control input in real-time.

Given Λ and the belief $\mathbf{b}_k$ at time k during the execution step, the online controller chooses $\mathbf{u}_k$ by performing a one-step look-ahead search as:

$$\mathbf{u}_k = \underset{\mathbf{u}\in\mathbb{U}}{\operatorname{argmin}} \left[\left(\mathbf{g}(\mathbf{u})^T + \gamma \int_{\mathbf{y}\in\mathbb{R}^d} \underset{\boldsymbol{\alpha}\in\Lambda\, T(\mathbf{y})\, M(\mathbf{u})}{\operatorname{argmin}} \boldsymbol{\alpha}\, T(\mathbf{y})\, M(\mathbf{u})\, \mathbf{b}_k\, d\mathbf{y} \right) \mathbf{b}_k \right]. \tag{32}$$

The control input in (32) can be obtained by performing the backup operator introduced in Algorithm 1 as:

$$\mathbf{u}_k = \text{BACKUP}(\Lambda, \mathbf{b}_k)\,. \tag{33}$$

The previous equation summarizes the entire action performed by the online controller at time step k. The robustness of the proposed online controller to small changes in dynamics during the execution process comes from the fact that one can apply the one-step look-ahead policy without having to recompute the offline control policy. Furthermore, for large GRNs in which convergence of offline controller might be slow, the online controller can be executed in parallel with the offline controller.

The following result, which is proved in the Appendix, guarantees that a better control policy is reached after the one-step look-ahead search from the current belief state.

Theorem 2: Let J be the cost obtained based on results of offline controller (Algorithm 3 or 4), J^{LA} be the cost after performing a look-ahead mapping from current cost function J, and J^* be the optimal cost function. Then,

$$||J^{\text{LA}} - J^*||_\infty \leq \gamma\, ||J - J^*||_\infty\,. \tag{34}$$

The previous result ignores the error introduced in the backup procedure in Algorithm 1 due to Monte-Carlo sampling; however, as mentioned previously, this error can be made as small as wanted by simply increasing the Monte-Carlo sample size N_s.

VI. Point-Based Methodology to Monitor and Control POBDS

In this section, the proposed procedure to monitor and control partially-observed Boolean dynamical systems is presented.

First, we describe briefly the optimal filtering problem, which consists of obtaining the optimal minimum mean square error (MMSE) Boolean estimator $\hat{\mathbf{X}}_{k|k}^{\text{MS}}$ of the current state, given the sequence of observations $\mathbf{y}_{1:k} = (\mathbf{y}_1, \ldots, \mathbf{y}_k)$ and control input $\mathbf{u}_{0:k-1} = (\mathbf{u}_0, \ldots, \mathbf{u}_{k-1})$.

For a Boolean vector $\mathbf{v} \in \{0,1\}^d$, define the *binarized vector* $\overline{\mathbf{v}}$, such that $\overline{\mathbf{v}}(i) = 1$ if $\mathbf{v}(i) > 1/2$ and $\overline{\mathbf{v}}(i) = 0$ otherwise, for $i = 1, \ldots, d$, the *complement vector* $\mathbf{v}^c$, such that $\mathbf{v}^c(i) = 1 - \mathbf{v}(i)$, for $i = 1, \ldots, d$, and the L_1*-norm* $||\mathbf{v}||_1 = \sum_{i=1}^{d} |\mathbf{v}(i)|$. It has been proven in [13] that the MMSE estimator is given by:

$$\hat{\mathbf{X}}_{k|k}^{\text{MS}} = \overline{E\left[\mathbf{X}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{0:k-1}\right]}\,, \tag{35}$$

with optimal conditional MSE as:

$$\begin{aligned} &\text{MSE}(\hat{\mathbf{X}}_{k|k}^{\text{MS}} \mid \mathbf{y}_{1:k},\, \mathbf{u}_{0:k-1}) \\ &\quad = \left|\left| \min\left\{ E\left[\mathbf{X}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{0:k-1}\right], \right.\right. \\ &\qquad\qquad \left.\left. E\left[\mathbf{X}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{0:k-1}\right]^c \right\} \right|\right|_1, \end{aligned} \tag{36}$$

where the minimum is computed componentwise. Both the optimal filter and its MMSE can be computed by a recursive matrix-based procedure, called the Boolean Kalman Filter (BKF) [15].

The combination of the BKF and the previously described online and offline controllers provides the scheme for POBDS monitoring and control, as can be seen in the the schematic diagram in Fig 2. First, an offline controller is run before starting execution to gather the necessary information over the belief space. Then, during execution, the BKF estimates the belief state as well as the Boolean state at each time point given the latest available information. The estimated belief at each time point and the results of offline controller are used by the online controller to select the appropriate control input. The procedure is summarized in Algorithm 5.

