Calin Belta

Drexel University
Philadelphia, PA, USA
calin@drexel.edu

Joel M. Esposito

US Naval Academy
Annapolis, MD, USA

Jongwoo Kim
Vijay Kumar

University of Pennsylvania
Philadelphia, PA, USA

Computational Techniques for Analysis of Genetic Network Dynamics

Abstract

In this paper we propose modeling and analysis techniques for genetic networks that provide biologists with insight into the dynamics of such systems. Central to our modeling approach is the framework of hybrid systems and our analysis tools are derived from formal analysis of such systems. Given a set of states characterizing a property of biological interest $\mathcal{P}$, we present the Multi-Affine Rectangular Partition (MARP) algorithm for the construction of a set of infeasible states $\mathcal{I}$ that will never reach $\mathcal{P}$ and the Rapidly Exploring Random Forest of Trees (RRFT) algorithm for the construction of a set of feasible states $\mathcal{F}$ that will reach $\mathcal{P}$. These techniques are scalable to high dimensions and can incorporate uncertainty (partial knowledge of kinetic parameters and state uncertainty). We apply these methods to understand the genetic interactions involved in the phenomenon of luminescence production in the marine bacterium V. fischeri.

KEY WORDS—genetic networks, hybrid systems, formal analysis, rapidly-exploring random trees

1. Introduction

The recent completion of a draft of the human genome and the sequencing of several other organisms has provided a vast amount of genomic data for advancing our understanding of fundamental biological processes. However, in order to understand different cellular behaviors such as differentiation, response to environmental signals, and cell-to-cell communication, we need to study the regulatory systems determining the expression of genes. This is usually a complex process, which can be regulated at several stages such as transcription (the best studied form of regulation), translation, and post-translational modification of proteins. An example of transcriptional regulation is repression: a regulatory molecule binds to a regulatory site of some gene preventing the ribonucleic acid (RNA) polymerase from transcribing the gene. The number of regulating factors is usually large, and it involves proteins (products of other genes and possibly of the gene itself), RNA, and other molecules. A collection of interacting genes, their products, and some other molecules involved in the regulation of the genes form a "genetic regulatory network".

The traditional approach to modeling of genetic networks leads to highly nonlinear systems of differential equations for which analytical solutions are not normally possible. One way to work around the difficulties of the nonlinearities is to use simplified, approximate models. Existing work focuses on very low-dimensional genetic networks. Decoupled piecewise linear differential equations (PLDEs) are considered in Glass (1975) and Mestl, Plathe, and Omholt (1995), where gene regulation is modeled as a discontinuous step function and chemical reactions are ignored. This (over)simplified approach to modeling allows for interesting qualitative analysis (de Jong et al. 2003). An even more radical idealization is obtained if the state of a gene is abstracted to a Boolean variable and the interaction among elements to Boolean functions, as in Boolean networks (Kauffmann 1969). While amenable for interesting analysis, the methods mentioned above are based on assumptions which disregard important biochemical phenomena. Most of them only capture protein dynamics but cannot accommodate chemical reactions (Kauffmann 1969).

Our modeling approach is based on hybrid systems (Lynch and Krogh 2000), i.e., systems in which discrete events are

The International Journal of Robotics Research
Vol. 24, No. 2–3, February–March 2005, pp. 219-235,
DOI: 10.1177/0278364905050359

combined with continuous differential equations. In rectangular regions of the state space where the chemical dynamics can be reasonably approximated as smooth we model them using deterministic, nonlinear (multi-affine) ordinary differential equations. We assume that the chemical concentrations are spatially homogeneous, locally, eliminating the need to use partial differential equations or rarefied molecular stochastic models. Similar to the Boolean approach described above, the discrete component of the model captures the switching behavior that is observed in phenomena such as transcription, protein–protein interactions, and cell division and growth. We also propose the use of hybrid system as the natural framework for giving a global description of a biological system described locally around operating points by simpler dynamics, which are easier to approach for analysis. Our own work using hybrid systems to model, simulate and perform preliminary analysis on low-dimensional genetic networks is given (Alur et al. 2001, 2002a, 2002b; Belta et al. 2001; Belta, Habets, and Kumar 2002); other work includes Glass (1975) and de Jong et al. (2003).

We are interested in developing general modeling, simulation, and analysis techniques for metabolic and genetic networks. Our ultimate goal is to create tools enabling us to answer biologically significant questions of the following types. "If an organism is initially in a state described by certain ranges of metabolite and enzyme concentrations, and levels of activation of genes, describe the set of states that the organism can reach in T seconds." Or, "describe the states with the property that if the system starts in any of them it will never reach an undesired state." The undesired state could correspond, for example, to a certain disease. We denote the set associated with properties of interest as $\mathcal{P}$. To answer the questions formulated above, we are interested in determining two disjoint sets of initial conditions that can be associated with the set $\mathcal{P}$ (see Figure 1). First, we are interested in characterizing feasible sets $\mathcal{F}$, consisting of initial conditions that make it possible for the system to enter the set $\mathcal{P}$ under some combination of parameter values and noise. We are also interested in the infeasible sets $\mathcal{I}$, from which it is impossible to enter the set $\mathcal{P}$. Knowledge of $\mathcal{F}$ may assist in experiment design; while a knowledge of the set $\mathcal{I}$, is particularly useful for model validation. If experimental data indicate the system enters $\mathcal{P}$ and if experimental conditions were known to lie in set $\mathcal{I}$, one could prove the model to be inconsistent with the experimental data. In many ways, the approaches to generating the two sets, and the information they encode, are complementary—neither approach alone is complete. Together however, knowledge of the sets $\mathcal{F}$ and $\mathcal{I}$ can enable us to make some powerful assertions about the system's behavior.

The connection between biomolecular networks and robotic systems exists on two levels. From a modeling point of view, robotic systems share many of the salient features of biological system models described above. Just as the rate

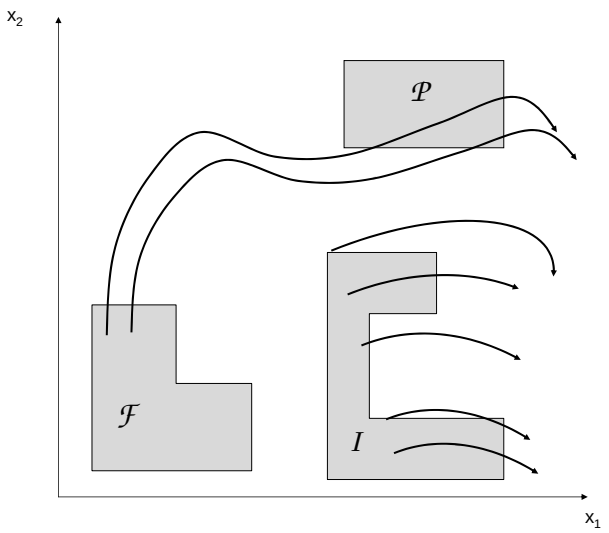

Fig. 1. The three sets of interest: the property set $\mathcal{P}$, the infeasible set $\mathcal{I}$ from which no trajectory can enter $\mathcal{P}$, and the feasible set $\mathcal{F}$ from which there exists at least one trajectory which can enter $\mathcal{P}$.

equations for biomolecular networks are known to qualitatively switch based on the presence or absence of various inhibitor genes, robotic systems often employ different controllers and estimators in different regimes (Das et al. 2002) and their dynamics switch based on the contact mechanics of rolling and sliding. Therefore, both types systems can be modeled as hybrid systems. In addition, many robotic systems consist of multi-agent teams, and therefore the interactions and messaging among the team members must be taken into account and many of the system properties are distributed spatially. Multicell networks behave in much the same way. Finally, the significant modeling uncertainty, which is central to our discussion and analysis of biological systems, is a common theme in mobile robotics operating in unstructured environments.

Perhaps a deeper connection between the two fields exists at the level of the types of problems we seek to solve. The problem of finding sets of states $\mathcal{F}$, from which the system may reach $\mathcal{P}$, is similar to the motion planning problem in robotics where the goal is to find a trajectory (if one exists) from the starting configuration to the goal configuration. Determining an infeasible set $\mathcal{I}$, from which it is impossible to reach the property set $\mathcal{P}$ is closely related to trajectory generation, controllability and steering. As one can imagine, the literature on simulation and verification of hybrid systems is also particularly relevant to our discussion (Henzinger et al. 1995, 2000; Lafferriere, Pappas, and Yovine 1999; Butchkarev and Tripakis 2000; Mitchell and Tomlin 2000; Asarin, Dang,

and Maler 2001; Chutinam and Krogh 2003; Tabuda and Pappas 2003).

In this paper we develop methodologies for finding the sets shown in Figure 1. In Section 2, we introduce the hybrid system modeling paradigm and provide the basic definitions that will be used throughout the paper. In Section 3, we exploit the particular structure of the hybrid models of genetic networks to derive a computationally attractive algorithm, the Multi-Affine Rectangular Partitioning (MARP) algorithm, to compute infeasible sets $\mathcal{I}$ based on evaluating the vector field at the vertices of the rectangular invariants. In Section 4 we extend the popular Rapidly Exploring Random Tree (RRT) algorithm from the motion planning literature to address time-invariant uncertainty such as unknown initial conditions or rate constants. The resulting algorithm, called the Rapidly Exploring Random Forest of Trees (RRFT) algorithm, allows one to address the reachability problem probabilistically for complex high-dimensional systems having both time-varying and time-invariant uncertainty, providing a natural way to determine if a set should be included in $\mathcal{F}$. The algorithm, which has potential for parallel implementation, estimates the growth and coverage of the trees and uses this information to modify the search. The usefulness of the two algorithms is illustrated in Section 5, where we study the phenomenon of bioluminescence production in the marine bacterium *V. fischeri* by analyzing the corresponding genetic network. The paper concludes with final remarks and directions of future research in Section 6.

