Robust Control of Uncertain Markov Decision Processes with Temporal Logic Specifications

Eric M. Wolff, Ufuk Topcu, and Richard M. Murray

***Abstract*—We present a method for designing a robust control policy for an uncertain system subject to temporal logic specifications. The system is modeled as a finite Markov Decision Process (MDP) whose transition probabilities are not exactly known but are known to belong to a given uncertainty set. A robust control policy is generated for the MDP that maximizes the worst-case probability of satisfying the specification over all transition probabilities in this uncertainty set. To this end, we use a procedure from probabilistic model checking to combine the system model with an automaton representing the specification. This new MDP is then transformed into an equivalent form that satisfies assumptions for stochastic shortest path dynamic programming. A robust version of dynamic programming solves for a ϵ-suboptimal robust control policy with time complexity $O(\log 1/\epsilon)$ times that for the non-robust case.**

I. Introduction

As the level of autonomous operation expected of robots, vehicles, and other cyberphysical systems increases, there is a growing need for formal methods for precisely specifying and verifying system properties. As autonomous systems often operate in uncertain environments, it is also important that system performance is robust to environmental disturbances. Furthermore, system models are only approximations of reality, which makes robustness to modeling errors desirable.

A promising approach for specifying and verifying system properties is the use of temporal logics such as linear temporal logic (LTL). LTL provides a natural framework to specify desired properties such as response (if A, then B), liveness (always eventually A), safety (always not B), stability (eventually always A), and priority (first A, then B, then C).

We model the system as a Markov Decision Process (MDP). MDPs provide a general framework for modeling non-determinism and probabilistic behaviors that are present in many real-world systems. MDPs are also amenable to formal verification techniques for temporal logic properties [5], that can be alternatively used to create control policies. These techniques generate a control policy for the MDP that maximizes the probability of satisfying a given LTL specification. However, these techniques assume that the state transition probabilities of the MDP are known exactly, which is often unrealistic. We relax this assumption by allowing the transition probabilities of the MDP to lie in uncertainty sets. We generate a control policy that maximizes the worst-case probability of a run of the system satisfying a given LTL specification over all admissible transition probabilities in the uncertainty set.

Authors are with the Department of Control and Dynamical Systems, California Institute of Technology, Pasadena, CA, USA. The corresponding author is ewolff@caltech.edu

Considering uncertainty in the system model is important to capture unmodeled dynamics and parametric uncertainty, as real systems are only approximated by mathematical models. Additionally, while we consider discrete systems in this paper, we are motivated by controlling continuous stochastic systems so that they satisfy temporal logic specifications. Constructive techniques for finite, discrete abstractions of continuous stochastic systems exist (see [1], [3]), but exact abstraction is generally difficult. Even if exact finite-state abstraction techniques are available for a dynamical system model, the resulting MDP abstraction will only represent the real system to the extent that the dynamical system model does. Moreover, if abstraction techniques approximate the dynamical system model, the MDP abstraction will be a further approximation of the real system.

Robustness of control policies for MDPs with respect to uncertain transition probabilities has been studied in the contexts of operations research, formal verification, and hybrid systems. Our approach most closely follows that of Nilim and El Ghaoui [19], who consider general uncertainty models and discounted rewards, but not temporal logic specifications. Related work includes [4], [12], [20]. Formal verification of temporal logic specifications is well developed for MDPs with exact transition matrices [5], [10] and standard software tools exist [14]. Work in verification of uncertain MDPs primarily considers simple interval uncertainty models for the transition probabilities [7], [8], [21], which our work includes as a special case. Recent work in hybrid systems that creates control policies for stochastic systems [11], [15] does not consider robustness. Robustness of non-probabilistic discrete systems to disturbances is explored in [17].

The main contribution of this paper is creating an optimal robust control policy π^* that maximizes the worst-case probability of satisfying an LTL specification for a system represented as a finite labeled MDP with transition matrices in an uncertainty set $\mathcal{P}$. A control policy π is a mapping from each MDP state to an allowable action. The uncertainty set $\mathcal{P}$ can be non-convex and includes interval uncertainty sets as a special case. This freedom allows more statistically accurate and less conservative results than interval uncertainty sets.

Preliminary definitions and the formal problem statement are given in Sections II and III respectively. We combine the system MDP with an automaton representation of the LTL specification to form a product MDP that represents system trajectories that satisfy both the system dynamics and the specification in Section IV. We use dynamic programming (see Section V) to create a robust control policy that maximizes the worst-case probability of satisfying the specification over all transition matrices in the uncertainty set. This approach can be viewed as a game between the system and its environment, where the controller selects actions to maximize the probability of satisfying the specification, while the environment selects transition matrices to minimize the probability of satisfying the specification. An example of our approach is presented in Section VII. We conclude with suggestions for future work in Section VIII.

II. Preliminaries

We now give definitions for both the system modeling and task specification formalisms, Markov Decision Processes (MDPs) and linear temporal logic (LTL), respectively. Throughout, (in)equality is component-wise for vectors and matrices. Also, $\mathbf{1}$ is a vector of ones of appropriate dimension. An *atomic proposition* is a statement that is *True* or *False*.

A. System Model

Definition 1 (labeled finite MDP)**.** A *labeled finite MDP* $\mathcal{M}$ is the tuple $\mathcal{M} = (S, A, P, s_0, AP, L)$, where S is a finite set of states, A is a finite set of actions, $P : S \times A \times S \to [0, 1]$ is the transition probability function, s_0 is the initial state, AP is a finite set of atomic propositions, and $L : S \to 2^{AP}$ is a labeling function. Let $A(s)$ denote the set of available actions at state s. Let $\sum_{s' \in S} P(s, a, s') = 1$ if $a \in A(s)$ and $P(s, a, s') = 0$ otherwise.

