Joint 48th IEEE Conference on Decision and Control and
28th Chinese Control Conference
Shanghai, P.R. China, December 16-18, 2009

A Probabilistic Approach for Control of a Stochastic System from LTL Specifications

M. Lahijanian, S. B. Andersson, and C. Belta
Mechanical Engineering, Boston University, Boston, MA 02215
{morteza,sanderss,cbelta}@bu.edu

Abstract— **We consider the problem of controlling a continuous-time linear stochastic system from a specification given as a Linear Temporal Logic (LTL) formula over a set of linear predicates in the state of the system. We propose a three-step solution. First, we define a polyhedral partition of the state space and a finite collection of controllers, represented as symbols, and construct a Markov Decision Process (MDP). Second, by using an algorithm resembling LTL model checking, we determine a run satisfying the formula in the corresponding Kripke structure. Third, we determine a sequence of control actions in the MDP that maximizes the probability of following the satisfying run. We present illustrative simulation results.**

I. Introduction

In control problems, "complex" models, such as systems of differential equations, are usually checked against "simple" specifications. Examples include the stability of an equilibrium, the invariance of a set, controllability, and observability. In formal analysis (verification), "rich" specifications such as languages and formulas of temporal logics, are checked against "simple" models of software programs and digital circuits, such as (finite) transition graphs. The most widespread specifications include safety (*i.e.,* something bad never happens) and liveness (*i.e.,* something good eventually happens). One of the current challenges in control theory is to bridge this gap and therefore allow for specifying the properties of complex systems in a rich language, with automatic verification and controller synthesis.

Most existing approaches are centered at the concept of abstraction, *i.e.,* a representation of a system with infinitely many states (such as a control system in continuous space and time) by one with finitely many states, called a symbolic, or abstract model. It has been shown that such abstractions can be constructed for systems ranging from simple timed, multi-rate, and rectangular automata (see [1] for a review) to linear systems [2]–[4] and to systems with polynomial dynamics [5], [6]. More complicated dynamics can also be dealt with through approximate abstractions [7], [8]. Recent works [9], [10] show that finite abstractions can also be constructed for particular classes of stochastic systems.

In this paper, we focus on continuous-time stochastic linear systems. We present a method to construct a feedback control strategy from a specification given as a Linear Temporal Logic (LTL) [11] formula over a set of linear predicates in the states of the system. Our approach consists of three steps. First, we construct an abstraction of the stochastic system in the form of a Markov Decision Process (MDP). This is achieved by partitioning the state space of the original system, choosing a finite set of controllers, and determining the transition probabilities of the controllers over the partition. Second, using the method developed in [12], we determine a sequence of states in the MDP satisfying the LTL specification. Finally, we determine a control strategy maximizing the probability of producing the satisfying run.

Stochastic systems are used as mathematical models in a wide variety of areas. For example, a realistic model for the motion of a robot should capture the noise in its actuators and sensors while a mathematical model of a biochemical network should capture the fluctuations in its kinetic parameters. "Rich" specifications for such systems (*e.g.,* "a robot should visit regions R_1 and R_2 infinitely often and never go to R_3" or "the concentration of a protein should never exceed value P") translate naturally to formulas of temporal logics. Recent results show that it is possible to control certain classes of non-stochastic dynamical systems from temporal logic specifications [12]–[15] and to drive a stochastic dynamical system between two regions [16]. There also exist probabilistic temporal logics such as probabilistic LTL, [17], probabilistic Computation Tree Logic (pCTL) [18], [19], linear inequality LTL (iLTL) [20], and the Continuous Stochastic Logic (CSL) [21]. A recent review of stochastic model checking based on both discrete and continuous time Markov chains can be found in [22].

Existing works focus primarily on Markov chains. The problem of constructing a control strategy for a partially observed Markov Decision Process (POMPD) from such a specification remains poorly understood. The main contribution of this work is to provide a preliminary and conservative solution to the open problem of controlling a stochastic system from a temporal logic specification. We focus on LTL so that we may take advantage of powerful existing results in the deterministic setting.

II. Problem statement and approach

In this work we consider the control of a stochastic linear system evolving in a full-dimensional polytope P in $\mathbb{R}^n$:

$$\begin{aligned} dx(t) &= (Ax(t) + Bu(t))\,dt + dw(t) \\ y(t) &= Cx(t) + v(t) \end{aligned} \tag{1}$$

where $x(\cdot) \in P \subset \mathbb{R}^n$, $u(\cdot) \in \mathbb{R}^m$, and $y(\cdot) \in \mathbb{R}^p$. The input and measurement noises are white noise processes.