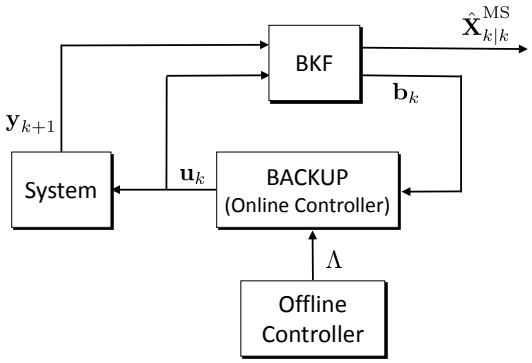

Fig. 2: Schematic diagram of the proposed point-based methodology for POBDS monitoring and control.

VII. Numerical Experiments

In this section, we report the results of an extensive set of numerical experiment using a Boolean model for a gene regulatory network implicated in metastatic melanoma [38]. The network contains 7 genes: WNT5A, pirin, S100P, RET1, MART1, HADHB and STC2. The regulatory relationship for

Algorithm 5 POBDS Monitoring and Control

OFFLINE STEP

1: Run an offline controller (Algorithm 3 or 4) to obtain Λ.

ONLINE STEP

2: Initialization: $\mathbf{b}_0(i) = P\left(\mathbf{X}_0 = \mathbf{x}^i\right)$, for $i = 1, \ldots, 2^d$.

3: $\mathbf{u}_0 \leftarrow$ BACKUP($\mathbf{b}_0$,Λ).

4: **for** $k = 1, 2, \ldots$ **do**

5: Posterior Update:

$$\mathbf{b}_k = \frac{T(\mathbf{y}_k)\, M(\mathbf{u}_{k-1})\, \mathbf{b}_{k-1}}{\|T(\mathbf{y}_k)\, M(\mathbf{u}_{k-1})\, \mathbf{b}_{k-1}\|_1}\,.$$

6: MMSE Estimator Computation: $\hat{\mathbf{X}}_{k|k}^{\mathrm{MS}} = \overline{A\mathbf{b}_k}$.

7: $\mathrm{MSE}(\mathbf{y}_{1:k}, \mathbf{u}_{0:k-1}) = ||\min\{A\mathbf{b}_k, (A\mathbf{b}_k)^c\}||_1$.

8: Online Controller: $\mathbf{u}_k \leftarrow$ BACKUP($\mathbf{b}_k$,Λ).

9: **end for**

this network is presented in Table I. The ith output binary string specifies the output value for ith input gene(s) in binary representation. For example, the last row of Table I specifies the value of STC2 at current time step k from different pairs of (pirin,STC2) values at previous time step $k-1$:

(pirin=0, STC2=0)$_{k-1}$ $\rightarrow$ STC2$_k$=1
(pirin=0, STC2=1)$_{k-1}$ $\rightarrow$ STC2$_k$=1
(pirin=1, STC2=0)$_{k-1}$ $\rightarrow$ STC2$_k$=0
(pirin=1, STC2=1)$_{k-1}$ $\rightarrow$ STC2$_k$=1

In the study reported in [39], the activation status of gene WNT5A was found to be highly predictive of cells with properties typically associated with high metastatic competence versus those with low metastatic competence. Therefore, an intervention that blocked the WNT5A gene from being activated might substantially reduce WNT5A's ability to induce a metastatic phenotype. For more information, the reader is referred to [38].

TABLE I: Boolean functions for the melanoma gene regulatory network.

Genes	Input Gene(s)	Output
WNT5A	HADHB	10
pirin	pirin, RET1, HADHB	00010111
S100P	S100P, RET1, STC2	10101010
RET1	RET1, HADHB, STC2	00001111
MART1	pirin, MART1, STC2	10101111
HADHB	pirin, S100P, RET1	01110111
STC2	pirin, STC2	1101

In [40], reduction of WNT5A's activation is accomplished indirectly, through the control of other gene activities. Accordingly, in our experiments, the intervention is applied to either RET1 or HADHB. In practice, control would be accomplished by means of drugs targeted at those genes. The cost of control is assumed to be 1 for any taken intervention and 0 when there is no intervention. Since the goal of control is preventing WNT5A gene to be upregulated, the cost function is defined as follows:

$$c(\mathbf{x}^j, \mathbf{u}) = \begin{cases} \eta_1 + \eta_2 ||\mathbf{u}||_1 & \text{if WNT5A is 1 for state } j, \\ \eta_2 ||\mathbf{u}||_1 & \text{if WNT5A is 0 for state } j. \end{cases} \tag{37}$$

where η_1 is the cost of observing undesirable states and η_2 denotes the cost of taking intervention over any gene. These constants (η_1 and η_2) should be set at the start of process based on the severity of undesirable conditions and cost of intervention process.

The noise process $\mathbf{n}_k$ in (1) is assumed to have independent components distributed as Bernoulli(p), where parameter $p > 0$ corresponds to the amount of "perturbation" to the Boolean state process – the closer it is to $p = 0.5$, the more chaotic the system will be, while a value of p close to zero means that the state trajectories are nearly deterministic, being governed tightly by the network function.

In all the numerical experiments, we assume the same fixed set of values for the system parameters, summarized in Table II.