2. Hybrid System Modeling of Genetic Networks

Hybrid systems are dynamical systems with both discrete and continuous state changes (Lynch and Krogh 2000). We are interested in a special class of hybrid systems, called switched systems, which are defined as having different dynamics in different non-overlapping regions of the state space. In our view, hybrid and switched systems are appropriate and attractive for modeling the dynamics of biomolecular networks for two main reasons:

Hybrid Systems are Global Descriptions from Simpler Local Models. Computationally attractive formalisms for modeling biomolecular networks such as linearizations, half-systems (Savageau and Voit 1987), synergistic (S) systems (Savageau 1969), generalized mass action (GMA; Peschel and Mende 1986), and power law (Heinrich and Schuster 1996) are only valid locally around operating points. For example, the S-systems can be thought of as linearizations of "real" systems in logarithmic coordinates (Savageau 1969). Then, in this case, a global description of the network is a collection of regions with different polynomial vector fields in each region, therefore a hybrid system. The specific nonlinearities of dynamics in each region are simpler than the dynamics of a global continuous description, and easier to approach for analysis.

Hybrid Systems Capture Discrete Events. Discrete dynamics are necessary to capture switching behavior that is observed in phenomena such as transcription, protein–protein interactions, and cell division and growth. Consider, for example, the case when a metabolite from the network regulates the production of a metabolic enzyme expressed from a gene with a strong promoter. Then, the gene can be "on" and "off", which induces two different dynamics of the network, as a function of the concentration of the metabolite.

Formally, hybrid systems are defined as tuples

$$HS = (Q, X, \mathbf{X}^0, I, T, F) \tag{1}$$

where Q is a finite set of discrete variables, X is the set of continuous variables x, $\mathbf{X}$ is the set of all evaluations of x over the corresponding domains, $\mathbf{Q}$ is the countable set of discrete states, called modes, or locations, $\mathbf{X}^0 \subset \mathbf{Q} \times \mathbf{X}$ is a set of initial states, I is a map which assigns to each discrete location in $\mathbf{Q}$ an invariant set, $T \subset \mathbf{Q} \times \mathbf{X} \times \mathbf{Q}$ is a set of discrete transitions, and $F : \mathbf{Q} \to (\mathbf{X} \to T\mathbf{X})$ is a mapping that specifies the continuous (possibly time-dependent) flow in each discrete state.

We focus on vector fields f that are products of the state components to capture the nonlinearities, which are specific to dynamics of chemical reactions. To take into account possible modeling noise and to accommodate non-deterministic modeling approaches, we also allow for an additive noise term $\nu(t)$ in the vector field. Therefore, as suggested by Kepler and Elston (2001), the vector field associated with the map F takes the form $F = f(x) + \nu(t)$. The form of f is made precise in the next subsection, while ν is discussed in Section 4.

We are also interested in constant parameters that might be unknown. Examples of these parameters are binding constants and other constants determining reaction rate kinetics. While these constants are known to lie within a certain range, their exact values are often unknown. It is useful to note that these unknown constants can be viewed as state variables with trivial dynamics, e.g., $x = c,\ \dot{x} = 0$, while the corresponding projection of $\mathbf{X}^0$ characterizes the known bounds. Thus, our definition of HS in eq. (1) allows for unknown parameters that lie in some specified set.

Finally, since molecular networks are qualitatively described in terms of ranges of concentrations of the involved species, a rectangular partition of the state space is naturally induced and the invariants are rectangular.

2.1. Rectangular Multi-Affine Hybrid Systems

Rectangular multi-affine hybrid systems are characterized by rectangular invariants and multi-affine continuous dynamics. A function with a vector argument is called multi-affine if it is affine in each component of its argument, i.e., when all other components are kept constant. More formally, we define a multi-affine function as follows.

DEFINITION 1. [Multi-affine function] A multi-affine function $f : \mathbb{R}^N \longrightarrow \mathbb{R}^N$ is a polynomial in the indeterminates $x_1, \ldots, x_N$ with the property that the degree of f in any of the indeterminates $x_1, \ldots, x_N$ is less than or equal to 1. Stated differently, f has the form

$$f(x_1, \ldots, x_N) = \sum_{i_1,\ldots,i_N \in \{0,1\}} c_{i_1,\ldots,i_N} x_1^{i_1} \cdots x_N^{i_N}, \qquad (2)$$

with $c_{i_1,\ldots,i_N} \in \mathbb{R}^N$ for all $i_1, \ldots, i_N \in \{0, 1\}$ and using the convention that if $i_k = 0$, then $x_k^{i_k} = 1$.

An N-dimensional rectangle in $\mathbb{R}^N$ is characterized by two vectors $a = (a_1, \ldots, a_N) \in \mathbb{R}^N$ and $b = (b_1, \ldots, b_N) \in \mathbb{R}^N$, with the property that $a_i < b_i$ for $i = 1, \ldots, N$:

$$R_N(a, b) = \{x = (x_1, \ldots, x_N) \in \mathbb{R}^N \mid i = 1, \ldots, N : a_i \le x_i \le b_i\}. \qquad (3)$$

The variables $x_i, i = 1, \ldots, N$ are species concentrations and are restricted to the positive quadrant. Also, there are practical upper bounds on the concentration of each species. Therefore, the set $\mathbf{X}$ as in eq. (1) is usually specified as an N-rectangle. A rectangular partition of X is defined as follows. Each axis Ox_i, $i = 1, \ldots, N$ is divided into $n_i \ge 1$ intervals by the thresholds $0 = \theta_i^0 < \theta_i^1 < \ldots < \theta_i^{n_i}$. The jth interval on the Ox_i-axis, $i = 1, \ldots, N$ is therefore defined as $\theta_i^{j-1} \le x_i < \theta_i^j$, $j = 1, \ldots, n_i$. By convention, $\theta_i^0 = 0$ and $\theta_i^{n_i}$ is an upper bound giving a physical limit of x_i. The thresholds θ are defined as values of species concentrations for each the dynamics of the overall system changes. For example, they can be concentrations of regulatory species for which specific genes are turned "on" and "off". The division of the axes determines a partition of the state space into $\prod_{i=1}^N n_i$ rectangles. If we let

$$a^{(q_1\ldots q_N)} = (\theta_1^{q_1-1}, \ldots, \theta_N^{q_N-1}) \in \mathbb{R}^N, \; b^{(q_1\ldots q_N)} = (\theta_1^{q_1}, \ldots, \theta_N^{q_N}) \in \mathbb{R}^N, \qquad (4)$$

for $q_i = 1, \ldots, n_i, i = 1, \ldots, N$, then an arbitrary rectangle in the partition is given by

$$R_N(a^{(q_1\ldots q_N)}, b^{(q_1\ldots q_N)}) = \{(x_1, \ldots, x_N) \in \mathbb{R}^N | \theta_i^{q_i-1} \le x_i \le \theta_i^{q_i}, \; i = 1, \ldots, N\}. \qquad (5)$$

Due to the different levels of gene transcription activation and enzymatic action, in each of the rectangles the system evolves along specific multi-affine vector fields (2):

$$f^{(q_1\ldots q_N)}(x_1, \ldots, x_N) = \sum_{i_1,\ldots,i_N \in \{0,1\}} c_{i_1,\ldots,i_N}^{(q_1\ldots q_N)} x_1^{i_1} \cdots x_N^{i_N}, \qquad (6)$$

where $x \in R_N(a^{(q_1\ldots q_N)}, b^{(q_1\ldots q_N)})$ and $c_{i_1,\ldots,i_N}^{(q_1\ldots q_N)} \in \mathbb{R}^N$ for all $i_1, \ldots, i_N \in \{0, 1\}$ captures specific reaction rates.

Therefore, our models of biomolecular networks are hybrid systems (1) with the set of labels for the discrete states $Q = (q_1 \ldots q_N)$, the set of all $\prod_{i=1}^N n_i$ modes $\mathbf{Q} = \{(q_1 \ldots q_N) | \, q_i = 1, \ldots, n_i, \, i = 1, \ldots, N\}$, X is the set of species symbols $x_1, \ldots, x_N$, the invariant $I(q_1 \ldots q_N)$ is the corresponding rectangle (5), and the map F has a deterministic part described by eq. (6). A transition $((q_1 \ldots q_N), x, (q_1' \ldots q_N'))$ corresponds to the crossing of the boundary between rectangles $I(q_1 \ldots q_N)$ and $I(q_1' \ldots q_N')$ at state x.

REMARK 1. Due to the particular shape of the invariants, a convenient way of representing a rectangular multi-affine hybrid system (6), (5) is as a simple graph with $\prod_{i=1}^N n_i$ nodes. Node $(q_1 \ldots q_N)$ corresponds to rectangle $I(q_1 \ldots q_N) = R_N(a^{(q_1\ldots q_N)}, b^{(q_1\ldots q_N)})$ and has associated dynamics (6). An edge in the graph connects nodes corresponding to adjacent rectangles, i.e., there is an edge between any pair of nodes that differ by a Hamming distance of 1. See Figure 7 for an example with $N = 3$, $n_1 = n_2 = n_3 = 3$.

2.2. Set Definitions

As stated before, we are interested in characterizing the properties of the system HS related to whether it can reach a given set of interest $\mathcal{P}$ described by a rectangle $I(p_1 \ldots p_N)$, $(p_1 \ldots p_N) \in \mathbf{Q}$.

DEFINITION 2. [Infeasible set] An infeasible set $\mathcal{I}$ is a collection of rectangles $I(q_1 \ldots q_N)$, $(q_1 \ldots q_N) \in \mathbf{Q}$ with the property that the system can never reach $I(p_1 \ldots p_N)$ if it starts in any of the initial states contained in any of rectangles in $\mathcal{I}$.

DEFINITION 3. [Feasible set] A feasible set $\mathcal{F}$ is a collection of rectangles $I(q_1 \ldots q_N)$, $(q_1 \ldots q_N) \in \mathbf{Q}$ with the property that they contain initial states that will reach $I(p_1 \ldots p_N)$ in a pre-specified finite time interval.

3. Computing An Infeasible Set $\mathcal{I}$

In this section we introduce the MARP algorithm, which uses the properties of hybrid multi-affine rectangular systems to construct infeasible sets $\mathcal{I}$, and extends some results presented in Belta, Habets, and Kumar (2002). In the form presented in this paper, the algorithm can only be applied to deterministic vector fields where $F = f$. However, as explained in Section 1, it can easily accommodate rectangular parametric uncertainties since set-valued uncertainty in a constant parameter can be included in the set of initial conditions, $\mathbf{X}^0$.

