52nd IEEE Conference on Decision and Control
December 10-13, 2013. Florence, Italy

Data-driven Verification of Synthetic Gene Networks

Ebru Aydin Gol, Douglas Densmore, and Calin Belta

Abstract— **Automatic design of synthetic gene networks with specific functions is an emerging field in synthetic biology. Quantitative evaluation of gene network designs is a missing feature of the existing automatic design tools. In this work, we address this issue and present a framework to probabilistically analyze the dynamic behavior of a gene network against specifications given in a rich and high level language. Given a gene network built from primitive DNA parts, and given experimental data for the parts, the tool proposed here allows for the automatic construction of a stochastic model of the gene network and *in silico* probabilistic verification against a rich specification.**

I. INTRODUCTION

Synthetic biology's mandate is to forward engineer living systems using engineering principles. DNA parts are encapsulated, composed to realize novel functionality, and introduced into "host organisms" that carry out that functionality. These techniques have been recently used for a wide variety of applications [1]–[4]. Such engineered organisms can be coupled with mechanical and electronic systems to create a whole new class of cyber-physical systems not yet explored [5]. However, these systems will be extremely complex and current experimental methods are *ad-hoc* at best. Bio-design automation frameworks are emerging [6]–[9], which generate synthetic gene networks from specifications, assign these networks discrete DNA segments, and physically assemble these circuits. What is notably absent from these workflows is a *verification stage* which quantitatively evaluates these designs against their specification.

In [10], we used Linear Temporal Logic (LTL) as a specification language and discrete-time piecewise affine (PWA) systems with polyhedral parameter uncertainty as mathematical models for synthetic gene networks. We showed that such models can be derived from experimental data and checked against arbitrary LTL formulas by constructing finite abstractions. Due to the wide uncertainty ranges from the experimental data and the conservativeness of the approach, the results were inconclusive. In this paper, we propose to capture the distribution of the experimental data into stochastic discrete-time PWA models and to use probabilistic verification techniques to analyze the behavior of the system. We find that this approach is much more conclusive and reflects more closely the biology it is modeling.

The probabilistic verification of finite state stochastic systems, such as Markov chains, is a well understood problem [11]. There are efficient tools, such as PRISM [12], that model check the system against a probabilistic temporal logic property. However, such tools cannot deal with stochastic systems with infinite state spaces. In [13], the authors proposed to use partitions of the state space to produce abstractions in the form of Markov chains, which can then be model checked with an explicit error bound on the probability of satisfaction. However, due to the dependence of the error bound on the partition size, the computation of the abstraction is infeasible for high-dimensional systems.

An alternative approach to the probabilistic verification problem of systems with large or infinite state spaces is statistical model checking (SMC) [14], [15], which applies statistical inference techniques to solve the verification problem. Since SMC relies on the model checking results of sample system traces, the outcome is probabilistic in nature, *i.e.* it is correct with a certain probability. The key advantage of this technique is that it can handle complex and high-dimensional systems with infinite state spaces in an efficient way, since the computation necessary for trace generation and model checking can be parallelized.

In this work, SMC is used to verify the dynamic behavior of a synthetic gene network assuming that the gene network is built from primitive DNA parts for which experimental data exists. The composite behavior of these parts is captured in a stochastic dynamical system whose parameters are obtained from experimental data. We use this model to solve two problems. The first is a verification problem against a specification expressed as a probabilistic bounded LTL (PBLTL) formula, which is used to check the correctness of the design. The second is a parameter optimization problem, in which we use SMC to find time bounds and species concentration threshold values that make a formula satisfiable with a given probability. This problem allows us to tune a design parameter to improve the performance. The optimized parameters can be further used to compare gene networks designed to satisfy the same specification.

II. PROBLEM FORMULATION

A. Gene Network

A synthetic gene network (circuit) is composed of two basic biological parts (encapsulated DNA sequences): promoters and genes. A *gene* g codes for a certain protein that degrades at a rate α_g. The concentration of the protein is denoted by x_g and it is assumed to be bounded in a relevant range $x_g^{min} \leq x_g \leq x_g^{max}$. The rate of expression of a gene is regulated by a *promoter* that precedes the gene (to the left when DNA is depicted visually) in the DNA. Regulators bind to the promoters and define regulations motifs. A *regulator*, which can be either a protein coded by a gene in the

This work was supported at Boston University by the ONR under grant MURI 014-001-0303-5.

Ebru Aydin Gol (ebru@bu.edu), Douglas Densmore (dougd@bu.edu), and Calin Belta (cbelta@bu.edu) are with Boston University.

978-1-4673-5717-3/13/$31.00 ©2013 IEEE

network or a small molecule (*external regulator*), can enable (activator) or disable (repressor) the ability of a promoter to initiate transcription (production of mRNA). The mRNA is translated into protein by the ribosome that recognizes and binds to the Ribosome Binding Site (RBS) of the mRNA. A promoter p can be regulated by multiple regulators, and the rate of expression β_p of a gene g that proceeds the promoter p depends on the concentrations of the regulators of the promoter p. In our simplified description of gene regulation, the rate of expression captures both transcription and translation.