TABLE II: Parameter values used in all experiments.

Parameter	**Value**
Number of genes d	7
Transition noise intensity p	0.05
Initial belief $\mathbf{b}_0(i)$, $i = 1, \ldots, 128$	1/128
Mean read counts μ_j^0, μ_j^1, $j = 1, \ldots, 7$	30, 60
Standard deviation $\sigma_j^0 = \sigma_j^1$, $j = 1, \ldots, 7$	10, 15
Discount factor γ	0.95
Control genes	RET1, HADHB
Constants in Cost Function (37)	$\eta_1 = 5$, $\eta_2 = 1$
Number of backup samples N_s	1000
Number of expansion samples N_e	1000
Threshold ϵ in Algorithm 3 and 4	0.05
PBVI belief size N^{PBVI}	32, 128, 512, 1024, 2048
Perseus belief size N^{Pers}	500, 1000, 5000, 10000, 50000
Value Iteration Threshold β [17]	10^{-8}

We compare the proposed methodology against the V_BKF [17] and Q_MDP [19] algorithms. Briefly, let $\boldsymbol{J}_{\mathrm{BDS}} \in \mathbb{R}^{2^d}$ and $\mu_{\mathrm{BDS}}(\mathbf{x}^i) \in \mathbb{U}$, for $i = 1, ..., 2^d$, be the optimal infinite-horizon cost and control policy corresponding to the directly-observable Boolean dynamical system. The V_BKF algorithm is a simple sate-feedback controller that uses the BKF as state observer. Thus, for a given belief state $\mathbf{b}_k$ at time step k, the control input $\mathbf{u}_k$ by V_BKF is chosen as:

$$\mathbf{u}_k^{\text{V\_BKF}} = \mu_{\mathrm{BDS}}(\overline{A\,\mathbf{b}_k})\,, \tag{38}$$

where $\hat{\mathbf{X}}_{k|k}^{\mathrm{MS}} = \overline{A\,\mathbf{b}_k}$ is the optimal MMSE estimate of state.

As for the Q_MDP method, define the $\boldsymbol{\alpha}$-vectors:

$$\boldsymbol{\alpha}_{\mathbf{u}}^{\text{Q\_MDP}} = \mathbf{g}(\mathbf{u})^T + \gamma\,(\boldsymbol{J}_{\mathrm{BDS}})^T\, M(\mathbf{u})\,. \tag{39}$$

for $\mathbf{u} \in \mathbb{U}$. For a given belief state $\mathbf{b}_k$ at time k, Q_MDP selects the control input $\mathbf{u}_k$ as:

$$\mathbf{u}_k^{\text{Q\_MDP}} = \underset{\mathbf{u}\in\mathbb{U}}{\operatorname{argmin}} \boldsymbol{\alpha}_{\mathbf{u}}^{\text{Q\_MDP}} \mathbf{b}_k \,. \tag{40}$$

One can see Q_MDP the same as a point-based controller with only $|\mathbb{U}|$ fixed $\boldsymbol{\alpha}$-vectors in Λ, such that the number and values of these vectors do not depend on measurement model.

The experiment is performed for both RET1 and HADHB as control genes. We recorded the average cost per step and rate of correct state estimation over 1000 time steps, repeated 50 times for randomly selected initial belief states. We also recorded the average running time in seconds, based on a PC with an Intel Core i7-4790 CPU@3.60 GHz clock and 16 GB of RAM. The results are displayed in Table III.

We can observe that the proposed Perseus-POBDS and PBVI-POBDS both display a smaller cost per step than the Q_MDP and V_BKF methods, i.e., they are more effective in keeping the system away from bad states where WNT5A is activated, especially in the presence of high measurement noise. The reason of course is that the underlying Boolean dynamical system is less identifiable in the presence of noisy measurements, and therefore, the policies obtained by Q_MDP and V_BKF, which are both based on the results for the underlying Boolean dynamical system, become less valid. Naturally, the superior performance of the Perseus-POBDS and PBVI-POBDS controllers come at a much higher computation time (which is mostly due to the offline controller step, while online execution times are comparable).

The RET1 gene seems to be a better control input for reducing the activation of WNT5A, as lower cost can be seen under the RET1 control gene in comparison to the HADHB gene in all cases.

We can see that Perseus-POBDS outperforms PBVI-POBDS, by a small margin. This is related to the larger belief set considered by the Perseus offline controller. In fact, the smaller number of $\boldsymbol{\alpha}$-vectors kept in Λ by Perseus speeds up the computation of the backup process in Algorithm 1), allowing the use of a larger belief set, and as a result more coverage over the belief space. This fact can also be appreciated by considering the similar running times achieved by both methods, despite the fact that Perseus uses 50,000 belief points compared to 2048 for PBVI.

Finally, the rate of correct state estimation by the BKF is better for smaller measurement noise, as expected. Fig. 3 displays sample trajectories of the WNT5A gene under control of the HADHB gene and without control, for the less noisy data. The vertical dashed lines show the time points at which the value of the HADHB control gene is flipped. Estimation is accurate in both cases. Note that more frequent activation of WNT5A, i.e., a larger rate of undesirable states, can be seen in the case of the system without control.