The control inputs are limited to a set of control *symbols*, $\mathcal{S} = \{s_1, \ldots, s_{N_s}\}$. Each symbol $s = (u, \xi)$ is the combination of a control action, u, together with a termination condition, ξ or sequences of such pairs. The control action is in general an output feedback control law executed by the system until the termination condition becomes true. These control symbols are essentially a simplified motion description language (see [23], [24]).

The polytope P captures known physical bounds on the state of the system or a region that is required to be invariant to its trajectories. Note that we assume the distributions on the noise processes are such that the system remains inside P in the absence of any control action.

We are interested in properties of the system specified in terms of a set of linear predicates. Such propositions can capture a wide array of properties, including specifying regions of interest inside the physical environment or sensor space of a robot, or expression states for gene networks. Specifically, let $\Pi = \{\pi_i | i = 1, \ldots, n\}$ be a set of atomic propositions given as strict linear inequalities in $\mathbb{R}^n$. Each proposition π describes an open half-space of $\mathbb{R}^n$,

$$[\![\pi_i]\!] = \{x \in \mathbb{R}^N | c_i^T x + d_i < 0\} \tag{2}$$

where $[\![\pi]\!]$ denotes the set of all states satisfying the proposition π. Π then defines a collection of subpolytopes of P. We denote this collection as $Q = \{q_1, \ldots, q_{N_P}\}$.

In this work the properties are expressed in a temporal logic, specifically a fragment of the linear temporal logic known as LTL_{-X}. Informally, LTL_{-X} formulas are made of temporal operators, Boolean operators, and atomic propositions from Π connected in any "sensible way". Examples of temporal operators include $\Diamond$ ("eventually"), $\Box$ ("always"), and $\mathcal{U}$ ("until"). The Boolean operators are the usual $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction), $\Rightarrow$ (implication), and $\Leftrightarrow$ (equivalence). The atomic propositions capture properties of interest about a system, such as the set of linear predicates π_i from the set Π in (2). The semantics of an LTL formula containing atomic propositions from Π is given over infinite words over 2^Π (the power set of Π). For example, formulas $\Diamond\pi_2$, $\Diamond\Box\pi_3 \wedge \pi_4$ and $(\pi_1 \vee \pi_2)\mathcal{U}\pi_4$ are all true over the word $\Pi_1\Pi_2\Pi_3\Pi_4\ldots$, where $\Pi_1 = \{\pi_1\}$,$\Pi_2 = \{\pi_2, \pi_3\}$, $\Pi_i = \{\pi_3, \pi_4\}$, for all $i = 3, 4, \ldots$.

Inside this framework, we define the following problem.

Problem 1: Given a system of the form (1) *and an LTL_{-X} formula ϕ over* Π, *determine a set of initial states and a control policy to maximize the probability that the trajectory of the closed-loop control system satisfies ϕ while remaining inside P.*

To fully describe Problem 1, we need to define the satisfaction of an LTL_{-X} formula by a trajectory of (1). A formal definition is given in [12]; intuitively it can be defined as follows. As the trajectory evolves in time, a set of satisfied predicates is produced. This in turn produces a word in the power set of Π. Since the semantics of ϕ are expressed over such words, one can use these semantics to determine if this word satisfies the formula.

A deterministic version of this problem was solved in [12]. The approach in that setting consisted of three steps: first, the evolution of the system was abstracted to a finite-state transition system that captured motion between the regions defined by the propositions Π. Second, standard tools based on Büchi automata were used to produce runs of the transition system that satisfied the formula ϕ. Such runs can be understood as a sequence of transitions between the subpolytopes. Third, a feedback control strategy was determined to steer the system through the sequence of subpolytopes corresponding to a particular run of the transition system satisfying ϕ.

The stochastic generalization introduced here is challenging. One of the interesting features of (1) is that one cannot in general ensure any given trajectory will move through a desired sequence of subpolytopes. Thus, abstraction to a transition system as in the deterministic setting is not possible, nor is the production of a feedback strategy that will guarantee a particular trajectory. It should be noted, however, that failure to follow a particular run satisfying the formula ϕ does not imply that the formula itself is not satisfied.