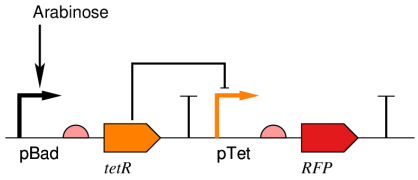


Fig. 1. A synthetic gene network. The promoters (pBad and pTet) are indicated by bent arrows. The genes ($tetR$ and RFP) are shown as colored polygons. The regulators (arabinose and $tetR$) are connected to the corresponding promoters: arrows show activations and stub connectors show repressions. The RBSs are shown with half circles.

Example 2.1: A synthetic gene network composed of 2 promoters and 2 genes is shown in Figure 1. The $pBad$ promoter, which is activated by the small-molecule species $Arabinose$, regulates the $tetR$ gene. $tetR$ represses the second promoter $pTet$, which regulates the RFP gene. [1] This gene network is expected to work as an inverter gate, where the concentration of the external regulator arabinose, x_{Ara}, is treated as the input and x_{RFP} is treated as the output. A high arabinose concentration, $x_{Ara} > T^H_{Ara}$, activates the $pBad$ promoter, and hence increases the production rate of $tetR$. As $tetR$ represses the $pTet$ promoter, high arabinose concentration eventually decreases x_{RFP}, $x_{RFP} < T^L_{RFP}$. In the absence of arabinose, $x_{Ara} < T^L_{Ara}$, $pBad$ is not activated, the $tetR$ gene is not transcribed and $pTet$ is not repressed. Consequently, the low arabinose concentration eventually increases x_{RFP}, $x_{RFP} > T^H_{RFP}$. This qualitative behavior will be "quantified" later in the paper. In the above, T^H_{Ara}, T^L_{RFP}, T^L_{Ara}, T^H_{RFP} are species concentration thresholds that characterize the circuit.

B. Specification

The evolution of a gene network in time is defined as a sequence of protein concentrations, called a *trajectory*, *e.g.* a trajectory of the gene network from Example 2.1 is given by

$$\{(x_{tetR}(k), x_{RFP}(k))\}_{k\in\mathbb{Z}_+}. \quad (1)$$

In this work, we use Bounded Linear Temporal Logic (BLTL) formulas over linear inequalities over the concentrations of proteins and external regulators to specify the dynamic behavior of a gene network. A detailed description of the syntax and semantics of BLTL is beyond the scope of this paper and can be found in [15]. Informally, a BLTL formula over a set of linear inequalities is inductively defined by using Boolean operators $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction), $\Rightarrow$ (implication) and a temporal operator U^k (until) with bound k. Additional bounded temporal operators can be defined, *e.g.* $\mathsf{F}^k\Phi = \mathbb{T}\,\mathsf{U}^k\Phi$ (eventually), or $\mathsf{G}^k\Phi = \neg\mathsf{F}^k\neg\Phi$ (always), where $\mathbb{T}$ stands for Boolean constant true.

We interpret BLTL formulas over trajectories of gene networks as defined above. For example, consider a simple formula

$$\Phi_S = (x_{tetR} < T_{tetR})\,\mathsf{U}^{k_1}(x_{RFP} > T_{RFP}),$$

where $T_{tetR}, T_{RFP} \in \mathbb{R}_+$. A trajectory σ of the form given in Equation (1) satisfies formula Φ_S, written as $\sigma \models \Phi_S$, if there exists $k \le k_1$, such that $x_{tetR}(i) < T_{tetR}$ for all $i = 0, \ldots, k-1$ and $x_{RFP}(k) > T_{RFP}$.

The inverter gate specification from Example 2.1 can be formally stated as the following BLTL formula:

$$\Phi_I = \big((x_{Ara} < T^L_{Ara}) \Rightarrow (F^{k_1}G^{k_2}(x_{RFP} > T^H_{RFP}))\big) \wedge \big((x_{Ara} > T^H_{Ara}) \Rightarrow (F^{k_1}G^{k_2}(x_{RFP} < T^L_{RFP}))\big). \quad (2)$$

Formula Φ_I requires the circuit to respond in k_1 time steps to the input and the output is interpreted as high (low) only when $x_{RFP} > T^H_{RFP}$ ($x_{RFP} < T^L_{RFP}$) is satisfied for k_2 consecutive time steps. Hence, k_1 can be considered as the response time and k_2 can be considered as the signaling time. For example, if $x_{Ara} < T^L_{Ara}$, a trajectory σ satisfies formula Φ_I if there exists $k < k_1$ such that $x_{RFP}(k+i) > T^H_{RFP}$, for all $i = 1, \ldots, k_2$. If all the bounds that appear in a formula Φ have finite values, then $\sigma \models \Phi$ can be decided based on a finite prefix of the trajectory σ [15].

As explained above, a trajectory of a gene network can be checked against a BLTL formula. A gene network is inherently a stochastic system due to complex biochemistry involved in the protein production, *e.g.* a regulator binds to a promoter with a certain probability. For this reason, we use Probabilistic BLTL (PBLTL) to specify the behavior of a gene network.