3.1. Preliminaries

The MARP algorithm is based on the fact that the value of a multi-affine function (6) is uniquely determined everywhere

in the rectangular invariant $I(q_1 \dots q_N)$ by its values at the vertices. Moreover, it is a convex combination of these values.

Formally, for an arbitrary rectangle (3), let

$$V_N(a, b) = \prod_{i=1}^{N} \{a_i, b_i\} \tag{7}$$

denote the set of its 2^N vertices. Let $\xi : \{a_1, \dots, a_N, b_1, \dots, b_N\} \longrightarrow \{0, 1\}$ be defined by

$$\xi(a_k) = 0, \quad \xi(b_k) = 1, \quad k = 1, \dots, N. \tag{8}$$

PROPOSITION 1. A multi-affine function $f : R_N(a, b) \longrightarrow \mathbb{R}^N$ is a convex combination of its values $f(v_1, \dots, v_N)$ at the vertices $V_N(a, b)$. Explicitly,

$$f(x_1, \dots, x_N) = \sum_{(v_1, \dots, v_N) \in V_N(a,b)} \prod_{k=1}^{N} \left(\frac{x_k - a_k}{b_k - a_k}\right)^{\xi(v_k)} \left(\frac{b_k - x_k}{b_k - a_k}\right)^{1-\xi(v_k)} f(v_1, \dots, v_N), \tag{9}$$

with

$$1 = \sum_{(v_1, \dots, v_N) \in V_N(a,b)} \prod_{k=1}^{N} \left(\frac{x_k - a_k}{b_k - a_k}\right)^{\xi(v_k)} \left(\frac{b_k - x_k}{b_k - a_k}\right)^{1-\xi(v_k)} \tag{10}$$

where $(v_1, \dots, v_N) \in V_N(a, b)$.

The proof of this proposition can be found in Belta, Habets, and Kumar (2002). Since the projection of a multi-affine vector field along a given direction is a multi-affine function, an immediate consequence of Proposition 1 can be used to develop a computationally efficient algorithm for constructing infeasible sets for rectangular multi-affine hybrid systems, as follows.

COROLLARY 1. The projection of a multi-affine vector field defined on a rectangle along a given direction is positive (negative) everywhere in the rectangle if and only if its projection along that direction is positive (negative) at the vertices.

3.2. MARP algorithm

We assume that the piecewise defined vector field (6) (possibly non-differentiable) is continuous everywhere, i.e., the vector fields in adjacent rectangles coincide on the common facet. A simple consequence of Corollary 1 can be used to qualitatively analyze the system.

Corollary 1 is applied to the facets of the N-rectangles (5) and to the projections of the vector fields along the corresponding outer normals. Each facet is an $(N-1)$-rectangle. An infeasible set $\mathcal{I}$ can be built by defining an orientation for the simple graph of the network defined in Remark 1. We allow for both unidirectional and bidirectional edges in the oriented graph. The semantics of the orientation are defined as follows. Let $(q_1 \dots q_N)$ and $(q'_1 \dots q'_N)$ be two adjacent nodes in the graph and $I(q_1 \dots q_N)$ and $I(q'_1 \dots q'_N)$ the corresponding adjacent rectangles. A unidirectional edge from $(q_1 \dots q_N)$ to $(q'_1 \dots q'_N)$ means that there exists at least one trajectory originating in $I(q_1 \dots q_N)$ that enters into $I(q'_1 \dots q'_N)$ through the separating facet, and there is no trajectory starting in $I(q'_1 \dots q'_N)$ going to $I(q_1 \dots q_N)$ through that facet. A bi-directional edge ensures the existence of at least one trajectory originating in $I(q_1 \dots q_N)$ entering into $I(q'_1 \dots q'_N)$ and at least one trajectory originating in $I(q'_1 \dots q'_N)$ entering into $I(q_1 \dots q_N)$.

Algorithm 1. Define an oriented graph

for each node $(q_1 \dots q_N)$, $q_i = 1, \dots, n_i$, $i = 1, \dots, N$ **do**
  for each incident edge **do**
    for each vertex of the corresponding facet **do**
      calculate the projection of $f^{(q_1 \cdots q_N)}$ along the outer normal of the facet
    end for
    if the projections are positive at all vertices **then**
      the edge is unidirectional oriented out of $(q_1 \dots q_N)$
    end if
    if the projections are negative at all vertices **then**
      the edge is unidirectional oriented towards $(q_1 \dots q_N)$
    end if
    if there is a sign change among the projections at the *vertices* **then**
      the edge is bidirectional
    end if
  end for
end for

Note that, in the oversimplified description above, Algorithm 1 seems inefficient. Indeed, if we apply it to all the rectangles in the partition, most of the vertices are visited more than once, and such multiple evaluation at vertices is avoidable because the vector fields in adjacent triangles match on the separating facet. A more efficient description would require more complicated notation and a more detailed discussion which we omit because it is peripheral to the main ideas in the paper.

Using the oriented graph, we can now construct an infeasible set $\mathcal{I}$. Let $\mathcal{P} = I(p_1 \dots p_N)$ denote the target rectangle, or, equivalently, $(p_1 \dots p_N)$ is the target node in the graph. The following algorithm constructs a set $\mathcal{R}$ of nodes with the property that if the system starts in any of the corresponding rectangles, then it may be possible to reach $\mathcal{P}$. The complement of this set is an infeasible set $\mathcal{I}$.

Algorithm 2. Construct an infeasible set $\mathcal{I}$

initialize $\mathcal{R}$ with $\mathcal{P}$

repeat
  for each element $(q_1 \dots q_N)$ of $\mathcal{R}$
    for all incident nodes $(q'_1 \dots q'_N)$ connected with an edge (uni or bi-directional) oriented towards $(q_1 \dots q_N)$ **do**
      if $(q'_1 \dots q'_N)$ is not already in $\mathcal{R}$ **then**
        add $(q'_1 \dots q'_N)$ to $\mathcal{R}$
      end if
    end for
  end for
until cardinality of $\mathcal{R}$ increases
$\mathcal{I} :=$ complement of $\mathcal{R}$ with respect to the set $\mathbf{Q}$ of all nodes

Algorithm 2 for the construction of the infeasible set $\mathcal{I}$ might be too conservative, i.e., the set $\mathcal{I}$ might be unnecessarily small. Indeed, our method guarantees the existence of a trajectory from a rectangle $I(q_1 \dots q_N)$ to an adjacent rectangle $I(q'_1 \dots q'_N)$ if the (unidirectional or bidirectional) edge between the corresponding nodes in the oriented graph has an arrow from $(q_1 \dots q_N)$ to $(q'_1 \dots q'_N)$. However, if there is an edge from $(q_1 \dots q_N)$ to $(q'_1 \dots q'_N)$ and also an edge from $(q'_1 \dots q'_N)$ to $(q''_1 \dots q''_N)$, we cannot guarantee that there is a trajectory of the system from $I(q_1 \dots q_N)$ to $I(q''_1 \dots q''_N)$. In our analysis, we simply say that there might be a trajectory from $I(q_1 \dots q_N)$ to $I(q''_1 \dots q''_N)$.

This "conservativeness" is the main issue in discrete abstractions, where the central problem is to determine partitions of continuous or hybrid systems such that the discrete quotient determined by the partition is equivalent with the initial system with respect to reachability properties. Intuitively, it is easy to see that this problem is solved if and only if all trajectories in a given region reach exactly one neighboring region. In this case, the initial continuous or hybrid system is called decidable and the discrete quotient induced by the partition is said to be bi-similar (Park 1980; Pappas 2003) with the initial system. Finding classes of decidable systems is a very hard problem that received much attention lately (Henzinger et al. 1995). In this context, Algorithm 1 produces a "sufficient" abstraction (Alur, Dang, and Ivancic 2002c), that can be used to "conservatively" construct infeasible sets.

REMARK 2. The above algorithm can be easily extended to construct infeasible sets $\mathcal{I}$ under rectangular parameter uncertainties. This is possible because the parameters c capturing kinetic constants enter the vector fields (6) in the same way as the variables x, so the system (1) defined on an extended space formed by species concentrations and parameters is still characterized by multi-affine vector fields. The components of the vector fields corresponding to parameters will be zero, meaning that the kinetic constants are assumed constant but unknown within given ranges.

REMARK 3. Algorithm 1 requires the vector field f to be evaluated at each vertex. If there are N_r rectangular sets in the partition, the number of evaluations is $N_r \times 2^N$. If each coordinate is divided into K intervals, $N_r = K^N$. Thus, the number of computations scales as $(2K)^N$. While the complexity of this algorithm is exponential in the number of dimension, the alternative, which is exhaustive simulation, is impractical.

4. Determining The Feasible Set $\mathcal{F}$

In this section we briefly describe the RRFT algorithm, a randomized algorithm that can be used to delineate a feasible set $\mathcal{F}$—a collection of rectangles $I(q_1, \dots, q_N)$, $(q_1, \dots, q_N) \in \mathbf{Q}$ which contain initial conditions that can reach $\mathcal{P}$. Our algorithm leverages the work in randomized motion planning pioneered by the robotics community. We briefly review this work before introducing our algorithm.

4.1. Motion Planning

Probabilistic Road Maps (PRMs) can be used to solve the motion problem (Amato and Wu 1996; Kavraki et al. 1996), which involves finding a path from a starting point to a goal point in configuration space. The problem is usually cast in a geometric setting with no kinematics or dynamics. In contrast, RRTs (LaValle and Kuffner 2001a) generate random states for dynamic systems directly by working in the space of admissible input functions $u(t) \in \mathcal{U}$. The algorithm (see Figures 2 and 3) constructs a tree $\mathcal{T}_{x^0}$ rooted at initial state x^0, whose vertices are states $x \in X$ and whose edges are inputs $u(t) \in \mathcal{U}$, which cause the system to evolve from one vertex to a connected vertex. The algorithm constructs the tree beginning with a user-supplied initial state, which we refer to as the seed value. A sample state is generated at random, $x^{rand} \in X$. It is then determined which of the existing states, $x^{near} \in \mathcal{T}_{x^0}$, in the tree are closest to the new state, and which $u^{new}(t) \in \mathcal{U}$, when applied for predetermined time interval Δt, would bring the system as close as possible to x^{rand}. The resulting new state x^{new} is added as a vertex to $\mathcal{T}_{x^0}$ with $u^{new}(t)$ the input characterizing the edge from x^{near} to x^{new}. This procedure has the effect of growing a tree whose distribution of vertices approaches that of the random distribution which was used to create x^{rand}, causing it to cover the state space rather rapidly. A survey of the algorithm's properties appears in LaValle and Kuffner (2001b).