To further investigate the performance of our proposed algorithms, we plotted in Fig. 4 the average cost per step as a function of belief set size, for high measurement noise ($\sigma_j^0 = \sigma_j^1 = 15$). The red and blue horizontal lines correspond to the average cost obtained by the Q_MDP and V_BKF methods respectively, whereas the green line represents the

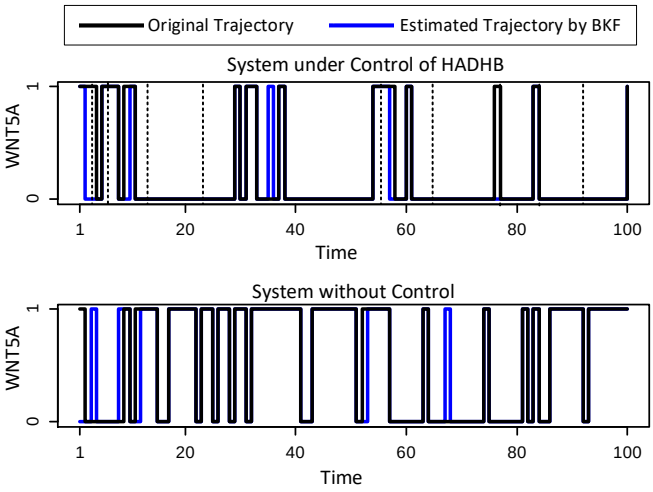

Fig. 3: Original (black lines) and estimated trajectory (blue lines) of WNT5A under control of HADHB gene and system without control. The vertical dashed lines specify the time points at which the value of the HADHB control gene is flipped.

lower bound in cost corresponding to control of the system with directly observed states. Clearly, the performance of the proposed algorithms improves and gets asymptotically close to the lower bound as the size of the belief set grows.

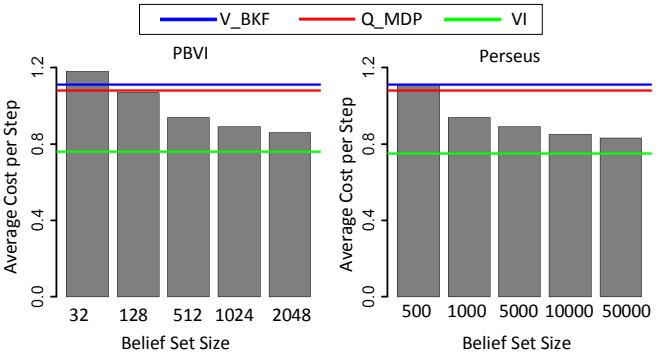

Fig. 4: Average cost per step as a function of the belief set size. Blue and red lines correspond to Q_MDP and V_BKF methods, respectively, while the green line refers to control of the system with directly observed states.

Next, we compared a naive method for partitioning the observation space to the proposed sampling-based partitioning scheme discussed in Section IV-A. Since the variance of activated and inactivated states are assumed to be the same ($\sigma_j^0 = \sigma_j^1$, $j = 1, \ldots, 7$), a naive discretization of observation space, which is obtained by finding the intersection of two distributions $P(\mathbf{y}(j) \mid \mathbf{x}(j) = 0)$ and $P(\mathbf{y}(j) \mid \mathbf{x}(j) = 1)$, is given by:

$$\mathbf{y}^{\text{naive}}(j) = \frac{\mu_j^0 + \mu_j^1}{2} \,, \tag{41}$$

for $j = 1, \ldots, d$. Using this naive partitioning process, the space of each observation variable is divided into two parts

TABLE III: Average results for different methods.

		RET1					HADHB				
$\sigma_j^0 = \sigma_j^1$	**Method**	**Cost/Step**	**Time**	$\|\mathbf{B}\|$	$\|\mathbf{\Lambda}\|$	**State Rate**	**Cost/Step**	**Time**	$\|\mathbf{B}\|$	$\|\mathbf{\Lambda}\|$	**State Rate**
15	Perseus-POBDS	0.83	7185.52	50000	103	0.56	0.95	7124.90	50000	148	0.56
	PBVI-POBDS	0.86	7233.26	2048	1223	0.56	0.99	7382.93	2048	1523	0.55
	Q_MDP	1.08	2.01	-	2	0.54	1.39	2.02	-	2	0.56
	V_BKF	1.11	2.04	-	-	0.56	1.46	2.04	-	-	0.55
10	Perseus-POBDS	0.81	7120.29	50000	94	0.92	0.92	7193.07	50000	131	0.92
	PBVI-POBDS	0.81	7395.63	2048	1053	0.92	0.93	7236.29	2048	1472	0.91
	Q_MDP	0.82	2.00	-	2	0.92	0.96	2.00	-	2	0.92
	V_BKF	0.83	2.01	-	-	0.92	0.97	2.01	-	-	0.91

and the observation space will be mapped into a size-2^d finite observation set, which allows the application of the original Perseus or PBVI methods developed for finite measurement spaces.