A PBLTL formula is a formula of the form $P_{\ge\theta}(\Phi)$, where Φ is a BLTL formula and $\theta \in [0,1]$ is a probability. A gene network satisfies PBLTL formula $P_{\ge\theta}(\Phi)$ if and only if the probability that a trajectory of the gene network satisfies BLTL formula Φ is greater than or equal to θ.

C. Verification and Parameter Optimization

Problem 2.1 (Verification): Assume we have a gene network and a specification expressed as a BLTL formula Φ over linear inequalities over the concentrations of the proteins and the external regulators.

(i) Assume a probability $\theta \in [0,1]$ is given. Decide whether the network satisfies the PBLTL formula $P_{\ge\theta}(\Phi)$.
(ii) Compute the probability θ with which the gene network satisfies the BLTL formula Φ.

In the second problem, our goal is to optimize a parameter that appears in the specification formula, *i.e.* either a threshold used in an inequality or a time bound of a temporal logic operator, while the rest of the parameters are assumed fixed.

[1] To keep the notation to a minimum, we use the same names for a gene and for the protein expressed from that gene.

Problem 2.2 (Parameter Optimization): Assume we have a gene network, a probability $\theta \in [0,1]$, and a specification expressed as a BLTL formula Φ in which the thresholds and the time bounds are fixed except one that is denoted as $\mathcal{T}$. Find the minimum (or maximum) value of $\mathcal{T}$ such that the gene network satisfies the PBLTL formula $P_{\geq\theta}(\Phi)$.

To illustrate the usefulness of Problem 2.2, consider formula Φ_I from Equation (2). Since T^H_{RFP} is used as a lower bound for x_{RFP}, increasing T^H_{RFP} decreases the probability that a trajectory of the gene network satisfies Φ_I. On the other hand, the gene network works as an inverter if it satisfies $P_{\geq\theta}(\Phi_I)$ for a high probability θ when $T^H_{RFP} > T^L_{RFP}$ and $T^{\bar{H}}_{Ara} > T^L_{Ara}$. We can use the solution of Problem 2.2 to find the maximum value for T^H_{RFP} for a given probability of satisfaction. Similarly, we can find the minimum value for T^L_{RFP} for a given probability of satisfaction. Solving these types of problems allows us to optimize the "qualitative" behavior of the circuit. Moreover, different network designs can be compared with respect to the optimized thresholds.

To provide solutions to the above problems, we assume that the degradation rates of all the proteins are (statistically) known (see Section III) and characterization data for all the promoters is available (more information on this type of data is given in Section III-A). We will use the available data to model the gene network as a discrete-time stochastic dynamical system. We will employ statistical model checking techniques to solve the problems presented above. Specifically, we will simulate the model, model check the produced trajectories against the specification formula, and use the sample set of model checking results to solve the problems by using statistical inference.

III. Mathematical Model

A gene network S composed of n genes and s promoters is modeled by

$$x_{g_i}(k+1) = \alpha_{g_i}(k)x_i(k) + \beta_{\gamma(g_i)}(k), i = 1,\ldots,n, \quad (3)$$

where $x_{g_i}(k) \in [x^{min}_{g_i}, x^{max}_{g_i}]$ and $\alpha_{g_i}(k) \in (0,1)$ are the concentration and the degradation rate, respectively, of the protein coded by gene g_i at time $k \in \mathbb{Z}_+$. Function $\gamma : \{g_1,\ldots,g_n\} \to \{p_1,\ldots,p_s\}$ maps a gene to the promoter that regulates it, *i.e.* gene g_i is expressed at rate $\beta_{\gamma(g_i)}(k)$ from the promoter $\gamma(g_i)$ at time $k \in \mathbb{Z}_+$. The expression and the degradation rates are modeled by random variables, whose distributions depend on their value at the previous time step, *i.e.*

$$\alpha_{g_i}(k) \sim P_{\alpha_{g_i}}(\cdot \mid \alpha_{g_i}(k-1)), \qquad i = 1,\ldots,n,$$
$$\beta_{p_j}(k) \sim P_{\beta_{p_j}}(\cdot \mid \boldsymbol{x}_{p_j}(k), \beta_{p_j}(k-1)), \quad j = 1,\ldots,s,$$

where $\boldsymbol{x}_{p_j}(k)$ is a vector containing the concentrations of the regulators of promoter p_j at time $k \in \mathbb{Z}_+$. We use π_0 to denote the distribution of the initial states, and S^{π_0} to denote the system initialized at π_0.

The stochastic model from Equation (3) captures our simplified view of the gene expression mechanism introduced in Section II. It also allows us to capture that the degradation rate of a protein or an expression rate from a promoter can not change drastically in a short time period in a living cell.

In our subsequent analysis, we assume that the distributions of the degradation rates are known and characterization data for each promoter is available. The distributions of the degradation rates are often available in literature [16] or can be obtained computationally [17].