In this paper we develop a conservative solution to Problem 1 consisting of three steps. First, we abstract the system (1) to an MDP evolving over the finite space defined by Q. This is done by considering the transitions (over Q) induced by each of the control symbols in $\mathcal{S}$. Note that due to the measurement process, the state of the system is unknown and thus in general the current subpolytope $q \in Q$ is not known either, requiring abstraction to a POMDP. For the purposes of this work, we make the simplifying assumption that q is known exactly even though the full system state x is not. While restrictive, such an assumption can be realized in some physical settings. For example, for $n = 2$, (1) may describe the kinematics and sensing of a planar robot with Q denoting regions of interest. A specific label could then be placed inside each region, allowing the robot to determine which subpolytope q it is currently on without determining its current state x.

Second, we determine a sequence of subpolytopes that satisfies ϕ. To do this, we take advantage of the first two steps of the deterministic solution in [12] and outlined above.

Finally, we determine a sequence of control symbols that maximizes the probability of moving through the sequence of subpolytopes produced by the second step. By construction, following this sequence ensures satisfaction of ϕ. As discussed above, however, failure to follow this sequence does *not* imply failure to satisfy ϕ. It is this fact that introduces a level of conservatism in our approach.

III. Abstraction and control

In the absence of noise, the work of [12] establishes how to assign linear feedback controllers in each polytope to ensure a transition in finite time to adjacent regions or that guarantee the polytope is invariant. This leads to a Kripke structure in which the states are the polytopes and the transitions capture our capability to design feedback controllers.

Such controllers can in some sense be viewed as a collection of symbols. They do not translate well into the stochastic setting, however, for two reasons. First, they are not robust with respect to actuator noise. For example, such controllers may steer the system either along the boundary between two regions or seek to exit a region near a point of intersection with multiple adjoining regions. In the absence of noise, the transitions will be perfect. In the presence of actuator noise, however, such motions have a high probability of causing an erroneous transition. Second, the deterministic laws are state feedback laws, each valid only in their corresponding polytopal region. In the stochastic setting the actual state is unknown and may differ from the predicted state.

Nevertheless, the abstraction provided by the deterministic approach can be used to find a word $w_a = w_a(1)\, w_a(2) \ldots w_a(k) \in 2^{\Pi}$, $k \geq 1$ satisfying the given specification ϕ. To find such a word, we simply use the tools of [12] for (1) where the noise terms have been set to 0. The word w_a can be viewed as a sequence of regions q_i to be traversed by the system. Traversing these regions in the given order ensures the system will satisfy the specification ϕ. In the remainder of this section, we assume an appropriate word has been found and describe the abstraction and control steps to maximize the probability of producing this word with (1).

A. Abstraction

As discussed in Sec. II, we assume the system is given a collection $\mathcal{S}$ of control symbols. While these symbols may have been designed to achieve particular actions, such as moving through a particular face of a polytope or converging to a location in the state space, moving in a fixed direction for a fixed time or performing a random "tumble" to choose a new heading direction, in our approach each symbol is viewed simply as an available controller, without any consideration of its intended effect. Rather, we capture its *actual* effect through the use of an MDP as described below.

To develop an abstraction of the stochastic system, we use the fact that Q captures all the regions of interest with respect to specifications ϕ. The execution of a symbol from $\mathcal{S}$ defines a natural discrete-time evolution for the system: a choice of control symbol is applied at time k to the system until the associated interrupt is triggered. At that point, a measurement over Q is obtained and time is incremented to $k+1$. As mentioned in Sec. II, this measurement is assumed to be perfect so that at the completion of each control symbol the current subpolytope is known.

To create the MDP representing the evolution of this system, we determine for each control symbol $s \in \mathcal{S}$ the transition probability that the symbol will terminate in subpolytope q_j given that it started in subpolytope q_i, denoted as $p(q_j|q_i, s)$. During execution of the symbol the system may actually move through several regions before termination of s. Because our goal is to follow *exactly* a specified sequence of regions, we must exclude such multiple transitions inside each time step. We therefore define an additional state, q_{N_p+1}, such that $p(q_{N_p+1}|q_i, s)$ represents the probability of passing through multiple regions before terminating, regardless of the final subpolytope.

While for some simple control actions and noise models the transition probabilities can be calculated exactly from the Fokker-Planck equation, in general exact analytic solutions are not feasible and the probabilities must be found through approximate methods. One powerful class of tools are the Monte Carlo or particle filter methods [25]. It is precisely such an approach we adopt in the example described in Sec. IV. Once determined, these probabilities for each control symbol are arranged into a Markov matrix $M_{tp}(s_\alpha)$, $\alpha = 1, \ldots, N_s$., where the ij^{th} element of $M_{tp}(s_\alpha)$ is the probability of transitioning to state i given that the system is currently on state j and that symbol s_α is executed. The MDP is then given by $\{Q, M_{tp}(1), \ldots, M_{tp}(N_s)\}$.