For this comparison, RET1 is used as a control gene and the Perseus offline controller with 10,000 belief states is employed. Figs. 6 and 5 display normalized histogram plots of observed states using 50 trajectories with 10,000 time steps each, comparing the naive and proposed partitioning methods, respectively. The histograms over desirable and undesirable states are shown by blue and red colors respectively (recalling that the undesirable states are those where WNT5A is activated). For comparison, Fig. 7 displays the same plot for the case where the system runs without any intervention.

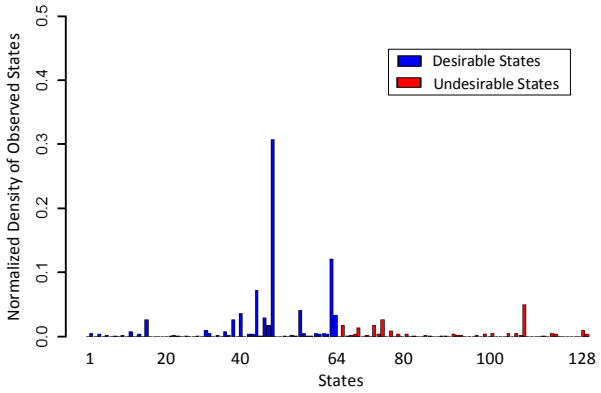

Fig. 5: Normalized histogram plots of observed states for Perseus controller with naive partitioning of the observation space.

It is clear that the system under control of the proposed partitioning method visited undesirable states less often than the naive partitioning case. Both approaches naturally perform better than the case with no control, which displays almost equal frequencies of visits to desirable and undesirable states.

VIII. Conclusion

In this paper a methodology for simultaneous monitoring and control of partially-observed gene regulatory networks

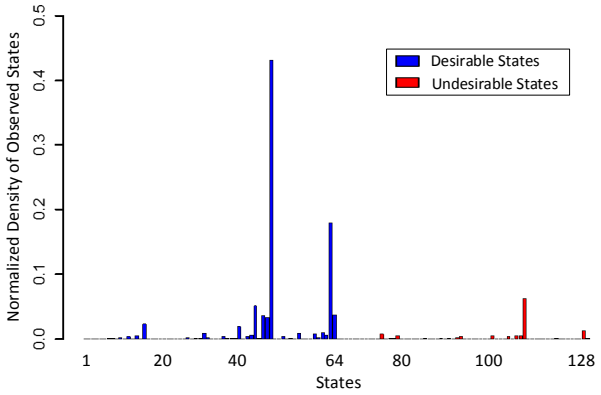

Fig. 6: Normalized histogram plots of observed states for Perseus controller with the proposed sampling-based dynamical partitioning of the observation space.

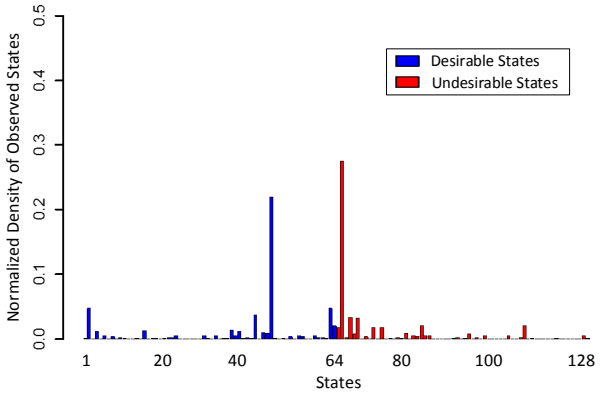

Fig. 7: Normalized histogram plots of observed states with no control.

was presented. The POBDS signal model was used for modeling gene regulatory networks. We introduced backup and expansion procedures to deal with an infinite observation space. Point-based controllers were developed by employing the proposed backup and expansion procedures in combination with offline controllers for finite observation spaces. While

any point-based method can be adopted for offline control, we used the PBVI and Perseus methods, which led to the PBVI-POBDS and Perseus-POBDS control algorithms.

To improve the policy obtained by the offline controller, an online controller was developed based on a one-step look-ahead search and the Boolean Kalman Filter during execution step. The ability of the proposed methodology to obtain good control policies were demonstrated by numerical experiments involving a Boolean model of a melanoma gene regulatory network observed through noisy gene expression data.

Future work will consider developing effective and robust adaptive controllers for large POBDS. In addition, the signal model used in this paper assumes the synchronous update of the network states; future work will consider the extension of the methodology to asyncronous networks, which can be accomplished by adding latent timing variables to the state vector.