A. Promoter Characterization

Our promoters are characterized by a rate of expression that depends on the corresponding set of regulator concentrations and the probability that the regulators bind to the promoter. The relation between the rate of expression from the promoter and the regulators can be captured from experimental data that simultaneously measures the concentrations of the regulators and the concentration of a protein whose expression is directly regulated by only the promoter. In our experimental set-up, a characterization circuit is constructed for each promoter. The characterization circuit involves the promoter and a gene coding for a fluorescent protein (a reporter protein). Thus, as the promoter is regulated to different levels of transcription, different levels of fluorescence will be observed. The cell culture is allowed to fluoresce, and then the fluorescence level in each cell is measured using a flow cytometer. The flow cytometer excites the fluorescent proteins with laser. The light emitted by the fluorescent proteins is measured and translated to fluorescence units [18]. The cells are assumed to be in steady state when the measurements are taken. From Equation (3), it follows that, for each gene g_i:

$$x_{g_i} = \alpha_{g_i}x_{g_i} + \beta_{\gamma(g_i)}. \quad (4)$$

The characterization circuit for a promoter that is regulated by only an external regulator consists the promoter and a gene coding a fluorescent protein. In the case that the promoter is regulated by a protein, a more complex characterization circuit is required as the concentration of the regulator protein can not be controlled directly. The concentrations of both the regulator protein and a protein expressed by the promoter should be measured. Generally, the regulator protein can not be measured and a fluorescent protein is used as a reporter. Characterizing a promoter with multiple regulators requires combining the techniques explained above.

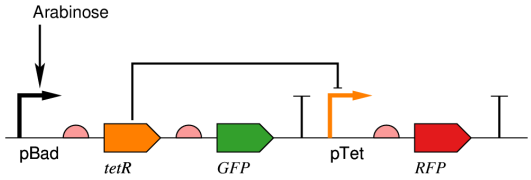


Fig. 2. Characterization circuit for a promoter regulated by a protein.

Example 3.1: A circuit that is composed of the $pBad$ promoter and a gene coding for a fluorescent gene such as GFP can be used to characterize $pBad$ from Example 2.1. Since the $pTet$ promoter is regulated by the $tetR$ protein and

$tetR$ is not fluorescent, the circuit given in Figure 3.1 is build in vivo and used to characterize both of the promoters as follows. A population of cells is partitioned into 7 parts and each part is subjected to a different arabinose concentration, $\{0, 0.5,\ 1, 2.5, 5, 7.5, 10\}$ millimolar (mM) arabinose. Then, the green (GFP) and the red (RFP) fluorescent proteins are measured simultaneously in fluorescent units. The GFP data, x_{GFP}, obtained at different arabinose levels is used to characterize the $pBad$ promoter. The paired data (x_{GFP}, x_{RFP}) for all arabinose levels is used to characterize the $pTet$ promoter, where x_{GFP} is used as the reporter of x_{tetR}.

We compute the distributions of the expression rates from the characterization data of the promoters as follows. Consider promoter p that regulates gene g, and let $\boldsymbol{x}_p = (\boldsymbol{x}_{p,g}, \boldsymbol{x}_{p,e})$ denote the set of concentrations of the corresponding regulators, where $\boldsymbol{x}_{p,g} \in \mathbb{R}_+^{n_g}$ is the vector containing concentrations of the protein regulators, and $\boldsymbol{x}_{p,e} \in \mathbb{R}_+^{n_e}$ is the vector containing concentrations of the external regulators. Even though a promoter usually has either one or two regulators, the general case, $n_g + n_e$ regulators, is considered here. To characterize the promoter, a set of concentration levels $\{x_{p,e,i}^1, \ldots, x_{p,e,i}^{b_i}\}$ is set for each external regulator $x_{p,e,i}$ and an experiment is conducted for each concentration combination $(x_{p,e,1}^{c_1}, \ldots, x_{p,e,n_e}^{c_{n_e}})$, where $1 \leq c_i \leq b_i$ for all $i = 1, \ldots, n_e$. Hence, $\Pi_{i=1}^{n_e} b_i$ experiments are necessary to characterize the promoter. In each of these experiments, the concentrations of both regulator proteins (or a reporter protein), $\boldsymbol{x}_{p,g}$, and the concentration of the protein coded by gene g, x_g, are measured, which results in the following data set:

$$D^{\boldsymbol{x}_{p,e}} = \{(\boldsymbol{x}_{p,g}, x_g)_k\}_{k \in \mathbb{Z}_+}, \qquad \boldsymbol{x}_{p,e} \in \mathbb{X}_{p,e}, \text{where} \tag{5}$$
$$\mathbb{X}_{p,e} = \{(x_{p,e,1}^{c_1}, \ldots, x_{p,e,n_e}^{c_{n_e}}) \mid 1 \leq c_i \leq b_i, \forall i = 1, \ldots, n_e\}.$$

As mentioned before, we assume that the distribution of the degradation rate $P_{\alpha_g}(\cdot)$ is known and it is independent from the regulator concentrations. Therefore, we can compute the distribution of the expression rate conditioned on the regulator concentrations from $P_{\alpha_g}(\cdot)$, (4), and $D^{\boldsymbol{x}_{p,e}}$ as:

$$P_{\beta_p}(\beta | \boldsymbol{x}_{p,g}, \boldsymbol{x}_{p,e}) = \int_{\alpha \in (0,1)} P_{\alpha_g}(\alpha) P_{x_g}(\frac{\beta}{1-\alpha} \mid \boldsymbol{x}_{p,g}, \boldsymbol{x}_{p,e}).$$