Of course, the application of such algorithms requires the ability to simulate system dynamics. For hybrid systems, this requires a special set of algorithms (e.g. Park and Barton 1996; Esposito, Pappas, and Kumar 2001a) to properly address the non-smooth nature, and integration algorithms capable of handling system evolving at disparate time-scales (e.g., Esposito and Kumar 2001; Esposito, Pappas, and Kumar 2001b).

4.2. The RRFT Algorithm

Unlike motion planning in which the end goal is to physically steer the system, our intention is merely to deter-

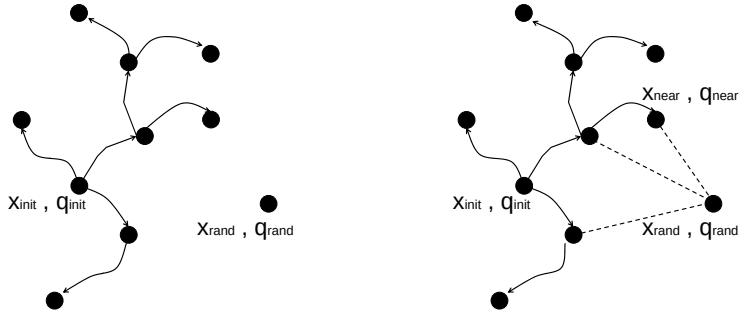

Fig. 2. Growth of individual trees in the RRT algorithm (LaValle and Kuffner 2001a). Each tree consists of vertices which are states x, and edges which are input functions $u(t) \in \mathcal{U}$. First, a new state is generated at random, x^{rand} (left). The algorithm then determines the closest state, x^{near} in the tree to the random state (right).

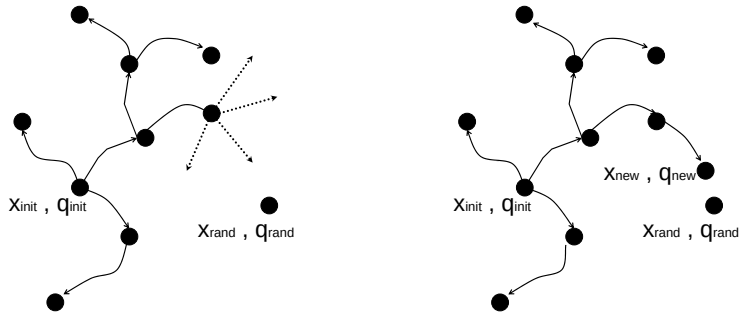

Fig. 3. Growth of individual trees in the RRT algorithm (continued from Figure 2). After finding the closest node, the algorithm determines which $u(t) \in \mathcal{U}$ brings x^{near} closest to x^{rand} (left). $u^{new}(t)$ is applied for a predetermined duration Δt and the new state x^{new} and u^{new} are added to the tree (right).

mine if it is possible for the system to reach $\mathcal{P}$ from some $x^0 \in I(q_1, \ldots, q_N) \subset \mathbf{X^0}$ within a finite time-span $t \in [t_0, t_f]$. If so, the set $I(q_1, \ldots q_N)$ is added to $\mathcal{F}$. All possible $I(q_1, \ldots q_N)$ in $\mathbf{X^0}$ are tested. In this way, our usage of the RRT method for analysis rather than synthesis (Karatas and Bullo 2001; Frazzoli, Dahleh, and Feron 2002; Kim and Ostrowski 2003) is closely related to our work on test generation for hybrid systems (Kim, Keller, and Kumar 2003). While the RRT algorithm is in many ways suited to applications such as ours, where the system dynamics are complex and high-dimensional, the RRT only addresses time-varying inputs such as $u(t)$. Recall that the evolution of our hybrid system is characterized by two elements:

- the initial condition $x^0 \in \mathbf{X^0}$ for the evolution of the state;
- The exogenous modeling noise $\nu(t) \in \mathcal{N}$ that "steers" the system.

In our algorithm, the repeated application of the RRT algorithm results in a tree for every choice of initial condition x^0. Accordingly, we need to consider a set of trees that rapidly explore the state space.

One key component of this approach is that each RRT can be computed in parallel on a different CPUs; therefore, we assume a fixed computational resource that will dictate the number of trees that can be simultaneously computed in parallel. Let this number be n_t. We propose the RRFT algorithm as follows. For each set $I(q_1, \ldots, q_N)$ in $\mathbf{X^0}$, a set of seed values $S = \{s^1, \ldots s^{n_t}\}$ is generated from a quasi-random sequence, where each $s^i \in I(q_1, \ldots, q_N)$. RRTs, $\mathcal{T}_{s^1}, \ldots \mathcal{T}_{s^{n_t}}$, are planted for each of these "seed" values. As the RRT algorithm progresses, we monitor the progress of each tree. If, at any point, the growth of one of the trees (as measured by a function $g(\mathcal{T}_{s^i})$) drops below a threshold $\bar{g}$, or the coverage of the state space (as measured by some function $c(\mathcal{T}_{s^i})$) drops below a threshold $\bar{c}$, the tree is terminated. Provided the set $I(q_1, \ldots, q_N)$ is not adequately covered (again as measured by some function $\mu(S)$) a new "seed" is planted and a new tree is initiated. The process of planting and growing new trees continues until a trajectory linking $I(q_1, \ldots q_N)$ and $\mathcal{P}$

is discovered (in which case $I(q_1, \ldots q_N)$ is added to $\mathcal{F}$), or until $I(q_1, \ldots q_N)$ is sufficiently covered ($\mu(S) \leq \bar{\mu}$) with trees that have stopped growing. A new set $I(q'_1, \ldots, q'_N)$ is selected and the process is repeated until all of $\mathbf{X^0}$ has been tested.

We defer the discussion of how to compute the functions $g(\mathcal{T}_{s^i})$, $c(\mathcal{T}_{s^i})$, and $\mu(S)$ until Section 4.3. Note that in the description of the algorithm below, we use the notation $x^0 + \int^{\Delta t} HS(v(t))\,\mathrm{d}t$ to denote the simulation of the hybrid system HS, characterizing the biomolecular network, over an interval time Δt, using the disturbance function $v(t)$, and initial condition x^0.

Algorithm 3. Construct a feasible set $\mathcal{F}$.

for each $I(q_1, \ldots, q_N)$ in $\mathbf{X^0}$ **do**
  Generate initial seed set $S = \{s^1, \ldots, s^{n_t}\}$ where $s^i \in I(q_1, \ldots, q_N)$
  for $i = 1, \ldots n_t$ **do**
    Initialize RRT $\mathcal{T}_{s^i}$
  end for
  while (true) **do**
    for $i = 1, \ldots n_t$ **do**
      Extend($\mathcal{T}_{s^i}$)
      if $\mathcal{T}_{s^i} \cap \mathcal{P} \neq \emptyset$ **then**
        add $I(q_1, \ldots, q_N)$ to $\mathcal{F}$
        break
      else
        if $g(\mathcal{T}_{s^i}) \leq \bar{g}$, OR,$c(\mathcal{T}_{s^i}) \leq \bar{c}$ **then**
          terminate $\mathcal{T}_{s^i}$
          $n_t \leftarrow n_t - 1$
          if $\mu(S) > \bar{\mu}$ **then**
            generate new seed point s^{new} and append to S
            initialize $\mathcal{T}_{s^{new}}$
            $n_t \leftarrow n_t + 1$
          end if
        end if
      end if
    end for
    if $n_t = 0$ **then**
      break
    end if
  end while
end for

Algorithm 4. Extend($\mathcal{T}$).

$x^{rand} \leftarrow$ random()
$x^{near} \leftarrow$ nearestNeighbor($\mathcal{T}, x^{rand}$)
$v^{new} = \arg\min_{v \in \mathcal{N}} \{dist((x^{rand}, x^{near} + \int^{\Delta t} HS(v(t))dt)\}$
$x^{new} = x^{near} + \int^{\Delta t} HS(v^{new}(t))dt$
add vertex x^{new} to $\mathcal{T}$
add edge v^{new}, from x^{near} to x^{new}, to $\mathcal{T}$

4.3. Adequacy Criteria

Theoretical results on RRTs from the motion planning literature (LaValle and Kuffner 2001b) suggest that as the number of nodes in the tree goes to infinity, the tree should cover the entire reachable set, although it is impossible to determine the reachable set in advance. However, because the input function space must be discretized and because the algorithm is inherently greedy, it is possible for the tree to create new nodes that are very close to the existing nodes. Therefore, there are two plausible reasons to stop growing a tree $\mathcal{T}_{s^i}$: (1) the state space is sufficiently covered that one can be confident no trajectory exists linking $I(q_1, \ldots, q_N)$ and $\mathcal{P}$; or (2) the tree is no longer actively growing.

In order to determine how to allocate our computational resources effectively we must monitor the progress of each tree. In particular, we are interested in three measures: the coverage of the state space by an individual RRT $c(\mathcal{T}_{s^i})$; the growth rate of an individual RRT, $g(\mathcal{T}_{s^i})$; and coverage of $I(q_1, \ldots, q_N)$ by the set of seeds S, $\mu(S)$. We explored different measures of growth and coverage including the discrepancy and dispersion (Branicky et al. 2001), the size of the Voronoi regions (LaValle and Kuffner 2001b), as well as the volume of the convex hull of the tree nodes and other bounding polygons. However, we found these measures to be either too expensive to compute in high dimensions or overly conservative. Instead, we begin by overlaying a grid of n_g points and spacing δ on the state space. Note that this grid is not used to construct the tree, merely to assess its coverage and growth. We calculate the minimum distance from each grid point to the set of nodes in the tree, d_j. The quantity $min(d_j, \delta)$ may be thought of as the radius of the largest ball centered at each grid point which does not contain a tree node or another grid point (see Figure 4). Clearly, the maximum value of the radius is δ, the spacing between adjacent grid points. It should be stressed that this list of distances can be updated incrementally as new tree nodes are added, since the effect of each new node is local. We define the coverage of the tree $\mathcal{T}_{s^i}$, $c(\mathcal{T}_{s^i})$, as the average of all the distances obtained in this manner, normalized by the grid spacing:

$$c(\mathcal{T}_{s^i}) = \frac{1}{\delta} \sum_{j=1}^{n_g} \frac{\min(d_j, \delta)}{n_g}. \tag{11}$$

Here, n_g is the number of grid points, and d_j is the radius of the largest ball centered at each grid point. Clearly, this measure is a monotonically decreasing function. If it goes to zero on a given grid, it tells us that any set whose distance along its smallest dimension is greater than the grid spacing has been entered. Said another way, the state space is covered up to a resolution equal to the grid spacing. This measure is similar to an approximation of dispersion (Branicky et al. 2001), but less conservative and faster to compute. Overall one of the advantages of this measure is that the grid size can be as fine or coarse as one chooses. Finer grids will require more distance queries but are more accurate indications of coverage.