APPENDIX

A. Proof of Lemma 1

Assume that the maximum error of having belief set B_m instead of the whole continuous belief space is caused by $\mathbf{b}' \in \mathbb{B}$, and let $\mathbf{b}$ be the closest L_1-norm belief point in B_m to $\mathbf{b}'$. Furthermore, assume that $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}'$ are the vectors which are minimal at $\mathbf{b}$ and $\mathbf{b}'$ respectively. The maximum possible error caused by having B_m is $\boldsymbol{\alpha}\mathbf{b}' - \boldsymbol{\alpha}'\mathbf{b}'$. Thus, based on the fact that at belief point $\mathbf{b}$, $\boldsymbol{\alpha}\mathbf{b} \leq \boldsymbol{\alpha}'\mathbf{b}$, one can obtain this error as:

$$\begin{aligned} \|\tilde{T}[J_{m-1}] - T[J_{m-1}]\|_\infty &\leq \boldsymbol{\alpha}\mathbf{b}' - \boldsymbol{\alpha}'\mathbf{b}' \\ &= \boldsymbol{\alpha}\mathbf{b}' - \boldsymbol{\alpha}'\mathbf{b}' + (\boldsymbol{\alpha}'\mathbf{b} - \boldsymbol{\alpha}'\mathbf{b}) \\ &\leq \boldsymbol{\alpha}\mathbf{b}' - \boldsymbol{\alpha}'\mathbf{b}' + \boldsymbol{\alpha}'\mathbf{b} - \boldsymbol{\alpha}\mathbf{b} \\ &= (\boldsymbol{\alpha}' - \boldsymbol{\alpha})(\mathbf{b} - \mathbf{b}'), \end{aligned} \tag{42}$$

By Holder's inequality, we have $(\boldsymbol{\alpha}' - \boldsymbol{\alpha})(\mathbf{b} - \mathbf{b}') \leq \|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_\infty \|\mathbf{b} - \mathbf{b}'\|_1$. To be able to find the upper bound for $\|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_\infty$, one can find the maximum possible distance in L_1 norm between $\boldsymbol{\alpha}$-vectors for large enough m as:

$$\begin{aligned} \boldsymbol{\alpha}' &= \max_{\mathbf{u}\in\mathbb{U}, i\in\{1,\ldots,2^d\}} c(\mathbf{x}^i, \mathbf{u}) + \gamma\,\boldsymbol{\alpha}', \\ \boldsymbol{\alpha} &= \min_{\mathbf{u}\in\mathbb{U}, i\in\{1,\ldots,2^d\}} c(\mathbf{x}^i, \mathbf{u}) + \gamma\,\boldsymbol{\alpha}, \end{aligned} \tag{43}$$

By solving the above equations for $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}'$ and substituting $g_{\max} = \max_{\mathbf{u}\in\mathbb{U}, i\in\{1,\ldots,2^d\}} c(\mathbf{x}^i, \mathbf{u})$ and $g_{\min} = \min_{\mathbf{u}\in\mathbb{U}, i\in\{1,\ldots,2^d\}} c(\mathbf{x}^i, \mathbf{u})$, the maximum bound for $\|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_\infty$ can be obtained as:

$$\|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_\infty \leq \frac{g_{\max} - g_{\min}}{1-\gamma}, \tag{44}$$

Furthermore, the maximum L_1 norm of the $\|\mathbf{b} - \mathbf{b}'\|_1$ is ϵ_{B_m} define in equation (28). Now, replacing equation (28) and (44) into equation (42), the upper bound of having finite belief set (B_m) can be obtained as:

$$\begin{aligned} \|\tilde{T}[J_{m-1}] - T[J_{m-1}]\|_\infty &\leq \|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_\infty \|\mathbf{b} - \mathbf{b}'\|_1 \\ &\leq \frac{(g_{\max} - g_{\min})}{1-\gamma}\,\epsilon_{B_m}. \end{aligned} \tag{45}$$

B. Proof of Theorem 1

Using the latest set of $\boldsymbol{\alpha}$-vectors computed by Algorithm 3 at time $m-1$, one can write $J_m = \tilde{T}[J_{m-1}]$. Furthermore, $J_m^* = T[J_{m-1}^*]$ where T is the optimal transformation. For a large enough m, we have $T[J_m^*] = J_m^* = J^*$, then using triangle inequality we have:

$$\|J_m - J^*\|_\infty \leq \|J_m - J_m^*\|_\infty + \|J_m - J^*\|_\infty. \tag{46}$$

By further simplification of the first term, we have:

$$\begin{aligned} \|J_m - J_m^*\|_\infty &= \|\tilde{T}[J_{m-1}] - T[J_{m-1}^*]\|_\infty \\ &\leq \|\tilde{T}[J_{m-1}] - T[J_{m-1}]\|_\infty \\ &\quad + \|T[J_{m-1}] - T[J_{m-1}^*]\|_\infty, \end{aligned} \tag{47}$$

where the last expression is obtained by use of triangle inequality. For the second part of last expression in equation (47), since the operator T is the optimal operator, it is guaranteed that:

$$\|T[J_{m-1}] - T[J_{m-1}^*]\|_\infty \leq \gamma\,\|J_{m-1} - J_{m-1}^*\|_\infty, \tag{48}$$