We assume, for notational convenience, that the available actions $A(s)$ are the same for every $s \in S$. We use P^a_{ij} as shorthand for the transition probability from state i to state j when using action a. We call $P^a \in \mathbb{R}^{n \times n}$ a *transition matrix*, where the (i, j)-th entry of P^a is P^a_{ij}. Where it is clear from context, we refer to the row vector P^a_i as p.

Definition 2. A *control policy* for an MDP $\mathcal{M}$ is a sequence $\pi = \{\mu_0, \mu_1, \ldots\}$, where $\mu_k : S \to A$ such that $\mu_k(s) \in A(s)$ for state $s \in S$ and $k = 0, 1, \ldots$. A control policy is *stationary* if $\pi = \{\mu, \mu, \ldots\}$. Let Π be the set of all control policies and Π_s be the set of all stationary control policies.

A *run* of the MDP is an infinite sequence of its states, $\sigma = s_0 s_1 s_2 \ldots$ where $s_i \in S$ is the state of the system at index i and $P(s_i, a, s_{i+1}) > 0$ for some $a \in A(s_i)$. A run is induced by a control policy.

Uncertainty Model: To model uncertainty in the system model, we specify uncertainty sets for the transition matrices.

Definition 3 (Uncertain labeled finite MDP)**.** Let the transition matrix uncertainty set be defined as $\mathcal{P}$ where every $P \in \mathcal{P}$ is a transition matrix. An *uncertain labeled finite MDP* $\mathcal{M} = (S, A, \mathcal{P}, s_0, AP, L)$ is a family of labeled finite MDPs such that for every $P \in \mathcal{P}$, $\mathcal{M}' = (S, A, P, s_0, AP, L)$ is a MDP.

Let $\mathcal{P}^a_s$ be the uncertainty set corresponding to state $s \in S$ and action $a \in A(s)$.

Definition 4. An *environment policy* for an (uncertain) MDP $\mathcal{M}$ is a sequence $\tau = \{\nu_0, \nu_1, \ldots\}$, where $\nu_k : S \times A \to P$ such that $\nu_k(s, a) \in \mathcal{P}^a_s$ for state $s \in S$, action $a \in A(s)$ and $k = 0, 1, \ldots$. An environment policy is *stationary* if $\tau = \{\nu, \nu, \ldots\}$. Let $\mathcal{T}$ be the set of all environment policies and $\mathcal{T}_s$ be the set of all stationary environment policies.

A *run* of the uncertain MDP is an infinite sequence of its states, $\sigma = s_0 s_1 s_2 \ldots$ where $s_i \in S$ is the state of the system at index i and $P(s_i, a, s_{i+1}) > 0$ for some $a \in A(s_i)$ and $P \in \mathcal{P}^a_{s_i}$. A run is induced by an environment policy and a control policy.

We associate a reward with each state-action pair in $\mathcal{M}$ through the function $r(s, a) : S \times A \to \mathbb{R}$. This reward is incurred at each stage k over the horizon of length N, where the total expected reward is

$$V^{\pi\tau}(s) := \lim_{N \to \infty} \mathbb{E}_{\pi\tau} \left[\sum_{k=0}^{N-1} r(s_k, \mu_k(s_k)) \mid s_0 = s \right],$$

and the expectation $\mathbb{E}_{\pi\tau}$ depends on both the control and environment policies.

The optimal worst-case total expected reward starting from state $s \in S$ is

$$V^*(s) := \max_{\pi \in \Pi} \min_{\tau \in \mathcal{T}} V^{\pi\tau}(s). \qquad (1)$$

B. Task Specification Language

We use linear temporal logic (LTL) to concisely and unambiguously specify the desired system behavior. We only touch on key aspects of LTL for our problem and defer the reader to [5] for details.

LTL is built up from (a) a set of atomic propositions, (b) the logic connectives: negation ($\neg$), disjunction ($\vee$), conjunction ($\wedge$) and material implication ($\Longrightarrow$), and (c) the temporal modal operators: next ($\bigcirc$), always ($\square$), eventually ($\diamond$) and until ($\mathcal{U}$). An LTL formula is defined inductively as follows: (1) any atomic proposition p is an LTL formula; and (2) given LTL formulas φ and ψ, $\neg\varphi$, $\varphi \vee \psi$, $\bigcirc\varphi$ and $\varphi \, \mathcal{U} \, \psi$ are also LTL formulas.

The semantics of LTL is defined inductively as follows: (a) For an atomic proposition p, $s_i \vDash p$ if and only if (iff) $s_i \Vdash p$; (b) $s_i \vDash \neg\varphi$ iff $s_i \nvDash \varphi$; (c) $s_i \vDash \varphi \vee \psi$ iff $s_i \vDash \varphi$ or $s_i \vDash \psi$; (d) $s_i \vDash \bigcirc\varphi$ iff $s_{i+1} \vDash \varphi$; and (e) $s_i \vDash \varphi \, \mathcal{U} \, \psi$ iff there exists $j \geq i$ such that $s_j \vDash \psi$ and $\forall k \in [i, j), s_k \vDash \varphi$. Based on this definition, $\bigcirc\varphi$ holds at position s_i iff φ holds at the next state s_{i+1}, $\square\varphi$ holds at position i iff φ holds at every position in σ starting at position i, and $\diamond\varphi$ holds at position i iff φ holds at some position $j \geq i$ in σ.