The derivative of $c(\mathcal{T}_i)$ indicates the growth of the tree. Therefore

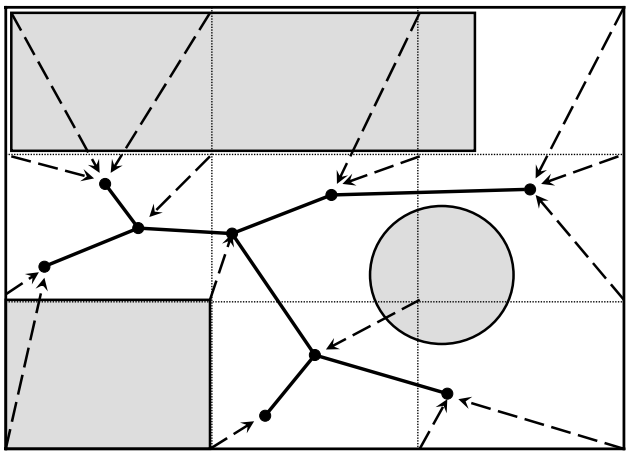

Fig. 4. A grid is superimposed on the state space. The shaded regions indicate unreachable sets. The average of the distances from the grid points to the closest nodes (shown as dashed arrows) should converge to a finite number as the tree fills the reachable space.

$$g(\mathcal{T}_{s^i}) = -\Delta c(\mathcal{T}_{s^i})/\Delta n_{v,i}, \tag{12}$$

where $n_{v,i}$ is number of vertices in tree rooted at s^i.

Regarding the coverage of the set $I(q_1, \ldots, q_N)$ by S, one appropriate measure, which first appeared in the Monte Carlo literature and later has been used in the context of RRTs (Branicky et al. 2001), is dispersion. Dispersion is the radius of the largest empty ball whose center lies in $I(q_1, \ldots, q_N)$ which does not include a point in S. It is a measure of the largest region which is not covered. Therefore, we use normalized dispersion as a criteria for coverage of $I(q_1, \ldots, q_N)$:

$$\mu(S) = \sup_{x \in I(q_1,\ldots,q_N)} \min_{\bar{x} \in \bar{X}} d(x, \bar{x})/\mu(S)_{max}. \tag{13}$$

Here, $\bar{X}$ is a set of the nodes in $I(q_1, \ldots, q_N)$ and $\mu(S)_{max}$ is the largest possible dispersion for a given space. From a computational point of view, it is impractical to compute dispersion in high dimensions. Fortunately, there exist sequences of quasi-random numbers which have low dispersion. Accordingly, we use Halton (1960) sequence to generate the initial seed values. However, we cannot use a deterministic sequence to plant a new seed due to the existing nodes generated in $I(q_1, \ldots, q_N)$ as trees grow. The grid-based method introduced above can be used to approximate the normalized dispersion. We overlay grid points in $I(q_1, \ldots, q_N)$ and measure the distance between each grid point and existing nodes in the set to find the radius of the largest empty ball whose center is one of the grid points. A new seed is planted at the center of the largest empty ball.

5. Case Study: Luminescence Control in *Vibrio fischeri*

In this section, we show how the theory and algorithms developed in this paper can be applied to study the behavior of a specific genetic regulatory network. We consider for illustration the phenomenon of bioluminescence production in the marine bacterium *V. fischeri*, which is controlled by the transcriptional activation of the *lux* genes (James et al. 2000; Belta et al. 2001). The *lux* regulon is organized in two transcriptional units, O_L and O_R, separated by a regulatory region called the *lux box*, as shown in Figure 5. The leftward operon, O_L, contains the[1] *luxR* gene encoding protein LuxR, a transcriptional regulator of the system. The rightward operon O_R consists of seven genes *luxICDABEG*. The expression of the *luxI* gene results in the production of protein LuxI, which is required for endogenous production of *autoinducer*, Ai, a small membrane-permeant signal molecule. The other genes in O_R are involved in the production of luminescence. Finally, the autoinducer Ai binds to protein LuxR to form a complex C, which has an electronic affinity to the *lux box*. The transcription of both *luxICDABEG* and *luxR* is activated by the binding of C to the *lux box*, which is modeled using a continuous piecewise linear activation function (see Figure 6).

A nine-dimensional model for this network is presented in Belta et al. (2001). For illustrative purposes, we consider a simplification that is possible under the assumption that the dynamics of protein LuxI are fast (James et al. 2000). With this simplification, the system becomes three-dimensional ($N = 3$) with state $x = [x_1\, x_2\, x_3]^{\mathrm{T}}$, where x_1, x_2, and x_3 represent the concentrations of protein LuxR, complex C, and autonducer Ai, respectively. The main reason for choosing this simplified model is because the reduction in dimensionality allows us to include three-dimensional trajectories and reachability graphs, which would not be possible in higher-dimensional space. Examples of analysis in higher-dimensional space are presented in James et al. (2000) and Belta et al. (2004).

Two additive exogeneous inputs $\nu = [\nu_1\, \nu_2]^{\mathrm{T}}$ ($m = 2$) are present in the model. In the presence of a plasmid that produces LuxR independently, ν_1 is the rate of transcription of the plasmid, while ν_2 models an external source of autoinducer. More generally, they can represent stochastic uncertainty that may be inherent in the model. We will let $\mathcal{N}$ be a rectangular set given by $[0, \nu_{1,max}] \times [0, \nu_{2,max}]$.

Regarding the rectangular partition of the state space, we consider $n_1 = n_2 = n_3 = 3$. The thresholds θ_j^2, $j = 1, 2$ represent the values of x_2 for which the dynamics are changed due to different activation rates (see Figure 6), while $j = 0, 3$ represent physical lower and upper bounds. The other division points θ_j^i, $i = 1, 3$, $j = 0, 1, 2, 3$ were chosen so that the

1. We use italics (e.g., *luxR*) to indicate the genes and plain font to denote the protein expressed by the gene (e.g., LuxR).

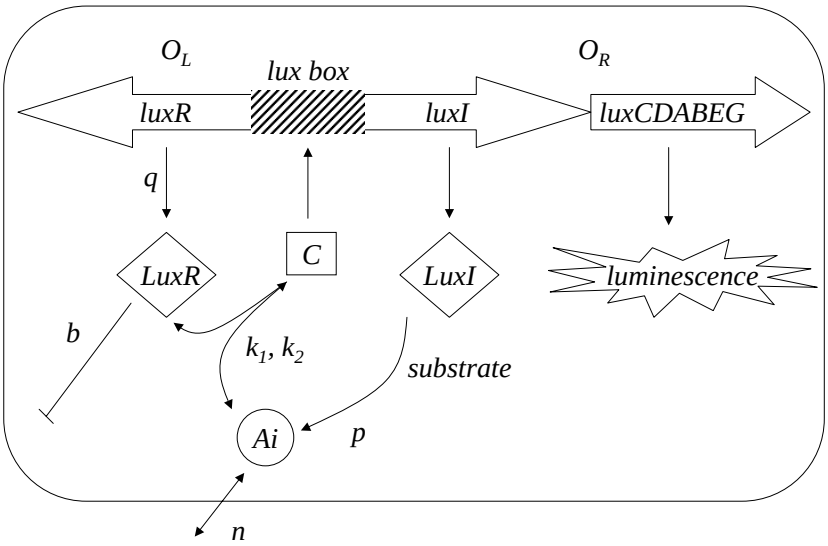

Fig. 5. Schematic representation of the genetic network regulating the luminescence production in the marine bacterium *V. fischeri*.

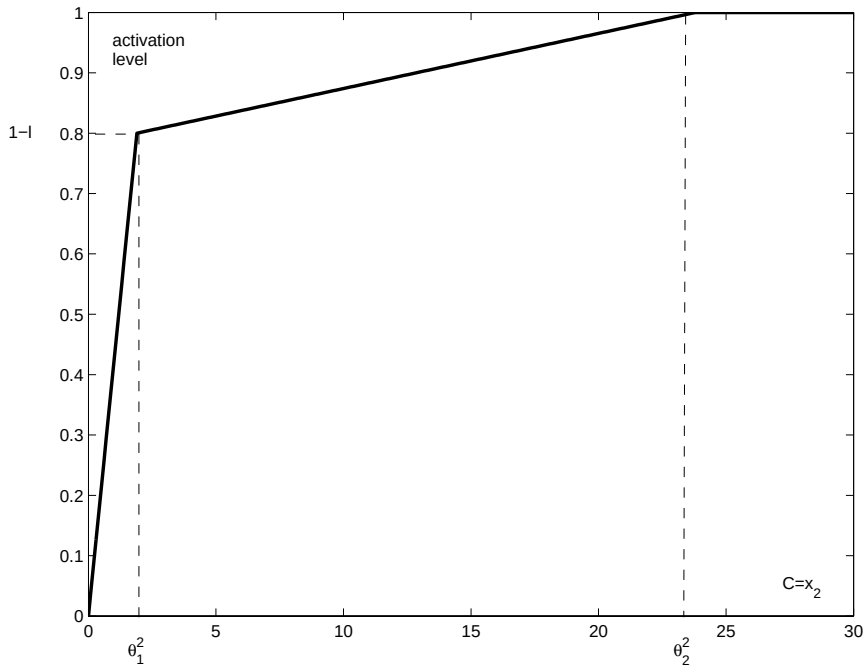

Fig. 6. Continuous piecewise linear activation function of the lux genes.