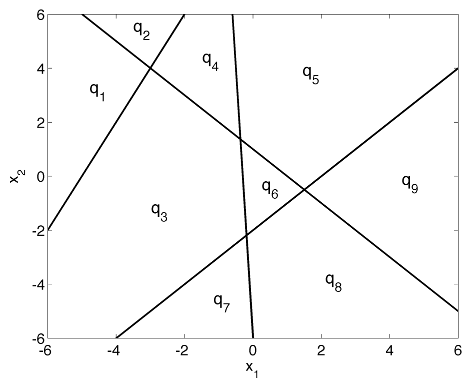

Fig. 1. A partitioning of the state space into polytopal regions.

In creating this abstraction, we assume that the transitions generated by the control symbols in $\mathcal{S}$ are Markovian in nature. In general this is not the case. Consider, for example, the regions illustrated in Fig. 1 and two motion scenarios: a system moving in q_2 near the border with q_1 and a system moving in q_3 near the border with q_1. In the first case, due to noise the system may randomly transition from q_2 to q_1. Once in q_1, noise is likely to push the system back to q_2. Similarly, in the second case noise is likely to cause a transition from q_3 into q_1 and then back into q_3. Thus, when in q_1, the transition probabilities depend on where the system came from, imparting memory to the system. This non-Markovian effect is influenced by the size of the noise and the geometry of the polytopal regions. The Markovian assumption can be enforced, at least approximately, through appropriate design of the control symbols. The resulting abstraction is an approximation of the original system and its accuracy would need to be determined. This can be captured, for example, through a notion of distance as determined by an approximate bisimulation [26].

B. Control

Given an MDP and a word w_a that satisfies the specification ϕ, our goal is to design a control policy to maximize the probability that (1) produces the word w_a. A control policy is a sequence of symbols from $\mathcal{S}$ and is denoted $\Gamma = \{s_1, \ldots, s_r\}$ where $r \geq 1$ is the length of the policy. To determine the optimal policy, we express this optimization in terms of a cost function involving the probability of producing the word and then use dynamic programming to optimize it over the set of policies.

Under our approach, we must follow the sequence in w_a *exactly*. As a result, the optimization reduces to a one-stage look-ahead in which at any step i, the optimal control symbol is the one with the maximum probability of transitioning the system to the next element of w_a. This forms a feedback control policy (over Q) as follows. Let i denote the current time step and k the current index into w_a so that the system is currently in the subpolytope denoted by $w_a(k)$. The control symbol that maximizes the probability of transitioning the system to the subpolytope denoted by $w_a(k+1)$ is selected and executed. At the completion of the symbol, time is incremented to $i+1$ and the current value of q is measured. If this is $w_a(k+1)$ then k is incremented while if it is equal to $w_a(k)$ then the index k is left unchanged. If the current state is neither $w_a(k)$ or $w_a(k+1)$ then the run is terminated because the system has failed to produce the desired word.

IV. Example

To illustrate our approach, we considered a two dimensional case. The system dynamics were given by (1) where

$$A = \begin{bmatrix} 0.2 & -0.3 \\ 0.5 & -0.5 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

and $x \in P$ where P is specified as the intersection of eight closed half spaces, defined by: $a_1 = [-1\ 0]^T$, $b_1 = -5$, $a_2 = [1\ 0]^T$, $b_2 = -7$, $a_3 = [0\ -1]^T$, $b_3 = -3$, $a_4 = [0\ 1]^T$, $b_4 = -6$, $a_5 = [-3\ -5]^T$, $b_5 = -15$, $a_6 = [1\ -1]^T$, $b_6 = -7$, $a_7 = [-1\ 2.5]^T$, $b_7 = -15$, $a_8 = [-2\ 2.5]^T$, $b_8 = -17.5$. The input and output noise processes were taken to be zero mean, Gaussian white noise with covariances $\mathcal{Q}$ and $\mathcal{R}$ respectively, where $\mathcal{Q} = \mathcal{R} = \text{diag}(9, 9)$. Note that these noise values were chosen such that the standard deviations were on the same order as the dimensions of P. These levels of noise are much larger than would be expected in, for example, a robotic system.