Definition 5. A run $\sigma = s_0 s_1 s_2 \ldots$ *satisfies* φ, denoted by $\sigma \vDash \varphi$, if $s_0 \vDash \varphi$.

Remark 1. LTL allows specification of guarantee, liveness, and response properties beyond the safety and stability properties typically used in controls and hybrid systems.

It will be useful to represent an LTL formula as a deterministic Rabin automaton, which can always be done [5].

Definition 6. A *deterministic Rabin automaton* is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, Acc)$ where Q is a finite set of states, Σ is an alphabet, $\delta : Q \times \Sigma \rightarrow Q$ is the transition function, $q_0 \in Q$ is the initial state, and accepting state pairs $Acc \subseteq 2^Q \times 2^Q$.

Let Σ^ω be the set of infinite words over Σ. A run $\sigma = \mathcal{A}_0\mathcal{A}_1\mathcal{A}_2 \ldots \in \Sigma^\omega$ denotes an infinite sequence $q_0q_1q_2\ldots$ of states in $\mathcal{A}$ such that $q_{i+1} \in \delta(q_i, \mathcal{A}_i)$ for $i \geq 0$. The run $q_0q_1q_2\ldots$ is accepting if there exists a pair $(L, K) \in Acc$ and an $n \geq 0$, such that for all $m \geq n$ we have $q_k \notin L$ and there exist infinitely many k such that $q_k \in K$.

Intuitively, a run is accepted by a deterministic Rabin automaton if the set of states L is visited finitely often and the set K is visited infinitely often.

III. Problem Statement

We now provide a formal statement of the main problem of the paper and an overview of our approach.

Definition 7. Let $\mathcal{M}$ be an uncertain MDP with initial state s_0 and atomic propositions AP. Let φ be an LTL formula over AP. Then, $\mathbb{P}^{\pi,\tau}(s_0 \vDash \varphi)$ is the *expected satisfaction probability* of φ by $\mathcal{M}$ under control policy π and environment policy τ.

Definition 8. An *optimal robust control policy* π^* for uncertain MDP $\mathcal{M}$ is

$$\pi^* = \arg\max_{\pi \in \Pi} \min_{\tau \in \mathcal{T}} \mathbb{P}^{\pi,\tau}(s_0 \vDash \varphi).$$

Problem 1. Given an uncertain labeled finite MDP $\mathcal{M}$ and an LTL formula φ over AP, create an optimal robust control policy π^*.

We solve Problem 1 by first creating the product MDP $\mathcal{M}_p$, which contains only valid system trajectories that also satisfy the LTL specification. We modify $\mathcal{M}_p$ so that all policies for it are proper, and thus it satisfies stochastic shortest path assumptions. Maximizing the worst-case probability of satisfying the specification is equivalent to a creating a control policy that maximizes the worst-case probability of reaching a certain set of states in $\mathcal{M}_p$. We solve for this policy using robust dynamic programming. Finally, we map the robust control policy back to $\mathcal{M}$. This procedure is detailed in the rest of the paper.

IV. The product MDP

In this section, we create a product MDP $\mathcal{M}_p$ which contains behaviors that satisfy both the system MDP $\mathcal{M}$ and the LTL specification φ. We transform $\mathcal{M}_p$ into an equivalent form $\mathcal{M}_{ssp}$ where all stationary control policies are proper in preparation for computing optimal robust control policy in Section VI.

A. Forming the product MDP

The product MDP $\mathcal{M}_p$ restricts behaviors to those that satisfy both the system transitions and the determinstic Rabin automaton $\mathcal{A}_\varphi$ representing the LTL formula φ.

Definition 9. For labeled finite MDP $\mathcal{M} = (S, A, P, s_0, AP, L)$ and deterministic Rabin automaton $\mathcal{A}_\varphi = (Q, 2^{AP}, \delta, q_0, Acc)$, the *product MDP* $\mathcal{M}_p = (S_p, A, P_p, s_{0p}, Q, L_p)$ with $S_p = S \times Q$,

- $P_p((s,q), \alpha, (s', q')) = \begin{cases} P(s, \alpha, s') & \text{if } q' = \delta(q, L(s')) \\ 0 & \text{otherwise,} \end{cases}$
- $s_{0p} = (s_0, q)$ such that $q = \delta(q_0, L(s_0))$,
- $L_p((s,q)) = \{q\}$.

The accepting product state pairs $Acc_p = \{(L_1^p, K_1^p), \ldots, (L_k^p, K_k^p)\}$ are lifted directly from Acc. Formally, for every $(L_i, K_i) \in Acc$, state $(s, q) \in S_p$ is in L_i^p if $q \in L_i$, and $(s,q) \in K_i^p$ if $q \in K_i$.

There is a one-to-one correspondence between the paths on $\mathcal{M}_p$ and $\mathcal{M}$, which induces a one-to-one correspondence for policies on $\mathcal{M}_p$ and $\mathcal{M}$. Given a policy $\pi^p = \{\mu_0^p, \mu_1^p, \ldots\}$ on $\mathcal{M}_p$, one can induce a policy $\pi = \{\mu_0, \mu_1, \ldots\}$ on $\mathcal{M}$ by setting $\mu_i(s_i) = \mu_i^p((s_i, q_i))$ for every stage $i = 0, 1, \ldots$. This policy always exists since $\mathcal{M}_p$ and $\mathcal{M}$ have the same action set A. If π^p is stationary, then π is finite-memory [5].