The conditional distribution of x_g, $P_{x_g}(\cdot \mid \boldsymbol{x}_{p,g}, \boldsymbol{x}_{p,e})$, can be computed from the data set $D^{\boldsymbol{x}_{p,e}}$ by using the Bayesian rule as follows:

$$P_{x_g}(x \mid \boldsymbol{x}_{p,g}, \boldsymbol{x}_{p,e}) = \frac{Prob(x, \boldsymbol{x}_{p,g} \mid \boldsymbol{x}_{p,e})}{Prob(\boldsymbol{x}_{p,g} \mid \boldsymbol{x}_{p,e})}. \tag{6}$$

This computation requires distribution fitting steps. To avoid additional computational burden and errors introduced by distribution fitting, we use the data directly to construct a piecewise constant conditional density function (*a multi-dimensional histogram*) for each $\boldsymbol{x}_{p,e} \in \mathbb{X}_{p,e}$.

To construct the density function in the form of a multi-dimensional histogram, first, $P_{\alpha_g}(\cdot)$ is approximated by a histogram H^{α_g} with a sufficiently large number of intervals. Then for each measured point $(\boldsymbol{x}_{p,g}, x_g) \in D^{\boldsymbol{x}_{p,e}}$ and $\alpha_{g,i}$, which is the center point of each interval i of H^{α_g}, an expression rate is computed as

$$\beta_{p,i} = (1 - \alpha_{g,i}) x_g.$$

These expression rates, the concentrations of the protein regulators $\boldsymbol{x}_{p,g}$ and the frequency f_i of the corresponding intervals are used to construct a data set $D^{\boldsymbol{x}_{p,e}, \beta_p}$. Specifically, for each expression rate $\beta_{p,i}$ as computed above, $(\boldsymbol{x}_{p,g}, \beta_{p,i})$ is added to the set $D^{\boldsymbol{x}_{p,e}, \beta_p}$ f_i times. Finally, a multi-dimensional histogram $H^{p, \boldsymbol{x}_{p,e}}$ is constructed from the data set $D^{\boldsymbol{x}_{p,e}, \beta_p}$ for the expression rate β_p.

The presented method for constructing the conditional density function $H^{p, \boldsymbol{x}_{p,e}}$ has several advantages. First, it is faster than fitting multi-dimensional density functions given in (6) and provides a general method, since the shapes of these density functions are unknown. Second, by considering the discretization levels of the flow cytometry instrument and the intervals of H^{α_g}, $H^{p, \boldsymbol{x}_{p,e}}$ can capture $P_{\beta_g}(\cdot \mid \boldsymbol{x}_{p,g})$ precisely, where a deviation can only occur due to the approximation of $P_{\alpha_g}(\cdot)$. However, a degradation rate is either known or has a distribution with a low variance and compact support, *i.e.* $\textbf{supp}(P_{\alpha_g}(\cdot)) \subset (0, 1)$. If the degradation rate is known then the derived density function $H^{p, \boldsymbol{x}_{p,e}}$ is an exact representation of $P_{\beta_g}(\cdot \mid \boldsymbol{x}_{p,g})$ with respect to the available data. If, however, $P_{\alpha_g}(\cdot)$ is given, $H^{p, \boldsymbol{x}_{p,e}}$ can approximate $P_{\beta_g}(\cdot \mid \boldsymbol{x}_{p,g})$ with arbitrarily high accuracy, since $\textbf{supp}(P_{\alpha_g}(\cdot))$ is a compact set.

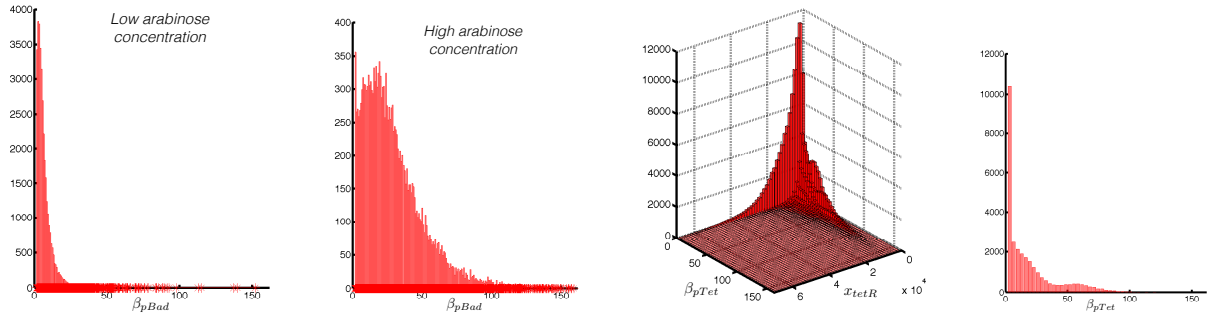
Example 3.2: The characterization data obtained as explained in Example 3.1 is used to construct histograms shown in Figure 3 for β_{pBad} and β_{pTet} by assuming that the degradation rates are known, *i.e.* for each $g \in \{tetR, GFP, RFP\}$ $P_{\alpha_g}(\bar{\alpha}_g) = 1$ for some $\bar{\alpha}_g \in (0, 1)$.