Lemma 1 specifies the upper bound for the first term in right hand side of equation (47). Thus:

$$\begin{aligned} &\|J_m - J_m^*\|_\infty \\ &\leq \|\tilde{T}[J_{m-1}] - T[J_{m-1}]\|_\infty + \gamma\,\|J_{m-1} - J_{m-1}^*\|_\infty \\ &\leq \frac{(g_{\max} - g_{\min})}{1-\gamma}\,\epsilon_{B_m} + \gamma\,\|J_{m-1} - J_{m-1}^*\|_\infty \\ &\leq \frac{(g_{\max} - g_{\min})}{(1-\gamma)^2}\,\epsilon_{B_m}. \end{aligned} \tag{49}$$

The last term can be obtained easily by the using the geometric sum. Equation (49) completes the first part of the proof of Theorem 2. Replacing the first term in the right hand side of equation (46) by the bound obtained in equation (49), and also the fact that $||J_m - J^*||_\infty \leq \gamma^m\,||J_0 - J^*||_\infty$, completes the proof of the theorem.

C. Proof of Theorem 2

Let T be the optimal mapping, and T^{LA} be a mapping corresponding to the look-ahead policy for J (cost obtained by the offline controller). Since the look-ahead policy optimizes its control policy with regard to J^*, we have: $T[J^*] = T^{\mathrm{LA}}[J^*]$. Now, using the triangular inequality we have:

$$\begin{aligned} ||J^{\mathrm{LA}} - J^*||_\infty &= ||T^{\mathrm{LA}}[J] - J^*||_\infty \\ &\leq ||T^{\mathrm{LA}}[J] - T[J^*]||_\infty \\ &\quad + ||T[J^*] - J^*||_\infty \\ &= ||T^{\mathrm{LA}}[J] - T^{\mathrm{LA}}[J^*]||_\infty \\ &\leq \gamma\,||J - J^*||_\infty = \gamma\epsilon. \end{aligned} \tag{50}$$

ACKNOWLEDGMENT

The authors acknowledge the support of the National Science Foundation, through NSF awards CCF-1320884 and CCF-1718924.

REFERENCES

[1] Guy Karlebach and Ron Shamir. Modelling and analysis of gene regulatory networks. *Nature Reviews Molecular Cell Biology*, 9(10):770–780, 2008.

[2] Ilya Shmulevich and Edward R Dougherty. *Genomic signal processing*. Princeton University Press, 2014.

[3] Stuart A Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437–467, 1969.

[4] Ilya Shmulevich, Edward R Dougherty, and Wei Zhang. From Boolean to probabilistic Boolean networks as models of genetic regulatory networks. *Proceedings of the IEEE*, 90(11):1778–1792, 2002.

[5] Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using Bayesian networks to analyze expression data. *Journal of computational biology*, 7(3-4):601–620, 2000.

[6] Daizhan Cheng and Hongsheng Qi. Controllability and observability of Boolean control networks. *Automatica*, 45(7):1659–1667, 2009.

[7] Aniruddha Datta, Ashish Choudhary, Michael L Bittner, and Edward R Dougherty. External control in Markovian genetic regulatory networks. *Machine learning*, 52(1-2):169–191, 2003.

[8] Ranadip Pal, Aniruddha Datta, and Edward R Dougherty. Optimal infinite-horizon control for probabilistic Boolean networks. *Signal Processing, IEEE Transactions on*, 54(6):2375–2387, 2006.

[9] Daizhan Cheng and Hongsheng Qi. A linear representation of dynamics of Boolean networks. *IEEE Transactions on Automatic Control*, 55(10):2251–2258, 2010.

[10] Yidong Chen, Edward R Dougherty, and Michael L Bittner. Ratio-based decisions and the quantitative analysis of cDNA microarray images. *Journal of Biomedical optics*, 2(4):364–374, 1997.

[11] Ali Mortazavi, Brian A Williams, Kenneth McCue, Lorian Schaeffer, and Barbara Wold. Mapping and quantifying mammalian transcriptomes by RNA-seq. *Nature methods*, 5(7):621–628, 2008.

[12] Jianping Hua, Chao Sima, Milana Cypert, Gerald C Gooden, Sonsoles Shack, Lalitamba Alla, Edward A Smith, Jeffrey M Trent, Edward R Dougherty, and Michael L Bittner. Dynamical analysis of drug efficacy and mechanism of action using GFP reporters. *Journal of Biological Systems*, 20(04):403–422, 2012.

[13] Mahdi Imani and Ulisses Braga-Neto. Maximum-likelihood adaptive filter for partially-observed Boolean dynamical systems. *IEEE transaction on Signal Processing*, 65:359–371, 2017.

[14] Mahdi Imani and Ulisses Braga-Neto. Particle filters for partially-observed Boolean dynamical systems. *Automatica*, 2017.

[15] Ulisses Braga-Neto. Optimal state estimation for Boolean dynamical systems. In *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*, pages 1050–1054. IEEE, 2011.