B. Reachability in the product MDP

We now show how to use the product MDP $\mathcal{M}_p$ to determine a robust control policy that maximizes the worst-case probability that a given LTL specification is satisfied. Given a control and environment policy, the probability of satisfying an LTL formula is equivalent to the probability of reaching an *accepting maximal end component* [5]. We call this probability the *reachability probability*. Informally, accepting maximal end components are sets of states that the system can remain in forever and where the acceptance condition of the deterministic Rabin automaton is satisfied. The following definitions follow [5].

Definition 10. A *sub-MDP* of a MDP is a pair of states and action sets (C, D) where: (1) $C \subseteq S$ is non-empty and the map $D : C \rightarrow 2^A$ is a function such that $D(s) \subseteq A(s)$ is non-empty for all states $s \in C$ and (2) $s \in C$ and $a \in D(s)$ implies $Post(s, a) = \{t \in S | P_{st}^a > 0\} \subseteq C$.

Definition 11. An *end component* is a sub-MDP (C, D) such that the digraph $G_{(C,D)}$ induced by (C, D) is strongly connected.

An end component (C, D) is *maximal* if there is no end component (C', D') such that $(C, D) \neq (C', D')$ and $C \subseteq C'$ and $D(s) \subseteq D'(s)$ for all $s \in C$. Furthermore, (C, D) is *accepting* for the deterministic Rabin automaton $\mathcal{A}$ if for some $(L, K) \in Acc$, $L \notin C$ and $K \in C$.

Given the accepting maximal end components of $\mathcal{M}_p$, one can determine a control policy that maximizes the worst-case probability of reaching an accepting maximal end component from the initial state. Without considering transition probability uncertainty, a non-robust policy can be computed using either linear or dynamic programming methods [5].

From the preceding discussion, it is clear that the LTL formula satisfaction probability depends on the connectivity of the product MDP $\mathcal{M}_p$. Thus, we will require that the uncertainty sets for the transition matrices of the system MDP $\mathcal{M}$ do not change this connectivity. Let F^a denote the nominal transition matrix for action a.

Assumption 1. $F^a_{ij} = 0$ if and only if $P^a_{ij} = 0$ for all $P^a \in \mathcal{P}^a$ and for all $i, j \in S$.

Assumption 1 says that if a nominal transition is zero (non-zero) if and only if it is zero (non-zero) for all transition matrices in the uncertainty set.

C. Stochastic shortest path form of product MDP

We now transform the product MDP $\mathcal{M}_p$ into an equivalent form $\mathcal{M}_{ssp}$ where all stationary control policies μ are proper. Note that $\mathcal{M}_p$ and $\mathcal{M}_{ssp}$ are equivalent only in terms of the probability of reaching an accepting maximal end component—both the states and the transition probabilities may change.

Let MDP $\mathcal{M}$ have a finite set of states $S = \{1, 2, \dots, n, t\}$ and actions $a \in A(s)$ for all $s \in S$. Let t be a special terminal state, which is absorbing ($P^a_{tt} = 1$) and incurs zero reward ($r(t, a) = 0$) for all $a \in A(t)$ and all $P \in \mathcal{P}^a_t$ [6].

Definition 12. A stationary control policy μ is *proper* if, under that policy, there is positive probability that the terminal state will be reached after at most n transitions, regardless of the initial state and transition matrices, that is, if

$$\rho_{\mu\tau} := \max_{s=1,\dots,n} \max_{\tau \in \mathcal{T}} \mathbb{P}_{\mu\tau}(s_n \neq t | s_0 = s) < 1. \qquad (2)$$

In the remainder of this section, we use the simplified notation $\mathcal{M}_p = (S, A, P)$ to describe the states, actions, and transition matrices of the product MDP. We refer to a state (s, q) of $\mathcal{M}_p$ as s when clear from context.

Partition the states S of $\mathcal{M}_p$ into three disjoint sets, B, S_0, and S_r. Let set B be the union of all accepting maximal end components in $\mathcal{M}_p$. By definition, every state $s \in B$ has reachability probability of 1. Let S_0 be the set of states that have zero probability of reaching B. Set S_0 can be computed efficiently by graph algorithms [5]. Finally, let set $S_r = S - (B \cup S_0)$ contain states not in an accepting maximal end component but with non-zero maximum reachability probability. It is easy to see that B, S_0, and S_r form a partition of S.

Algorithm 1 Appending the terminal state

```
Require: $\mathcal{M}_p = (S, A, P)$ and $S_r, S_0, B$
    $S := S \cup \{t\}$ and $A(t) := \{u\}$ and $r(t, u) := 0$;
    $A(s) := A(s) \cup \{u\}$ and $P^u_{st} := 1$ for all $s \in B \cup S_0$.
```

In Algorithm 1, we augment S with a terminal state t which is absorbing and incurs zero reward. Algorithm 1 does not change the probability of reaching an accepting maximal end component for any state $s \in S$ under any control and environment policies.

We eliminate the maximal end components in S_r and replace them with new states in Algorithm 2. This procedure is from Section 3.3 of [10], where it is proven (Theorem 3.8 in [10]) that the reachability probability is unchanged by this procedure. The intuition behind this result is that one can move between any two states r and s in a maximal end component in S_r with probability one.