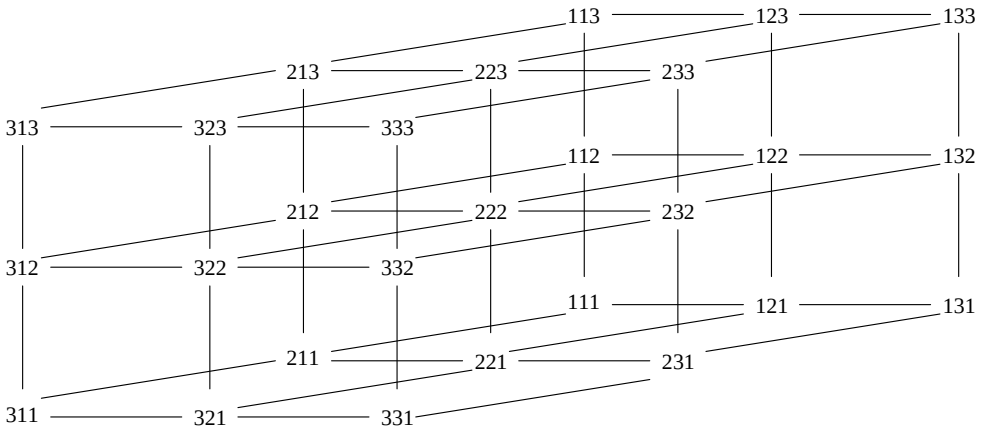

Fig. 7. The simple graph for the partitioning in eq. (14).

state space is divided into regions of interest, which could be thought of as "small", "medium", and "large" with respect to the corresponding specie concentrations. The numerical values of these constants are given by

$$\begin{array}{llll} \theta_1^0 = 0 & \theta_1^1 = 10 & \theta_1^2 = 50 & \theta_1^3 = 100 \\ \theta_2^0 = 0 & \theta_2^1 = 1.9 & \theta_2^2 = 23.8 & \theta_2^3 = 100 \\ \theta_3^0 = 0 & \theta_3^1 = 10 & \theta_3^2 = 50 & \theta_3^3 = 100 \\ \nu_{1,max} = 200 & \nu_{2,max} = 200 & & \end{array} \tag{14}$$

and by eq. (4)

$$a^{(q_1q_2q_3)} = (\theta_1^{q_1-1}, \theta_2^{q_2-1}, \theta_3^{q_3-1}),$$
$$b^{(q_1q_2q_3)} = (\theta_1^{q_1}, \theta_2^{q_2}, \theta_3^{q_3}),\ q_{1,2,3} \in \{1, 2, 3\}.$$

Following the notation introduced in Section 2, the system can be represented as the simple graph in Figure 7. The dynamics in each of the 27 rectangles $R_3(a^{(q_1q_2q_3)}, b^{(q_1q_2q_3)}) = I(q_1q_2q_3)$, with $q_{1,2,3} \in \{1, 2, 3\}$ are given by

$$\dot{x} = f^{(q_1q_2q_3)}(x) + B\nu \tag{15}$$

where

$$f^{(q_1q_2q_3)} = \begin{bmatrix} k_2x_2 - k_1x_1x_3 - bx_1 + qr^{(q_1q_2q_3)} \\ k_1x_1x_3 - k_2x_2 \\ k_2x_2 - k_1x_1x_3 - nx_3 + pr^{(q_1q_2q_3)} \end{bmatrix},$$

$$B = \begin{bmatrix} s & 0 \\ 0 & 0 \\ 0 & n \end{bmatrix} \tag{16}$$

and

$$r^{(q_1 1 q_3)} = \frac{(1-l)x_2}{\theta_2^1},\ r^{(q_1 2 q_3)} = 1 - \frac{l(\theta_2^2 - x_2)}{\theta_2^2 - \theta_2^1},\ r^{(q_1 3 q_3)} = 1,$$

for $q_1,\ q_2 = 1, 2, 3$ and $l = 0.2$ (see Figure 6). The dynamics are everywhere continuous, and therefore the vector fields on adjacent rectangles coincide on the common facet. The significance of the state variables and parameters is given as follows:

- x_1 = protein LuxR (ml^{-3});
- x_2 = complex C (ml^{-3});
- x_3 = autoinducer Ai (ml^{-3});
- k_1 = binding rate constant (30 $l^3m^{-1}t^{-1}$);
- k_2 = dissociation rate constant (10 t^{-1});
- n = diffusion constant (10 t^{-1});
- b = degradation constant for LuxR ($3t^{-1}$);
- p = formation of Ai due to lux gene activity ($30ml^{-3}t^{-1}$);
- q = formation of LuxR due to lux gene activity ($5ml^{-3}t^{-1}$);
- s = scaling constant (10 t^{-1}).

5.1. Obtaining The Infeasible Set $\mathcal{I}$ by Using MARP

Since the vector field given by eqs. (15) and (16) with $\nu_1 = \nu_2 = 0$ is continuous, we can simply apply Algorithm 1 to determine the orientation of the edges of the graph given in Figure 7. The result is given in Figure 8.

Assume the property set $\mathcal{P}$ is given by the target rectangle (222). This requires the execution of the repeat loop in Algorithm 2 four times:

- $\mathcal{R} := \mathcal{P} = \{(222)\}$
- $\mathcal{R} = \{(222), (223), (322), (212)\}$
- $\mathcal{R} = \{(222), (223), (322), (212), (213), (323), (312)\}$
- $\mathcal{R} = \{(222), (223), (322), (212), (213), (323), (312), (313)\}$.

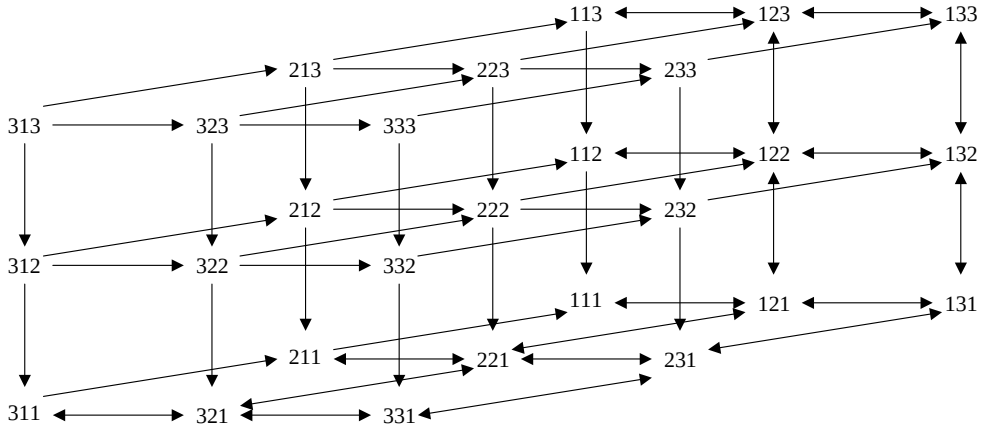

Fig. 8. The oriented graph obtained by applying Algorithm 1 to the simple graph in Figure 7. Note that $\mathcal{P} = \{(222)\}$.

The infeasible set $\mathcal{I}$ is the complement of $\mathcal{R}$, and it consists of the remaining $27 - 8 = 19$ rectangles. The biological significance of this result is that luminescence (which is described by relatively high values of complex x_2 and autoinducer x_3) can only be achieved if the initial level of autoinducer x_3 is high.

5.2. Obtaining A Feasible Set $\mathcal{F}$ Using The RRFT Algorithm

In this section, we consider the construction of feasible set $\mathcal{F}$ for the system with noise $v(t)$ using RRFT. In Figure 9, several sample trajectories illustrate the computation of candidate trajectories corresponding to the (111)–(121) transition, and the (121)–(111) transition. Each of these trajectories is the result of planting a tree with a different seed. Thus, the RRFT algorithm can be used to obtain sample trajectories for edges in the directed graph in Figure 8.

Recall that the property set $\mathcal{P}$ is the rectangle (222). We first consider the rectangle (111) to determine candidate points for the feasible set $\mathcal{F}$. Figure 10 shows the forest of trees where a solution trajectory is found. Ten initial seeds are generated and a forest starts to grow until a solution is found. The solution trajectory, the modes, and the transitions are shown in Figure 11. Figure 12 shows $c(\mathcal{T}_{s^i})$ and $g(\mathcal{T}_{s^i})$ for the trees. As thresholds for coverage criteria, we use $\bar{c} = 1 \times 10^{-6}$, $\bar{g} = 1 \times 10^{-6}$, and $\bar{\mu} = 0.1$. Four new seeds are generated beyond the initial seeds in this case. Figure 13 shows the coverage of the set (111) as new seeds are generated. First, ten initial seeds are generated using a Halton sequence and the next four seeds are generated by the grid-based method.

6. Conclusion

Hybrid systems are widely used in robotics to model the use of specific controllers and estimators in different regimes, switches based on contact mechanics of rolling or sliding, or to take into account the interactions and messaging in a multirobot team. Hybrid systems also arise naturally as models of genetic and metabolic networks. They capture the switching behavior that is observed in phenomena such as transcription, protein–protein interactions, and cell division and growth, and also provide global descriptions of biological systems described locally around operating points. More importantly, as shown in this paper, they allow local approximations that lend themselves to symbolic reasoning and more efficient computation.

In this paper, we develop computationally efficient techniques to analyze hybrid models of biomolecular networks by exploiting their specific structure. We have defined the framework of multi-affine rectangular hybrid systems, where the vector fields have product type nonlinearities to capture the dynamics of chemical reactions and the invariants are rectangular, because different behaviors emerge as a function of different ranges of concentrations of regulatory species. To prove qualitative properties of such systems, which are biologically significant, we developed the MARP algorithm and the RRFT algorithm. The MARP algorithm yields conservative results due to overapproximations of the underlying reachable sets. In contrast, the RRFT algorithm, which is based on trajectories generated from simulation, always underapproximates the reachable set. However, both algorithms provide complementary tools for analysis. This is illustrated by a case study, the phenomenon of luminescence production in the marine bacterium *V. fischeri*. While a low-dimensional case study was deliberately chosen to facilitate graphical illustration, the techniques here are applicable to very high-dimensional systems (Alur et al. 2002a). Future work is being directed towards developing tools for formal analysis of larger classes of hybrid systems, which could capture more complicated biochemical phenomena, and developing control laws for species in the network that can be directly controlled from outside the cell.