B. Gene Network Simulator

In this section, we describe a simulator that generates trajectories of the stochastic model defined in (3). The simulator is initialized by constructing the density functions from characterization data for each promoter as described in Section III-A. Then for a given trajectory length N, and initial state $\boldsymbol{x}(0)$, first a degradation rate α_{g_i} for each gene $g_i, i = 1, \ldots, n$, and an expression rate β_{p_j} for each promoter $p_j, j = 1, \ldots, s$ is sampled from the corresponding distributions and the state at time $k = 1$, $\boldsymbol{x}(1)$, is computed according to (3). In the subsequent time steps, *i.e.* for $1 \leq k \leq N$, the random variables $(\alpha_{g_i},\ \beta_{p_j})$ are sampled from a distribution that depends on the sampled value of the random variable in the previous iteration. In particular, we use *truncated sampling* that is explained next for a random variable α_g.

Truncated sampling The value of $\alpha_g(k+1)$ is sampled from the distribution of α_g truncated to the semi-open interval $(\alpha_g(k) - w, \alpha_g(k) + w]$, where $w \in \mathbb{R}_+$. Specifically, $\alpha_g(k+1)$ is sampled from the distribution function

$$F_{\alpha_g}(\alpha | \alpha_g(k)) = \frac{F_{\alpha_g}(\alpha) - F_{\alpha_g}(\alpha_g(k) - w)}{F_{\alpha_g}(\alpha_g(k) + w) - F_{\alpha_g}(\alpha_i(k) - w)}, \tag{7}$$



(a) $H^{pBad,0}$ (b) $H^{pBad,10}$ (c) H^{pTet} (d) A slice from (c)

Fig. 3. (a-b) The expression rates computed from the measured concentrations and $\bar{\alpha}_{GFP}$ are shown with red marks on the $x-$axis. (d) A slice from histogram (c), where $x_{tetR} \in [3382.8, 4745.1)$

where $F_{\alpha_g}(\alpha) = Prob(\alpha_g \leq \alpha)$. Note that, when w tends to infinity, $F_{\alpha_g}(\cdot|\alpha_g(k))$ converges to $F_{\alpha_g}(\cdot)$.

Example 3.3: We generate the trajectories of the gene network from Example 2.1 as explained above. The random variables β_p, $p = \{pBad, pTet\}$, are sampled from the histograms constructed as in Example 3.2. The expression rate from the $pTet$ promoter is sampled with respect to x_{tetR}. For example, if $x_{tetR} \in [3382.8, 4745.1)$, β_{pBad} is sampled from the histogram shown in Figure 3 (d).

IV. Statistical Analysis

In this section, we provide solutions to the problems given in Section II based on statistical analysis. We use the *Bayesian Interval Estimation* and *Bayesian Hypothesis Testing* algorithms presented in [15]. Both of the algorithms iteratively generate trajectories of system S (3) and model check the trajectories against the specification formula.

A. Statistical Hypothesis Testing

Statistical hypothesis testing is a widely used tool to prove (with bounded error) statistical assumptions on a stochastic system. In [15] a statistical model checking algorithm based on iterative Bayesian Hypothesis Testing is proposed, where the hypothesis is defined as the satisfaction of a PBLTL formula $H_0 : S \models P_{\geq\theta}(\Phi)$, and the iterative algorithm decides between $H_0 : S \models P_{\geq\theta}(\Phi)$ and $H_1 : S \models P_{<\theta}(\Phi)$ for a stochastic system S.

We use the Bayesian Hypothesis Testing algorithm to solve Problem 2.1-*(i)*. The algorithm sequentially draws a sample δ (model checking result of a trajectory σ),

$$\delta = \begin{cases} 1 & \text{if } \sigma \models \Phi \\ 0 & otherwise, \end{cases}$$

updates the available data set Δ of model checking results of sample trajectories, computes the Bayes factor $\mathcal{B} = \frac{Prob(\Delta|H_0)}{Prob(\Delta|H_1)}$ with respect to the prior knowledge, and then compares it against a fixed threshold $\lambda \geq 1$: i) accepts H_0 if $\mathcal{B} > \lambda$; ii) accepts H_1 if $\mathcal{B} < 1/\lambda$. If both i) and ii) are not satisfied, the algorithm continues by drawing another sample.

Error: For any discrete random variable and prior, both probabilities of accepting H_0 when it is wrong (Type 2 error) and rejecting H_0 when it is correct (Type 1 error) are upper bounded by $1/\lambda$. Consequently, when the algorithm accepts H_0, $S \models P_{\geq\theta}(\Phi)$ is correct with probability $1 - 1/\lambda$.