Algorithm 2 End component elimination (de Alfaro [10])

```
Require: MDP $\mathcal{M}_p = (S, A, P)$ and $S_r, S_0, B$
Ensure: MDP $\mathcal{M}_{ssp} = (\hat{S}, \hat{A}, \hat{P})$
    $\{(C_1, D_1), \dots, (C_k, D_k)\}$ max end components in $S_r$
    $\hat{S}_0 := S_0$ and $\hat{B} := B$;
    $\hat{S} := S \cup \{\hat{s}_1, \dots, \hat{s}_k\} - \cup_{i=1}^k C_i$;
    $\hat{S}_r := S_r \cup \{\hat{s}_1, \dots, \hat{s}_k\} - \cup_{i=1}^k C_i$;
    $\hat{A}(s) := \{(s, a) \mid a \in A(s)\}$ for $s \in S - \cup_{i=1}^k C_i$;
    $\hat{A}(\hat{s}_i) := \{(s, a) \mid s \in C_i \wedge a \in A(s) - D(s)\}$ for $1 \le i \le k$;
    For $s \in \hat{S}, t \in S - \cup_{i=1}^k C_i$ and $(u, a) \in \hat{A}(s)$, $\hat{P}^{(u,a)}_{st} := P^a_{ut}$
    and $\hat{P}^{(u,a)}_{s\hat{s}_i} := \sum_{t \in C_i} P^a_{ut}$.
```

After applying Algorithms 1 and 2, we call the resulting MDP $\mathcal{M}_{ssp}$. Note that $\hat{S}_r$, $\hat{B}$, and $\hat{S}_0$ form a disjoint partition of $\hat{S}$. All stationary control policies for $\mathcal{M}_{ssp}$ are proper, i.e., they will almost surely reach the terminal state t.

Theorem 1. *All stationary control policies for $\mathcal{M}_{ssp}$ are proper.*

Proof: Suppose instead that there exists a stationary control policy μ such that the system starting in state $s_0 \in \hat{S}_r$ has zero probability of having reached the terminal state t after n stages. This implies that under μ there is zero probability of reaching any state $s \in \hat{B} \cup \hat{S}_0$ from $s_0 \in \hat{S}_r$. Then, under policy μ, there exists a set $U \subseteq \hat{S}_r$ such that if state $s_k \in U$ for some finite integer k, then $s_k \in U$ for all k. Let $U' \subseteq U$ be the largest set where each state is visited infinitely often. Set U' is an end component in $\hat{S}_r$, which is a contradiction. Note that one only needs to consider $s_0 \in \hat{S}_r$, as all $s \in \hat{B} \cup \hat{S}_0$ deterministically transition to t. ■

MDP $\mathcal{M}_{ssp}$ is equivalent in terms of reachability probabilities to the original product MDP $\mathcal{M}_p$ and all stationary control policies are proper.

V. Robust Dynamic Programming

We now prove results on robust dynamic programming that will be used in Section VI to compute the optimal robust control policy.

A. Dynamic Programming

We require control policies to be proper, i.e. they almost surely reach the terminal state t for all transition matrices in the uncertainty set (see Section IV-C).

Assumption 2. All stationary control policies are proper.

Remark 2. This assumption implies that the terminal state will eventually be reached under any stationary policy. This assumption allows us to make statements regarding convergence rates. While this assumption is usually a rather strong condition, it is not restrictive for Problem 1 (see Theorem 1).

In preparation for the main result of this section, we give the following classical theorem [18].

Theorem 2 (Contraction Mapping Theorem). *Let (M, d) be a complete metric space and let $f : M \to M$ be a contraction, i.e., there is a real number β, $0 \leq \beta < 1$, such that $d(f(x), f(y)) \leq \beta d(x, y)$ for all x and y in M. Then there exists a unique point x^* in M such that $f(x^*) = x^*$. Additionally, if x is any point in M, then $\lim_{k\to\infty} f^k(x) = x^*$, where f^k is the composition of f with itself k times.*

We now define mappings that play an important role in the rest of this section. The value $V(s)$ is the total expected reward starting at state $s \in S$. The shorthand V represents the value function for all $s \in S\backslash t$ and can be considered a vector in $\mathbb{R}^N$. Since the reward is zero at the terminal state t, we do not include it. The T and $T_{\mu\nu}$ operators are mappings from $\mathbb{R}^n$ to $\mathbb{R}^n$. For each state $s \in S\backslash t$, define the s-th component of TV and $T_{\mu\nu}V$ respectively as

$$(TV)(s) \ := \ \max_{a\in A(s)} [r(s,a) + \min_{p\in\mathcal{P}_s^a} p^T V], \qquad (3)$$

$$(T_{\mu\nu}V)(s) \ := \ r(s,\mu(s)) + \nu(s,\mu(s))^T V. \qquad (4)$$

In the following two lemmas, we show that these mappings are monotonic and contractive. We prove these for (3); the proofs for (4) follow by limiting the actions and transition probabilities at each state s to $\mu(s)$ and $\nu(s,\mu(s))$ respectively. T^k is the composition of T with itself k times.

Lemma 1 (Monotonicity). *For any vectors $u, v \in \mathbb{R}^n$, such that $u \leq v$, we have that $T^k u \leq T^k v$ for $k = 1, 2, \ldots$.*

Proof: Immediate from (3) since $\mathcal{P}_s^a$ is in the probability simplex. ■

Definition 13. The *weighted maximum norm* $\| \cdot \|_w$ of a vector $u \in \mathbb{R}^n$ is defined by $\| u \|_w = \max_{i=1,\ldots,n} |u(i)|/w(i)$ where vector $w \in \mathbb{R}^n$ and $w > 0$.

Lemma 2 (Contraction). *If all stationary control policies are proper, then there exists a vector $w > 0$ and a scalar $\gamma \in [0, 1)$ such that $\| Tu - Tv \|_w \leq \gamma \| u - v \|_w$ for all $u, v \in \mathbb{R}^n$.*

Proof: See Appendix. ■

We now prove the main result of this section. We remind the reader that the function $V^* : S \to \mathbb{R}$ (equivalently a vector in $\mathbb{R}^n$), defined in (1), is the optimal worst-case total expected reward starting from state $s \in S$.