The set Π was defined using ten predicates of the form (2) where $c_1 = [0\ 1]^T$, $d_1 = 0$, $c_2 = [1 - 1]^T$, $d_2 = 0$, $c_3 = [4\ 1]^T$, $d_3 = 12$, $c_4 = [4 - 7]^T$, $d_4 = 34$, $c_5 = [-2 - 1]^T$, $d_5 = 4$, $c_6 = [-1 - 12]^T$, $d_6 = 31$, $c_7 = [-1 - 1]^T$, $d_7 = 11$, $c_8 = [1\ 0]^T$, $d_8 = -3$, $c_9 = [0\ -1]^T$, $d_9 = -1.5$, $c_{10} = [-6\ -4.5]^T$, $d_{10} = -12$. These values yield 33 feasible full-dimensional subpolytopes in P, illustrated in Fig. 2.

A set of control symbols were designed based on the intuition that to move to an adjacent polytope, the system should steer to the center of the shared facet and then proceed to the center of the new polytope. It is important to keep in mind the discussion in Sec.III-A, namely that the intended effect (move to center of facet then to center of polytope) is irrelevant to the approach. It is only the resulting transition probabilities in the MDP structure that are important.

The control actions had the form of an affine state estimate feedback law given by $u_{x_d}(t) = -L(\hat{x} - x_d) - Ax_d$, where $\hat{x}$ is the state estimate given by a Kalman-Bucy filter and x_d is a desired point to move to. The feedback gain was set to

$$L = \begin{bmatrix} 20.2 & -0.30 \\ 0.50 & 20.0 \end{bmatrix}.$$

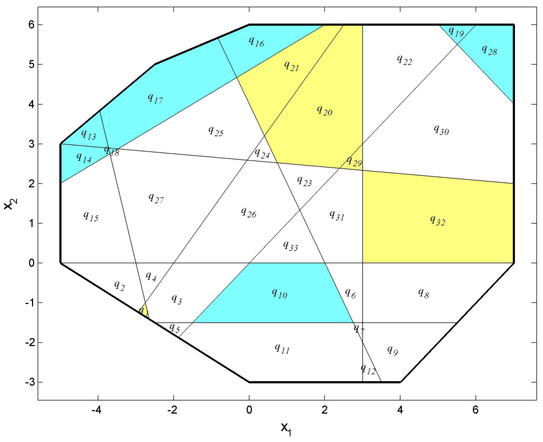

Fig. 2. The region P and the subpolytopes defined by Π. Subpolytopes in blue denote regions to be avoided while those in yellow denote target areas as designated by the policy ϕ specified in (4).

We defined four interrupts. In words, these were: the system exits the current subpolytope, the system exits the union of the previous and current polytope, the system has converged to the desired point, or the action has executed for a sufficiently long time. The second condition was added to allow the system to switch back and forth between two adjacent polytopes if desired. These four interrupts can be expressed as follows:

Interrupt	Triggering condition
ξ_{exit}	$x(t) \notin q_o$, the initial polytope
ξ_{exit2}	$x(t) \notin q_{prev} \bigcup q_{curr}$
ξ_{x_d}	$\|\hat{x}(t) - x_d\| \leq \epsilon$, $\epsilon > 0$
ξ_T	$t \geq T$

Based on the intuitive description for moving between subpolytopes, we created a collection of basic symbols

$$s_{f_i} = \left(u_{x_{i,j}}, \left(\xi_{x_{i,j}} \vee \xi_{exit} \vee \xi_T\right)\right), \tag{3a}$$

$$s_{cp} = \left(u_{x_{cp}}, \left(\xi_{x_{cp}} \vee \xi_{exit2} \vee \xi_T\right)\right), \tag{3b}$$

$$s_r = \left(u_{x_{cp}}, \left(\xi_{exit2} \vee \xi_T\right)\right), \tag{3c}$$

$$s_I = \left(u_{x_{cp}}, \left(\xi_{exit} \vee \xi_T\right)\right). \tag{3d}$$

Here x_i denotes the center of the i^{th} shared face and x_{cp} denotes the center of the current polytope. The first symbol is designed to steer the system to one of the shared faces and terminates either when the state estimate converges to within ϵ of the center of the face, when the state exists the current polytope, or when T seconds have elapsed. The second symbol is designed to steer the system to the center of the current polytope and terminates either when the state estimate converges to the center, when the system enters a subpolytope which is not either the current or previous one (defined when the symbol is executed), or after T seconds. Note that it allows multiple transitions between the current polytope and the previous one since such transitions in essence do not effect the word generated by the system under the assumption that sequences of repeated values in w_a can be collapsed to a single copy of that value. The third symbol is similar to the second but lacks the convergence termination condition. It serves as a "randomizer", causing the system to lose any memory of which polytope it was previously on. It is this symbol which helps to enforce the Markov condition needed for the abstraction. The final symbol is designed to make the current polytope invariant.