Example 4.1: We use the algorithm outlined above to decide whether the gene network described in Example 2.1 works as an inverter. Since we do not consider the dynamics of the external regulators and our data only covers fixed values of the external regulators, we initialize the external regulator and model check the gene network against PBLTL formulas $P_{\geq 0.95}(\Phi_{IL})$ and $P_{\geq 0.95}(\Phi_{IH})$, where Φ_{IL} and Φ_{IH} are sub-formulas of formula Φ_I (2):

$$\Phi_{IL} = F^{360}G^{240}(x_{RFP} > 5200), \quad (8)$$

$$\Phi_{IH} = F^{360}G^{240}(x_{RFP} < 3200). \quad (9)$$

We define the hypothesis as $H_0 : S^{\pi_0^L} \models P_{\geq 0.95}(\Phi_{IL})$, where π_0^L is the distribution of initial states such that $x_{Ara} = 0mM$ with probability 1, and the concentration of each protein is uniformly distributed over its domain. The hypothesis testing algorithm terminates after 18676 iterations, with 17763 satisfying trajectories by proving that $S^{\pi_0^L} \models P_{\geq 0.95}(\Phi_{IL})$ holds with probability $0.99, (\lambda = 100)$. Next, we define the hypothesis as $H_0 : S^{\pi_0^H} \models P_{\geq 0.95}(\Phi_{IH})$, where π_0^H is the same as π_0^L except that $x_{Ara} = 10mM$ with probability 1. The hypothesis testing algorithm terminates after 87 iterations, with 75 satisfying trajectories by proving that the alternative hypothesis $H_1 : S^{\pi_0^H} \models P_{<0.95}(\Phi_{IH})$ holds with probability $0.99(\lambda = 100)$.

B. Bayesian Interval Estimation

Interval estimation is used to find a probability range Θ for a well defined but unknown probability θ such that $\theta \in \Theta$ with arbitrarily high probability. We use the Bayesian Interval Estimation algorithm [15] to solve Problem 2.1-*(ii)*. The half size $\delta \in (0, \frac{1}{2})$ of the desired interval estimate Θ and a coverage goal $c \in (\frac{1}{2}, 1)$, $c \leq Prob(\theta \in \Theta)$, are the parameters of the algorithm. Similar to the Bayesian Hypothesis Testing algorithm, the algorithm sequentially draws a sample model checking result and updates the Bayesian estimate. The algorithm stops and outputs the current estimate $\hat{\theta}$ when the coverage goal is achieved, otherwise it continues by drawing another sample.

Error: For any discrete random variable and prior, the probability that $\theta \notin \Theta = [\hat{\theta} - \delta, \hat{\theta} + \delta]$ is upper bounded by $\frac{(1-c)\pi}{c(1-\pi)}$, where π is the prior probability that $\theta \in \Theta$.

Example 4.2: Consider the gene network from Example 2.1, and BLTL formulas Φ_{IL} (8) and Φ_{IH} (9), and the initial distributions π_0^L and π_0^H from Example 4.1. System $S^{\pi_0^L}$ satisfies Φ_{IL} with probability 0.958 and system $S^{\pi_0^H}$ satisfies Φ_{IH} with probability 0.843, which shows that the circuit works as an inverter with high probability. These probabilities are found by using the Bayesian Interval Estimation algorithm. A beta prior with $\alpha = \beta = 1$ is used and the algorithm parameters are set to $\delta = 0.01$ (half interval size), $c = 0.99$ (coverage goal), meaning that when the algorithm results in a probability estimate $\hat{\theta}$, then the unknown probability θ that the circuit satisfies the specification is in $[\hat{\theta} - 0.01, \hat{\theta} + 0.01]$ with probability $1 - \frac{(1-0.99)\times 0.02}{0.99\times 0.98}$.

C. Parameter Optimization

The problems solved in the previous sections require to specify the formula fully, *i.e.* all the thresholds and time bounds must be set. Here we show that we can use the Bayesian Hypothesis Testing algorithm to solve Problem 2.2. Specifically, we propose to iteratively use the testing algorithm to minimize or maximize one of these parameters when the rest of them together with a probability bound θ are given. Algorithm 1 presents an example of such a search routine, where a threshold is minimized through a binary search.

Algorithm 1 Threshold Minimization

Input: A system S, a BLTL formula Φ with $x_o < T$ appearing in Φ, a probability bound θ, a precision bound τ.
Output: T such that $S \models P_{\geq\theta}(\Phi)$.
  Set $T^L = x_o^{min}$ and $T^H = x_o^{max}$.
  $testT = \frac{T^L + T^H}{2}$.
  while $T^H - T^L > \tau$ **do**
    $\Phi^t = \Phi$ with $T = testT$.
    if $BHT(S, \Phi^t, \theta)$ **then** {Bayesian Hypothesis Testing}
      $T^H = testT$. $\{S \models P_{\geq\theta}(\Phi^t)\}$
    else
      $T^L = testT$. $\{S \not\models P_{\geq\theta}(\Phi^t)\}$
    end if
    $testT = \frac{T^L + T^H}{2}$.
  end while

Example 4.3: Consider the gene network from Example 2.1, and Φ_{IL} (8), Φ_{IH} (9), π_0^L and π_0^H from Example 4.1. The minimum output threshold $T^L_{RFP} = 3534$ for formula Φ_{IH} (9) and system $S^{\pi_0^H}$ is found by using Algorithm 1 with $\tau = 10, \theta = 0.95$. Via a similar algorithm with $\tau = 10, \theta = 0.95$, the maximum output threshold for formula Φ_{IL} (8) and system $S^{\pi_0^L}$ is found as $T^H_{RFP} = 5219$.