Theorem 3 (Robust Dynamic Programming). *Under the assumption that all stationary control policies μ are proper for a finite MDP $\mathcal{M}$ with transition matrices in the uncertainty set $\mathcal{P}^a$ for $a \in A$, the following statements hold.*

(a) The optimal worst-case value function V^* is the unique fixed-point of T,

$$V^* = TV^*. \qquad (5)$$

(b) The optimal worst-case value function V^* is given by,

$$V^* = \lim_{k\to\infty} T^k V, \qquad (6)$$

for all $V \in \mathbb{R}^n$. This limit is unique.

(c) A stationary control policy μ and a stationary environment policy ν are optimal if and only if

$$T_{\mu\nu} V^* = TV^*. \qquad (7)$$

Proof: Parts (a) and (b) follow immediately from Theorem 2 and Lemma 2.

Part (c): First assume that $T_{\mu\nu}V^* = TV^*$. Then, $T_{\mu\nu}V^* = TV^* = V^*$ from (5) and $V^{\mu\nu} = V^*$ from the uniqueness of the fixed-point. Thus, μ and ν are optimal policies. Now assume that μ and ν are optimal policies so that $V^{\mu\nu} = V^*$. Then, $T_{\mu\nu}V^* = T_{\mu\nu}V^{\mu\nu} = V^{\mu\nu} = V^*$. ■

Corollary 1. *Given the optimal worst-case value function V^*, the optimal control actions a^* satisfy*

$$a^*(s) \in \arg\max_{a\in A(s)} [r(s,a) + \min_{p\in\mathcal{P}_s^a} p^T V^*], \quad s \in S. \qquad (8)$$

and, with some abuse of notation, the optimal transition vectors (for the environment) are

$$P_s^{*a} \in \arg\min_{p\in\mathcal{P}_s^a} p^T V^*, \quad s \in S, a \in A(s). \qquad (9)$$

Proof: Follows from Part (c) in Theorem 3 and (3). ■

To recap, we showed that T is monotone and a contraction with respect to a weighted max norm. This let us prove in Theorem 3 that T has a unique fixed-point that can be found by an iterative procedure (i.e., *value iteration*). We gave conditions on the optimality of stationary policies and showed how to determine optimal actions for the system and the environment.

B. Uncertainty Set Representations

Refering back to the operator T defined in (3), we see that it is composed of two nested optimization problems—the outer maximization problem for the system and the inner minimization problem for the environment. To be clear, the environment optimization problem for a given state $s \in S$ and control action $a \in A(s)$ refers to $\min_{p\in\mathcal{P}_s^a} p^T V$.

The tractability of the environment optimization problem depends on the structure of the uncertainty set $\mathcal{P}_s^a$. In the remainder of this section, we investiate interval and likelihood uncertainty sets, as these are both statistically meaningful and computationally efficient. Due to lack of space, we do not discuss maximum a priori, entropy, scenario, or ellipsoidal uncertainty models, even though these are included in this framework. The reader should refer to Nilim and El Ghaoui for details [19].

We assume that the uncertainty sets of the MDP factor by state and action for the environmental optimization [19].

Assumption 3. $\mathcal{P}^a$ can be factored as the Cartesian product of its rows, so its rows are uncorrelated. Formally, for every $a \in A$, $\mathcal{P}^a = \mathcal{P}_1^a \times \ldots \times \mathcal{P}_n^a$ where each $\mathcal{P}_i^a$ is a subset of the probability simplex in $\mathbb{R}^n$.

1) Interval Models: A common description of uncertainty for transition matrices corresponding to action $a \in A$ is by intervals $\mathcal{P}^a = \{P^a \mid \underline{P}^a \leq P^a \leq \overline{P}^a,\ P^a\mathbf{1} = 1\}$, where $\underline{P}^a$ and $\overline{P}^a$ are nonnegative matrices $\underline{P}^a \leq \overline{P}^a$. This representation is motivated by statistical estimates of confidence intervals on the individual components of the transition matrix [16]. The environmental optimization problem can be solved in $O(n\log(n))$ time using a bisection method [19].

2) Likelihood Models: The likelihood uncertainty model is motivated by determining the transition probabilities between states through experiments. We denote the experimentally measured transition probability matrix corresponding to action a by F^a and the optimal log-likelihood by $\beta_{\max}$.

Uncertainty in the transition matrix for each action $a \in A$ is described by the *likelihood region* [16]

$$\mathcal{P}^a = \{P^a \in \mathbb{R}^{n\times n} | P^a \geq 0, P^a\mathbf{1} = \mathbf{1}, \sum_{i,j} F^a_{ij}\log P^a_{ij} \geq \beta^a\}, \quad (10)$$

where $\beta^a < \beta^a_{\max}$ and can be estimated for a desired confidence level by using a large sample Gaussian approximation [19]. As described in Assumption 1, we enforce that $F^a_{ij} = 0$ if and only if $P^a_{ij} = 0$ for all $i, j \in S$ and all $a \in A$.

Since the likelihood region in (10) does not satisfy Assumption 3, it must be projected onto each row of the transition matrix. Even with the approximation, likelihood regions are more accurate representations than intervals, which are further approximations of the likelihood region. A bisection algorithm can approximate the environment optimization problem to within an accuracy δ in $O(\log(V_{\max}/\delta))$ time, where $V_{\max}$ is the maximum value of the value function [19].