The set Π defines 54 shared faces. The basic symbols were used to create 55 control symbols $s_i = (s_{f_i}, s_{cp}, s_r), i = 1$ to 54 together with the invariance controller s_I.

A. *Creating the abstraction*

As noted in Sec. III, the collection of states in Q was augmented with a state capturing multiple transitions, yielding 34 states. For each of the 55 symbols, 2000 particles were initialized on each polytope. Each particle was evolved according to (1) under control of the symbol until the termination condition was met. The ending subpolytope for each particle was then recorded. The simulations were performed in Matlab running under Lunix on an Intel Xeon Quad-Core 2.66 GHz processor equipped with 3 GB of RAM. The 55 transition matrices (each of dimension 34×34) took approximately 21 hours to create.

As an illustrative example, in Fig. 3 we show the transition matrix corresponding to the symbol designed to move from q_1 to q_2. The entry $[M_{tp}]_{ij}$ denotes the probability of ending on region q_j given that the system was initially on region q_i and that the system evolved with this control symbol.

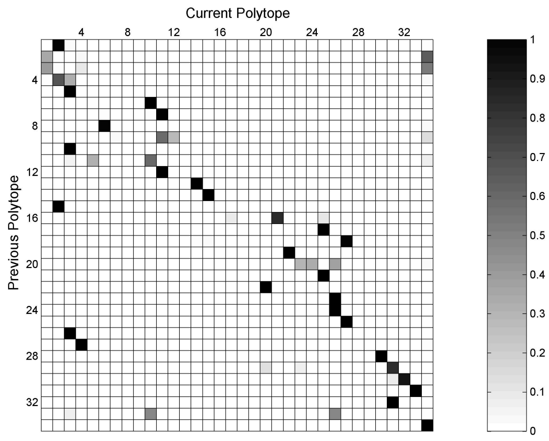

Fig. 3. Transition probabilities for the symbol designed for $q_1 \to q_2$.

B. *The specification and results*

We chose an LTL_{-X} formula inspired from robot motion planning. It involved visiting a sequence of three regions while avoiding three "obstacle" regions. The regions to be visited were, in order:

$$r_1 = q_1,\ r_2 = \bigcup_{i\in\{20,21,29\}} q_i,\ r_3 = q_{32}.$$

The obstacles were represented by the polyhedral regions

$$o_1 = \bigcup_{i\in\{13,14,16,17,18\}} q_i,\ o_2 = \bigcup_{i\in\{19,28\}} q_i,\ o_3 = q_{10}.$$

These regions are illustrated in Fig. 2. The corresponding LTL$_{-X}$ formula can be written as

$$\phi = \Diamond(r_1 \wedge \Diamond(r_2 \wedge \Diamond r_3)) \wedge \Box\neg(o_1 \vee o_2 \vee o_3). \tag{4}$$

For each initial state, we used the deterministic tools of [12] to produce a word to follow. A control policy was then found by selecting the sequence of control symbols which maximized the one-step transition probabilities. The probability of following that word was then calculated by multiplying those probabilities. In Fig. 4 we show, for every region in Q, the probability of satisfying the specification ϕ. To verify our abstraction, we also determined the probabilities through Monte Carlo simulation. These results are also shown in Fig. 4. Differences in the two results arise from three primary sources: (1) the finite number of particles used in the Monte Carlo simulations, (2) a mismatch between the distributions over each region q_i used to initialize the Monte Carlo simulations and the actual distributions during a run, and (3) non-Markovian behavior in the transitions.

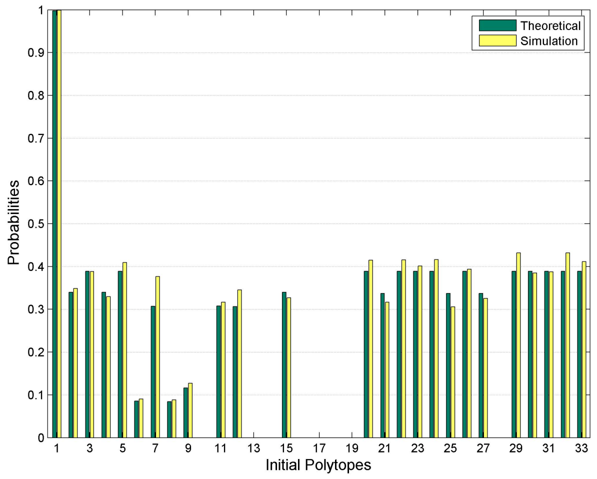

Fig. 4. Theoretical and Monte Carlo-based probabilities of a run following a given word satisfying ϕ..