[16] Mahdi Imani and Ulisses Braga-Neto. Optimal state estimation for Boolean dynamical systems using a Boolean kalman smoother. In *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 972–976. IEEE, 2015.

[17] Mahdi Imani and Ulisses Braga-Neto. State-feedback control of partially-observed Boolean dynamical systems using RNA-seq time series data. In *American Control Conference (ACC), 2016*, pages 227–232. IEEE, 2016.

[18] M. Imani and U.M. Braga-Neto. Multiple model adaptive controller for partially-observed Boolean dynamical systems. In *Proceedings of the 2017 American Control Conference (ACC 2017), Seattle, WA*, 2017.

[19] Michael L. Littman, Anthony R. Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning (ICML)*. Morgan Kaufmann, 1995.

[20] Matthijs TJ Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of artificial intelligence research*, pages 195–220, 2005.

[21] Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, pages 335–380, 2006.

[22] Stéphane Ross, Brahim Chaib-Draa, et al. Aems: An anytime online search algorithm for approximate policy refinement in large POMDPs. In *IJCAI*, pages 2592–2598, 2007.

[23] Joelle Pineau, Geoff Gordon, Sebastian Thrun, et al. Point-based value iteration: An anytime algorithm for POMDPs. In *IJCAI*, volume 3, pages 1025–1032, 2003.

[24] Karl Johan Åström and PR Kumar. Control: A perspective. *Automatica*, 50(1):3–43, 2014.

[25] Trey Smith and Reid Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 520–527. AUAI Press, 2004.

[26] Trey Smith and Reid Simmons. Point-based POMDP algorithms: Improved analysis and implementation. *arXiv preprint arXiv:1207.1412*, 2012.

[27] Josep M Porta, Nikos Vlassis, Matthijs TJ Spaan, and Pascal Poupart. Point-based value iteration for continuous POMDPs. *The Journal of Machine Learning Research*, 7:2329–2367, 2006.

[28] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[29] Mahdi Imani and Ulisses Braga-Neto. Point-based value iteration for partially-observed Boolean dynamical systems with finite observation space. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 4208–4213. IEEE, 2016.

[30] Sandrine Dudoit, Yee Hwa Yang, Matthew J Callow, and Terence P Speed. Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica sinica*, pages 111–139, 2002.

[31] Ranadip Pal, Aniruddha Datta, and Edward R Dougherty. Bayesian robustness in the control of gene regulatory networks. *IEEE Transactions on Signal Processing*, 57(9):3667–3678, 2009.

[32] Diane Longo and Jeff Hasty. Dynamics of single-cell gene expression. *Molecular systems biology*, 2(1):64, 2006.

[33] R.A. Weinberg. *The Biology of Cancer*. Princeton: Garland Science, 2006.

[34] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995.

[35] Edward J Sondik. The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. *Operations Research*, 26(2):282–304, 1978.

[36] Anthony Cassandra, Michael L Littman, and Nevin L Zhang. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In *Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence*, pages 54–61. Morgan Kaufmann Publishers Inc., 1997.

[37] Christian P Robert. *Monte carlo methods*. Wiley Online Library, 2004.

[38] Edward R Dougherty, Ranadip Pal, Xiaoning Qian, Michael L Bittner, and Aniruddha Datta. Stationary and structural control in gene regulatory networks: basic concepts. *International Journal of Systems Science*, 41(1):5–16, 2010.

[39] Meltzer Bittner, P Meltzer, Y Chen, Y Jiang, E Seftor, M Hendrix, M Radmacher, Rm Simon, Z Yakhini, A Ben-Dor, et al. Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature*, 406(6795):536–540, 2000.

[40] Ashani T Weeraratna, Yuan Jiang, Galen Hostetter, Kevin Rosenblatt, Paul Duray, Michael Bittner, and Jeffrey M Trent. Wnt5a signaling directly affects cell motility and invasion of metastatic melanoma. *Cancer cell*, 1(3):279–288, 2002.

Mahdi Imani received his B.Sc. degree in Mechanical Engineering and his M.Sc. degree in Electrical Engineering, both from University of Tehran in 2012 and 2014. He is currently a Ph.D. student at the Department of Electrical and Computer Engineering of Texas A&M University, College Station, TX. His current research interests include estimation and control of stochastic Boolean dynamical systems, with applications in genomic signal processing.

Ulisses M. Braga-Neto is an Associate Professor at the Department of Electrical and Computer Engineering and a member of the Center for Bioinformatics and Genomic Systems Engineering at Texas A&M University, College Station, TX. He holds a Ph.D. degree in Electrical and Computer Engineering from The Johns Hopkins University, Baltimore, MD and has held post-doctoral positions at the University of Texas M.D. Anderson Cancer Center, Houston, TX and at the Oswaldo Cruz Foundation, Recife, Brazil. His research interests include Pattern Recognition and Statistical Signal Processing. Dr. Braga-Neto is a Senior Member of the IEEE. He is author of the textbook Error Estimation for Pattern Recognition (IEEE-Wiley, 2015) and has received the NSF CAREER Award for his work in this area.