VI. Computing the optimal control policy

We now find a robust control policy that maximizes the probability of satisfying φ over all transitions in an uncertainty set. We use robust value iteration as described in Section V on the transformed product MDP $\mathcal{M}_{ssp}$ created in Section IV. Finally, we project this control policy to a policy for $\mathcal{M}$.

As we formulated the dynamic programming approach in terms of total expected reward maximization, we define the total expected reward as the reachability probability, which is equivalent to the probability of satisfying the LTL formula. Thus, for all $a \in \hat{A}$, the appropriate rewards are $r(s, a) = 1$ for all $s \in \hat{B}$ and $r(s, a) = 0$ for all $s \in \hat{S}_0$. For the remaining states, $s \in \hat{S}_r$, we initialize the rewards arbitrarily in $[0, 1]$ and compute the optimal worst-case value function using the iteration presented in Theorem 3. The resulting value function V^*_{ssp} gives the satisfaction probability for each state in $\mathcal{M}_{ssp}$.

The value function V^*_p for $\mathcal{M}_p$ is determined from V^*_{ssp}. For $s_p \in S_p$, determine the corresponding state $s_{ssp} \in \hat{S}$ and let $V^*_p(s_p) = V^*_{ssp}(s_{ssp})$. This mapping is surjective, as there is at least one s_p for each s_{ssp}.

Given the optimal worst-case value function V^*_p for the original product MDP $\mathcal{M}_p$, the optimal actions $a^*(s) \in A(s)$ for each $s \in S_r$ can be computed. We do not consider actions for states in $S_0 \cup B$ at this time. However, one cannot simply use the approach for selecting actions given by (8), because not all stationary control policies on $\mathcal{M}_p$ are proper. For states in a maximal end component in S_r, there may be multiple actions that satisfy (8). Arbitrarily selecting actions can lead to situations where the stationary control policy stays in the maximal end component forever and thus never satisfies the specification. We avoid this situation by only selecting an action if it is both optimal (i.e., satisfies (8)) and it has a non-zero probability of transitioning to a state that is not in a maximal end component in S_r. Algorithm 3 selects the action with the highest probability of transitioning to a state not in a maximal end component in S_r.

Algorithm 3 Product MDP Control Policy

```
Require: $V^*_p \in \mathbb{R}^n$, $\mathcal{M}_p = (S, A, P)$, $S_r$, $S_0$, $B$
Ensure: Robust control policy $\mu$
  $visited := S_0 \cup B$;
  $possAct(s) := \{a \in A(s) | (T_a V^*_p)(s) = V^*_p(s)\}$;
  for $s \in S_r$ do
    if $|possAct(s)| = 1$ then
      $\mu(s) := possAct(s)$ and $visited := visited \cup \{s\}$;
    end if
  end for
  while $visited \neq S$ do
    for $s \in S_r \backslash visited$ do
      $maxLeaveProb := 0$;
      $leaveProb := \max_{a \in possAct(s)} \sum_{t \in visited} P^a_{st}$;
      if $leaveProb > maxLeaveProb$ then
        $optAct := a$ and $optState := s$;
      end if
    end for
    $\mu(s) := optAct$ and $visited := visited \cup \{optState\}$;
  end while
```

Theorem 4. *Algorithm 3 returns a robust control policy μ that satisfies $V^{\mu\nu}_p = V^*_p$ for the worst-case environmental policy ν.*

Proof: For each state $s \in S_r$, only actions $a \in A(s)$ that satisfy $(T_a V^*_p)(s) = V^*_p(s)$ need to be considered as all other actions cannot be optimal. We call these *possible actions*. Every state has at least one possible action by construction. A state $s \in visited$ if a possible action has been selected for it that also has a positive probability of leaving S_r. Thus, states in $visited$ are not in an end component in S_r. Initialize $visited = S_0 \cup B$. For every state with only one possible action, select that action and add the state to $visited$. For states with multiple possible actions, only select an action if it has a non-zero probability of reaching $visited$, and thus, leaving S_r. It is always possible to choose an action in this manner from the definition of S_r. Select actions this way until $visited = S$ and return the corresponding policy μ. By construction, μ satisfies $T_{\mu\nu} V^*_p = V^*_p$ and is proper. ∎

The optimal control policy for satisfying the LTL specification φ consists of two parts: a stationary deterministic policy for reaching an accepting maximal end component, and a finite-memory deterministic policy for staying there. The former policy is given by Algorithm 3 and denoted μ_{reach}. The latter policy is a finite-memory policy π_B that selects actions in a round-robin fashion to ensure that the system stays inside the accepting maximal end component forever and satisfies φ [5]. The overall optimal policy is $\pi_p^* = \mu_{reach}$ if $s \notin B$ and $\pi_p^* = \pi_B$ if $s \in B$. We induce an optimal policy π^* on $\mathcal{M}$ from π_p^* as described in Section IV-A.

Complexity

The (worst-case) size of the deterministic Rabin automaton $\mathcal{A}_\varphi$ is doubly-exponential in the length of the LTL formula φ [9]. Experimental work in [13] has shown that deterministic Rabin automaton sizes are often exponential or lower for many common types of LTL formulae. Also, there are fragments of LTL, which include all safety and guarantee properties, that generate a determinstic Rabin automaton whose size is singly-exponential in the length of the formula [2].

The size of the product MDP $\mathcal{M}_p$ is equal to the size of $\mathcal{M}$ times the size of $\mathcal{A}_\varphi$. $\mathcal{M}_p$ has n states and m transitions. Maximal end components can be found in $O(n^2)$ time. Since T is a contraction, an ϵ-suboptimal control policy can be computed in $O(n^2 m \log(1/\epsilon))$ time without uncertainty sets [6] and $O(n^2 m \log(1/\epsilon)^2)$ when using likelihood transition uncertainty sets. Thus, the computational cost for incorporating robustness is $O(\log(1/\epsilon))$ times the non-robust case.