In Fig. 5 we show three sample runs starting from the subpolytope q_{12}. The word to follow (highlighted in green on the figure) from this regions was found to be

$$q_{12}q_{11}q_5q_3q_1q_3q_{26}q_{23}q_{29}q_{22}q_{30}(q_{32})$$

where (q_{32}) means to repeat this element infinitely often. The actual evolution of the system is shown in blue while the estimate of the trajectory is shown in green. Of the three runs, one successfully followed the word while two failed.

V. Conclusions and future works

In this paper, we discuss the problem of controlling a stochastic dynamical system from a specification given as

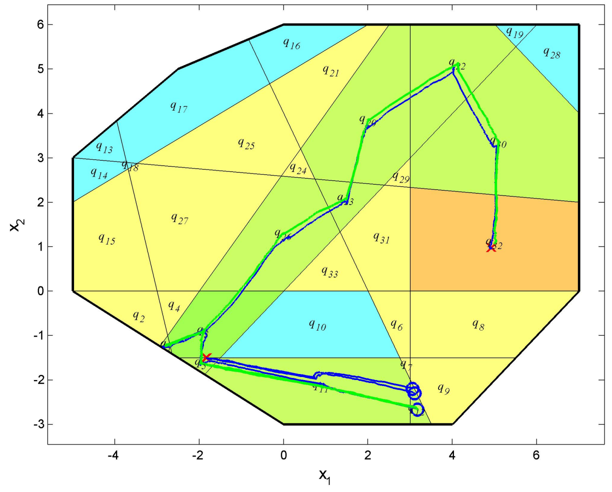

Fig. 5. Three sample trajectories of the system starting from q_{12}. The word to follow was $q_{12}q_{11}q_5q_3q_1q_3q_{26}q_{23}q_{29}q_{22}q_{30}(q_{32})$ and is shown as green regions. Both the actual (blue) and estimated (green) trajectories are shown. The initial positions are indicated with blue circles and the final positions with red crosses.

a temporal logic formula over a set of state-predicates. We focus on linear systems and LTL. This work can be seen as an extension of our previous results on temporal logic control of linear systems, where we showed that a particular choice of controllers reduces the problem to controlling an abstraction in the form of a finite transition system from an LTL formula. In the stochastic framework considered here, the abstraction becomes in general a POMDP. Since controlling a POMDP from a (probabilistic) temporal logic specifications is an open problem, here we present an approach based on following the solution of the deterministic problem with maximum probability and under the assumption of perfect observations over the MDP. Directions of future research include using approximations to POMDPs combined with dynamic programming to determine a policy that maximizes the probability of satisfying the specification directly and the development of game theoretic approaches for POMDP control.

Acknolwedgements

This work was supported in part by NSF under grants CAREER 0447721 and 0834260 at Boston University.

References

[1] R. Alur, T. Henzinger, G. Lafferriere, and G. J. Pappas, "Discrete abstractions of hybrid Sys.," *Proc. IEEE*, vol. 88, no. 2, pp. 971–984, 2000.

[2] P. Tabuada and G. Pappas, "Linear time logic control of discrete-time linear Sys.," *IEEE Trans. Autom. Control*, vol. 51, no. 12, pp. 1862–1877, 2006.

[3] G. J. Pappas, "Bisimilar linear Sys.," *Automatica*, vol. 39, no. 12, pp. 2035–2047, 2003.

[4] B. Yordanov and C. Belta, "Parameter synthesis for piecewise affine Sys. from temporal logic specifications," in *Hybrid Sys.: Comp. and Control: 11th Int. Workshop*, ser. Lec. Notes in Comp. Sci., M. Egerstedt and B. Mishra, Eds. Springer, 2008, vol. 4981, pp. 542–555.

[5] A. Tiwari and G. Khanna, "Series of abstractions for hybrid automata," in *Hybrid Sys.: Comp. and Control: 5th Int. Workshop*, ser. Lec. Notes in Comp. Sci.. Springer, 2002, vol. 2289, pp. 425–438.

[6] M. Kloetzer and C. Belta, "Reachability analysis of multi-affine Sys.," in *Hybrid Sys.: Comp. and Control: 9th Int. Workshop*, ser. Lec. Notes in Comp. Sci., J. Hespanha and A. Tiwari, Eds. Springer, 2006, vol. 3927, pp. 348 – 362.