These optimized thresholds can be used to compare different gene networks designed as inverters. Assume a set of gene networks $S_i, i = 1, \ldots, l$ and corresponding characterization data are given. High $T^{H,i}_{RFP}$ and low $T^{L,i}_{RFP}$ output thresholds can be found for each S_i as explained above. Then, the gene network with the maximum threshold gap, *i.e.* $\arg\max_{i=1,\ldots,l} T^{H,i}_{RFP} - T^{L,i}_{RFP}$, can be considered as the *most robust* design. Moreover, such optimized thresholds can further be used to couple the engineered cells with electronic systems.

V. Conclusion

We developed a computational framework that allows for statistical verification of a synthetic gene network given information on the decay rates of its proteins and fluorescent microscopy experimental data characterizing its promoters. The framework is based on (1) the construction of a mathematical model in the form of a discrete-time stochastic system with parameter distributions derived from the experimental data, and (2) statistical model checking over simulated trajectories of the model. We applied the proposed computational tool to verify the behavior of a synthetic gene circuit designed to behave as a logical inverter.

References

[1] J. C. Anderson, E. J. Clarke, A. P. Arkin, and C. A. Voigt, "Environmentally Controlled Invasion of Cancer Cells by Engineered Bacteria," *Journal of Molecular Biology*, vol. 355, no. 4, pp. 619–627, Jan. 2006.

[2] J. R. Kirby, "Synthetic biology: Designer bacteria degrades toxin," *Nature Chemical Biology*, vol. 6, no. 6, pp. 398–399, Jun. 2010.

[3] S. Atsumi, T. Hanai, and J. C. Liao, "Non-fermentative pathways for synthesis of branched-chain higher alcohols as biofuels," *Nature*, vol. 451, pp. 86–89, January 2008.

[4] H. Salis, A. Tamsir, and C. Voigt, "Engineering bacterial signals and sensors," *Contrib Microbiol*, vol. 16, pp. 194–225, 2009.

[5] A. Prindle, P. Samayoa, I. Razinkov, T. Danino, L. S. Tsimring, and J. Hasty, "A sensing array of radically coupled genetic 'biopixels'," *Nature*, vol. 481, no. 7379, pp. 39–44, Jan 2012.

[6] B. Xia, S. Bhatia, B. Bubenheim, M. Dadgar, D. Densmore, and J. Anderson, "Developer's and user's guide to clotho v2.0 a software platform for the creation of synthetic biological systems." *Methods Enzymol*, vol. 498, 2011.

[7] L. Bilitchenko, A. Liu, S. Cheung, E. Weeding, B. Xia, M. Leguia, J. C. Anderson, and D. Densmore, "Eugene - A Domain Specific Language for Specifying and Constraining Synthetic Biological Parts, Devices, and Systems," *PLoS ONE*, vol. 6, no. 4, p. e18882, 2011.

[8] J. Beal, T. Lu, and R. Weiss, "Automatic Compilation from High-Level Biologically-Oriented Programming Language to Genetic Regulatory Networks," *PLoS ONE*, vol. 6, no. 8, p. e22490, Aug. 2011.

[9] J. Beal, R. Weiss, D. Densmore, A. Adler, E. Appleton, J. Babb, S. Bhatia, N. Davidsohn, T. Haddock, J. Loyall, R. Schantz, V. Vasilev, and F. Yaman, "An end-to-end workflow for engineering of biological networks from high-level specifications," *ACS Synthetic Biology*, vol. 1, no. 8, pp. 317–331, 2012.

[10] B. Yordanov, E. Appleton, R. Ganguly, E. Gol, S. Carr, S. Bhatia, T. Haddock, C. Belta, and D. Densmore, "Experimentally driven verification of synthetic biological circuits," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2012, pp. 236–241.

[11] C. Baier, J.-P. Katoen, and K. G. Larsen, *Principles of Model Checking*. MIT Press, 2008.

[12] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems," in *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, ser. LNCS, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 585–591.

[13] A. Abate, J. Katoen, J. Lygeros, and M. Prandini, "Approximate model checking of stochastic hybrid systems," *European Journal of Control*, vol. 16, no. 6, pp. 624–641, 2010.

[14] H. L. S. Younes and D. J. Musliner, "Probabilistic plan verification through acceptance sampling," in *In AIPS Workshop on Planning via Model Checking*. AAAI Press, 2002, pp. 81–88.

[15] P. Zuliani, A. Platzer, and E. M. Clarke, "Bayesian statistical model checking with application to Simulink/Stateflow verification," in *HSCC*, K. H. Johansson and W. Yi, Eds. ACM, 2010, pp. 243–252.

[16] "BioNumbers–the database of key numbers in molecular and cell biology." *Nucleic acids research*, vol. 38, no. Database issue, pp. D750–753, Jan. 2010.

[17] E. Gasteiger, C. Hoogland, A. Gattiker, S. Duvaud, M. R. Wilkins, R. D. Appel, and A. Bairoch, "Protein identification and analysis tools on the expasy server," March 2005.

[18] J. C. Wood, "Fundamental flow cytometer properties governing sensitivity and resolution." *Cytometry*, vol. 33, no. 2, pp. 260–266, 1998.