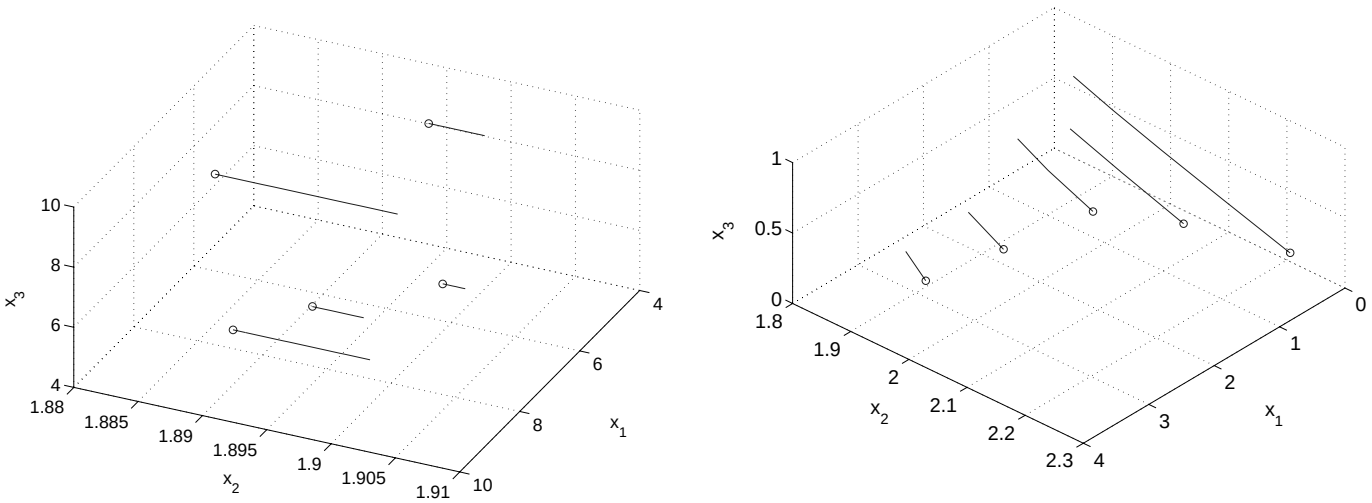

Fig. 9. Sample trajectories from (111) to (121) (left) and (121) to (111) (right).

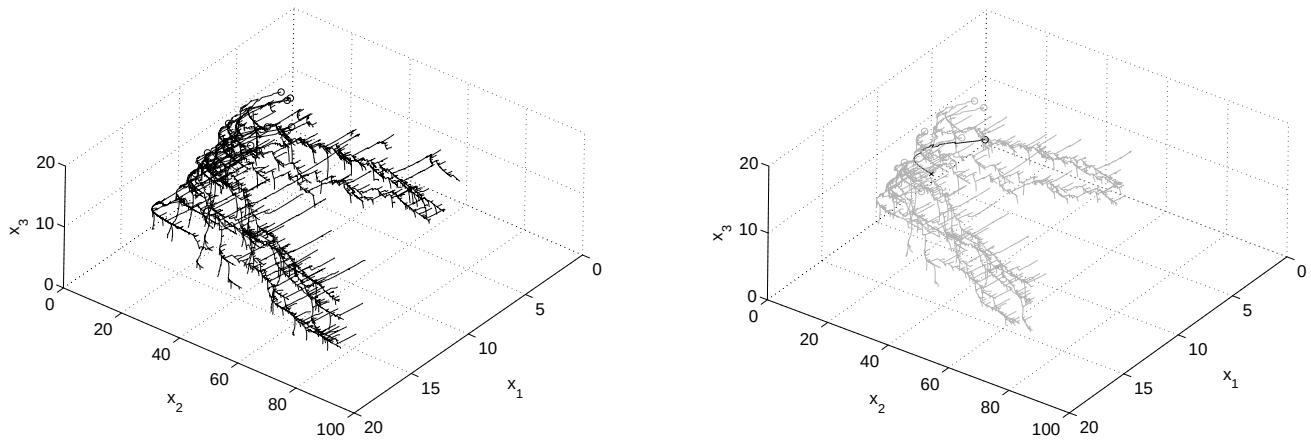

Fig. 10. The forest of trees computed by Algorithm 3 with initial conditions from the (111) rectangle with the goal of finding a trajectory that reaches (222). The forest is shown on the left. The trajectory found by the algorithm that reaches (222) is highlighted (shown dark) on the right.

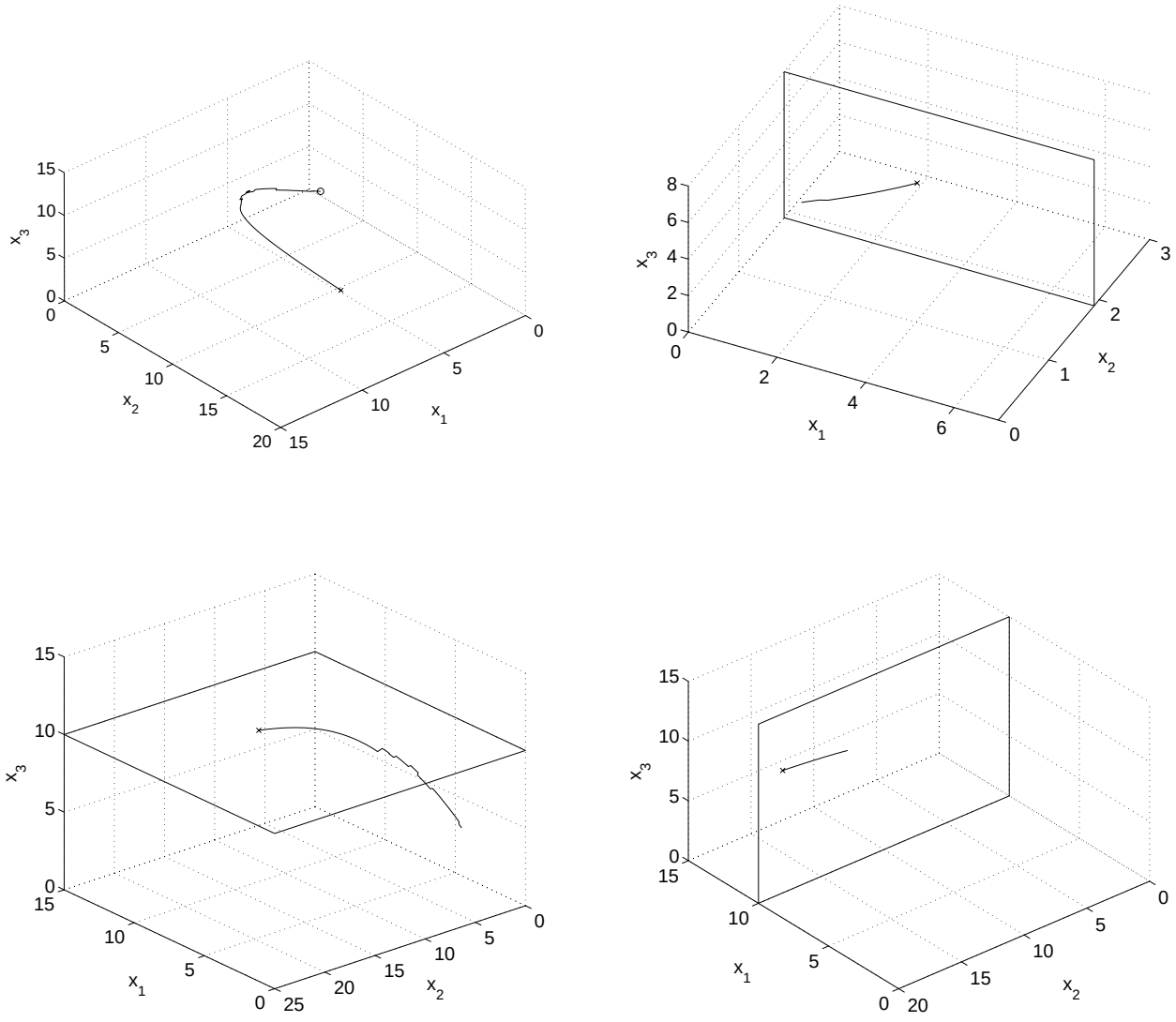

Fig. 11. A close up of the trajectory found by Algorithm 3, shown in Figure 10, showing the trajectory in different modes: top left, overall solution trajectory; top right, segment from (111) to (121) (note that the axis has been rotated to facilitate visualization); bottom left, (121)–(122); bottom right, (122)–(222).

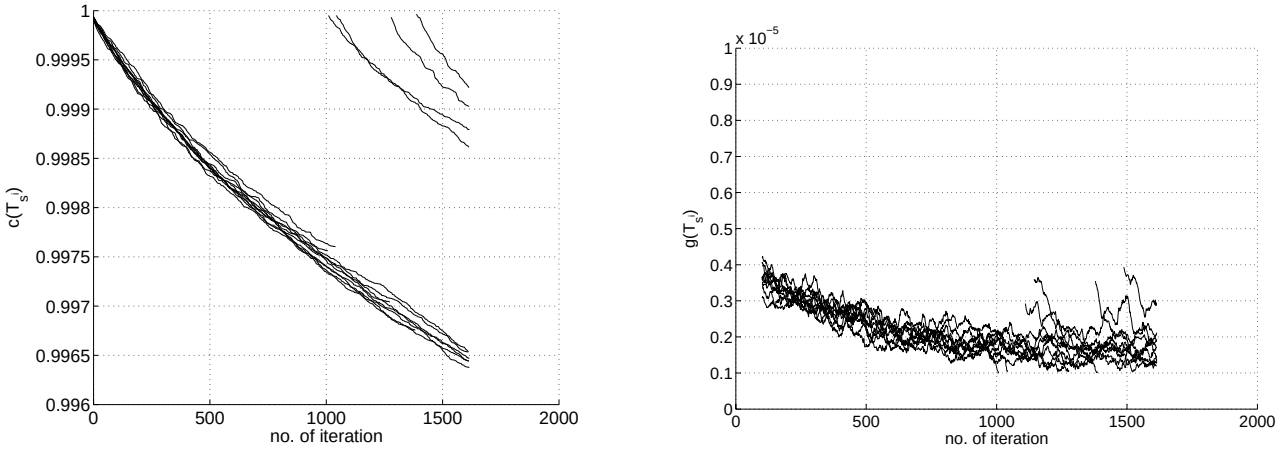

Fig. 12. Coverage of the trees. New trees are started at 1046, 1072, 1294, and 1379 iterations because the growth rate slows below the specified threshold ($\bar{g} = 1 \times 10^{-6}$)). Four new trees are generated until solution is discovered.