[7] P. Tabuada, "Symbolic control of linear Sys. based on symbolic subSys.," *IEEE Trans. Autom. Control*, vol. 51, no. 6, pp. 1003–1013, 2006.

[8] A. Girard, "Approximately bisimilar finite abstractions of stable linear Sys.," in *Hybrid Sys.: Comp. and Control: 10th Int. Workshop*, ser. Lec. Notes in Comp. Sci., A. Bemporad, A. Bicchi, and G. Buttazzo, Eds. Springer, 2007, vol. 4416, pp. 231–244.

[9] A. Abate, A. D'Innocenzo, M. Di Benedetto, and S. Sastry, "Markov set-chains as abstractions of stochastic hybrid Sys.," in *Hybrid Sys.: Comp. and Control*, ser. Lec. Notes in Comp. Sci., M. Egerstedt and B. Misra, Eds. Springer, 2008, vol. 4981, pp. 1–15.

[10] A. D'Innocenzo, A. Abate, M. D. Benedetto, and S. Sastry, "Approximate abstractions of discrete-time controlled stochastic hybrid Sys.," in *Proc. IEEE Conf. on Decision and Control*, 2008, pp. 221–226.

[11] E. M. M. Clarke, D. Peled, and O. Grumberg, *Model checking*. MIT Press, 1999.

[12] M. Kloetzer and C. Belta, "A fully automated framework for control of linear Sys. from temporal logic specifications," *IEEE Trans. Autom. Control*, vol. 53, no. 1, pp. 287–297, 2008.

[13] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas, "Hybrid controllers for path planning: a temporal logic approach," in *Proc. IEEE Conf. on Decision and Control*, 2005, pp. 4885–4890.

[14] H. Kress-Gazit, D. Conner, H. Choset, A. Rizzi, and G. Pappas, "Courteous cars," *IEEE Robotics and Automation Magazine*, vol. 15, pp. 30–38, March 2008.

[15] P. Tabuada and G. Pappas, "Model checking LTL over controllable linear Sys. is decidable," in *Hybrid Sys.: Comp. and Control: 6th Int. Workshop*, ser. Lec. Notes in Comp. Sci., O. Maler and A. Pnueli, Eds. Springer, 2003, vol. 2623, pp. 498–513.

[16] S. B. Andersson and D. Hristu-Varsakelis, "Symbolic feedback control for navigation," *IEEE Trans. Autom. Control*, vol. 51, no. 6, pp. 926–937, 2006.

[17] C. Baier, "On algorithmic verification methods for probabilistic Sys.," Ph.D. dissertation, Univ. of Mannheim, Germany, 1998.

[18] A. Bianco and L. de Alfaro, "Model checking of probabilistic and nondeterministic Sys.," in *FST TCS 95: Foundations of Software Technology and Theoretical Comp. Sci.*, ser. Lec. Notes in Comp. Sci.. Springer, 1995, vol. 1026, pp. 499–513.

[19] A. Aziz, V. Singhal, F. Balarin, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "It usually works: the temporal logic of stochastic Sys.," in *Comp. Aided Verification*, ser. Lec. Notes in Comp. Sci.. Springer, 1995, vol. 939, pp. 155–165.

[20] Y. Kwon and G. Agha, "iLTLChecker: a probabilistic model checker for multiple DTMCs," in *Proc. IEEE Int. Conf. on the Quantitative Evaluation of Sys.*, 2005.

[21] C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen, "Model-checking algorithms for continuous-time Markov chains," *IEEE Trans. Softw. Eng.*, vol. 29, no. 6, pp. 524–541, June 2003.

[22] M. Kwiatkowska, G. Norman, and D. Parker, "Stochastic model checking," in *Formal Methods for the Design of Comp., Communication and Software Sys.: Performance Evaluation*, ser. Lec. Notes in Comp. Sci., M. Bernardo and J. Hillston, Eds. Springer, 2007, vol. 4486, pp. 220–270.

[23] R. W. Brockett, "Formal languages for motion description and map making," in *Robotics*. American Mathematical Society, 1990, pp. 181–193.

[24] V. Manikonda, P. S. Krishnaprasad, and J. Hendler, "Languages, behaviors, hybrid architectures, and motion control," in *Mathematical Control Theory*, J. Baillieul, Ed. Springer, 1998, pp. 199–226.

[25] O. Cappe, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential monte carlo," *Proc. IEEE*, vol. 95, no. 5, pp. 899–924, May 2007.

[26] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous Sys.," *IEEE Trans. Autom. Control*, vol. 52, no. 5, pp. 782–798, May 2007.