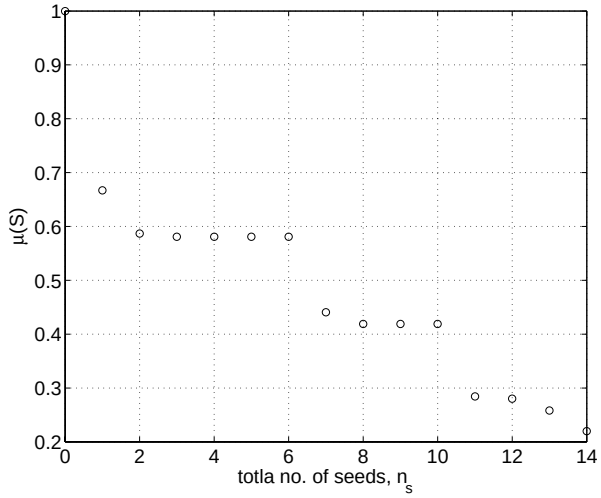

Fig. 13. The coverage improves ($\mu(S)$ decreases) as new trees are seeded.

References

Alur, R., Belta, C., Ivancic, F., Kumar, V., Mintz, M., Pappas, G., and Schug, J. 2001. Hybrid modeling and simulation of biomolecular networks. *Lecture Notes in Computer Science*, Vol. 2034. Springer-Verlag, Berlin, pp. 19–32.

Alur, R., Belta, C., Ivancic, F., Kumar, V., Rubin, H., Schug, J., Sokolsky, O., and Webb, J. 2002a. Visual programming for modeling and simulation of bioregulatory networks. *Proceedings of the International Conference on High Performance Computing*, Bangalore, India.

Alur, R., Belta, C., Kumar, V., Mintz, M., Pappas, G. J., Rubin, H., and Schug, J. 2002b. Modeling and analyzing biomolecular networks. *Computing in Science and Engineering* 4(1):20–30.

Alur, R., Dang, T., and Ivancic, F. 2002c. Reachability analysis of hybrid systems via predicate abstraction. *Proceedings of the 5th International Workshop on Hybrid Systems: Computation and Control*, Stanford, CA.

Amato, N. M., and Wu, Y. 1996. A randomized roadmap method for path and manipulation planning. *Proceedings of the IEEE International Conference on Robotics and Automation*, Minneapolis, MN, May, pp. 113–120.

Asarin, E., Dang, T., and Maler, O. 2001. d/dt: a tool for reachability analysis of continuous and hybrid systems. *Proceedings of the 5th IFAC Symposium on Nonlinear Control Systems*, St Petersburg, Russia.

Belta, C., Schug, J., Dang, T., Kumar, V., Pappas, G. J., Rubin, H., and Dunlap, P. V. 2001. Stability and reachability analysis of a hybrid model of luminescence in the marine bacterium *vibrio fischeri*. *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, FL.

Belta, C., Habets, L., and Kumar, V. 2002. Control of multi-affine systems on rectangles with applications to hybrid biomolecular networks. *Proceedings of the 41st IEEE Conference on Decision and Control*, Las Vegas, NV.

Belta, C., Finin, P., Habets, L. C. G. J. M., Halasz, A., Imielinksi, M., Kumar, V., and Rubin, H. 2004. Understanding the bacterial stringent response using reachability analysis of hybrid systems. *Hybrid Systems: Computation and Control, Proceedings of the 7th International Workshop*, Philadelphia, PA, March 25–27. *Lecture Notes in Computer Science* Vol. 2993. Springer-Verlag, Berlin, pp. 111–125.

Branicky, M., LaValle, S., Olson, K., and Yang, L. 2001. Quasi-randomized path planning. *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May, pp. 1481–1487.

Butchkarev, O., and Tripakis, S. 2000. Verification of hybrid systems with linear differential inclusions using elipsoidal roximations. *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science* Vol. 1790. Springer-Verlag, Berlin, pp. 73–88.

Chutinam, A., and Krogh, B. 2003. Computational techniques for hybrid system verification. *IEEE Transactions on Automatic Control* 48(1):64–75.

Das, A., Fierro, R., Ostrowski, J., Kumar, V., Spletzer, J., and Taylor, C. J. 2002. Vision-based formation control of multiple robots. *IEEE Transactions on Robotics and Automation* 18:813–825.

de Jong, H., Gouzé, J. L., Hernandez, C., Page, M., Sari, T., and Geiselmann, J. 2003. Hybrid modeling and simulation of genetic regulatory networks: a qualitative approach. *Hybrid Systems: Computation and Control, Proceedings of the 6th International Workshop*, Prague, Czech Republic, April 3–5. *Lecture Notes in Computer Science* Vol. 2623. Springer-Verlag, Berlin, pp. 267–282.

Esposito, J., and Kumar, V. 2001. Efficient dynamic simulation of robotic systems with hierarchy. *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May, pp. 2818–2823.

Esposito, J. M., Pappas, G. J., and Kumar, V. 2001a. Accurate event detection for simulating hybrid systems. *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science* Vol. 2034, M. D. Di Benedetto and A. Sangiovanni-Vincentelli, editors. Springer-Verlag, Berlin, pp. 204–217.

Esposito, J. M., Pappas, G. J., and Kumar, V. 2001b. Multi-agent hybrid system simulation. *IEEE Conference on Decision and Control*, Orlando, FL, December.

Frazzoli, E., Dahleh, M., and Feron, E. 2002. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Control, and Dynamics* 25(1):116–129.

Glass, L. 1975. Classification of biological networks by their qualitative dynamics. *Journal of Theoretical Biology* 54:85–107.

Halton, J. H. 1960. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerical Mathematics* 2:84–90.

Heinrich, R., and Schuster, S. 1996. *The Regulation of Cellular Systems*. Chapman and Hall, New York.

Henzinger, T. A., Kopke, P. W., Pujri, A., and Varaiya, P. 1995. What is decidable about hybrid automata? *Proceedings of the 27th Annual ACM Symposium on the Theory of Computation (STOC)*, New York, May, pp. 373–382.

Henzinger, T. A., Horowitz, B., Majumdar, R., and Wong-Toi, H. 2000. Beyond HyTech: hybrid systems analysis using interval numerical methods. *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science*, Vol. 1790. Springer-Verlag, Berlin, pp. 130–144.

James, S., Nilsson, P., James, G., Kjelleberg, S., and Fagerstrom, T. 2000. Luminescence control in the marine bacterium vibrio fischeri: an analysis of the dynamics of lux regulation. *Journal of Molecular Biology* 296:1127–1137.

Karatas, T., and Bullo, F. 2001. Randomized searches and nonlinear programming in trajectory planning. *Proceedings of the 40th IEEE Conference on Decision and Control*, Or-

lando, FL, December, pp. 5032–5037.

Kauffmann, S. A. 1969. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology* 22:437–467.

Kavraki, L. E., Svestka, P., Latombe, J. C., and Overmars, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration space. *IEEE Transactions on Robotics and Automation* 12:566–580.

Kepler, T. B., and Elston, T. C. 2001. Stochasticity in transcriptional regulation: origins, consequences, and mathematical representations. *Biophysical Journal* 81:3116–3136.

Kim, J., and Ostrowski, J. 2003. Motion planning of aerial robot using rapidly-exploring random trees with dynamic constraints. *Proceedings of the IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September 14–19.

Kim, J., Keller, J., and Kumar, V. 2003. Design and verification of controllers for airships. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, NV, October 27–31.

Lafferriere, G., Pappas, G. J., and Yovine, S. 1999. A new class of decidable hybrid systems. *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science*, Vol. 1590. Springer-Verlag, Berlin, pp. 137–151.

LaValle, S. M., and Kuffner, J. J. 2001a. Randomized kinodynamic planning. *International Journal of Robotics Research* 20(5):378–400.

LaValle, S. M., and Kuffner, J. J. 2001b. Rapidly exploring random trees: progress and prospects. *Algorithmic and Computational Robotics: New Directions*, B. Donald, K. Lynch, and D. Rus, editors, A. K. Peters, Wellesley, MA, pp. 293–308.

Lynch, N., and Krogh, B. H., editors. 2000. *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science*, Vol. 1790. Springer-Verlag, Berlin.

Mestl, T., Plathe, E., and Omholt, S. W. 1995. Periodic solutions in systems of piecewise-linear differential equations. *Dynamics and Stability of Systems* 10(2):179–193.

Mitchell, I., and Tomlin, C. 2000. Level set methods for computation in hybrid systems. *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science*, Vol. 1790, B. Krogh and N. Lynch, editors. Springer-Verlag, Berlin, pp. 310–323.

Pappas, G. J. 2003. Bisimilar linear systems. *Automatica* 39(12):2035–2047.

Park, D. M. R. 1980. *Concurrency and Automata on Infinite Sequences*, *Lecture Notes in Computer Science*, Vol. 104. Springer-Verlag, Berlin.

Park, T., and Barton, P. 1996. State event location in differential-algebraic models. *ACM Transactions on Modeling and Computer Simulation* 6(2):137–165.

Peschel, M., and Mende, W. 1986. *The Predator–Prey Model: Do We Live in a Volterra World?* Akademie Verlag, Berlin.

Savageau, M. A. 1969. Biochemical systems analysis: I. Some mathematical properties of the rate law for the component enzymatic reactions. *Journal of Theoretical Biology* 25:365–369.

Savageau, M. A., and Voit, E. O. 1987. Recasting nonlinear differential equations as s-systems: a canonical nonlinear form. *Mathematical Biosciences* 87:83–115.

Tabuada, P., and Pappas, G. J. 2003. Model checking LTL over controllable linear systems is decidable. *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science*, Vol. 2623. Springer-Verlag, Berlin.