VII. Example

We demonstrate our robust control approach on a discrete-time point-mass robot model. The system model is $x_{k+1} = x_k + (u_x + d_x)\Delta t$, with state $x \in \mathcal{X} \subset \mathbb{R}^2$, control $u \in \mathcal{U} \subset \mathbb{R}^2$, disturbance $d \in \mathcal{D} \subset \mathbb{R}^2$, and time interval $\Delta t = t_{k+1} - t_k$ for $k = 0, 1, \ldots$. The disturbance $d \sim \mathcal{N}(0, \Sigma)$ where $\Sigma = \text{diag}(0.225^2, 0.225^2)$ and d has support on $\mathcal{D}$. The control set $\mathcal{U} = [-0.3, 0.3]^2$ and the disturbance set $\mathcal{D} = [-0.225, 0.225]^2$.

The task for the robot is to sequentially visit three regions of interest while always remaining safe. Once the robot has visited the regions, it should return to the start and remain there. The atomic propositions are $\{\text{home}, \text{unsafe}, R1, R2, R3\}$. The LTL formula for this task is $\varphi = \text{home} \wedge \Diamond\Box\,\text{home} \wedge \Box\neg\text{unsafe} \wedge \Diamond(R1 \wedge \Diamond(R2 \wedge \Diamond R3))$.

We used Monte Carlo simulation (75 samples) to create a finite MDP abstraction $\mathcal{M}$ of our system model, where each state of $\mathcal{M}$ is a square region $[0,1]^2$. The actions at each state include transitioning to one of four neighbors (up,left,down,right) or not moving (which guarantees that the robot remains in its current state). Due to symmetry of the regions and robot, we only calculated transitions for one region. The transition probabilities corresponding to an action of 'up' are (up,left,down,right,self,error) = (0.8133, 0.0267, 0.000, 0.0133, 0.120, 0.0267), with other actions defined similarly. The error state is entered if the robot accidentally transitions across multiple states. We used large sample approximations to estimate β^a in (10) for each uncertainty level [19].

All computations were run on an 2.4 GHz dual-core desktop with 2 GB of RAM. The deterministic Rabin automaton representing φ has 8 states. The product MDP has 437 states and 8671 transitions. It took 1.8 seconds to compute $\mathcal{M}_{ssp}$. It took on average 5.7, 4.7, and 0.47 seconds to generate an

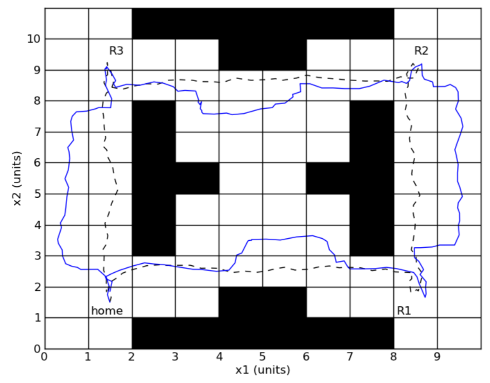

Fig. 1. Sample trajectories of the robot using the nominal (dotted black) and robust likelihood (solid blue) control policies. "unsafe" cells are in black.

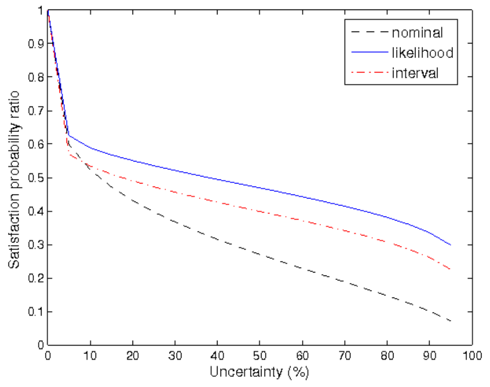

Fig. 2. Ratio of worst-case satisfaction probability of φ for nominal and robust (likelihood and interval) control policies to the nominal control policy with no uncertainty. Larger ratios are better.

ϵ-suboptimal control policy with likelihood, interval, and no (nominal) uncertainty sets. In all cases, $\epsilon = 10^{-6}$.

We calculated the worst-case satisfaction probabilities for the optimal nominal and robust (likelihood and interval) policies by letting the environment pick transition matrices given a fixed control policy. Transitions were assumed to be exact if they did not lead to an "unsafe" cell. The robust likelihood policy outperformed the nominal and interval, as shown in Figure 2. Figure 1 shows sample trajectories of the robot using the robust likelihood and nominal control policies.

VIII. Conclusions and Future Directions

We have presented a method for creating robust control policies for finite MDPs with temporal logic specifications. This robustness is useful when the model is not exact or comes from a finite abstraction of a continuous system. Designing an ϵ-suboptimal robust control policy increases the time complexity only by a factor of $O(\log(1/\epsilon))$ compared to the non-robust policy for statistically relevant uncertainty sets.

In the future, we plan to extend these results to other temporal logics, such as probabilistic computational tree logic (PCTL). We are also looking into weakening the assumptions on the transition matrix uncertainty sets to allow for correlation. Finally, further work needs to be done on abstraction of selected classes of dynamical systems as finite MDPs.

Acknowledgements

The authors would like to thank Scott Livingston for helpful comments. This work was supported by an NSF Graduate Research Fellowship and the Boeing Corporation.

Appendix

Lemma 2: The proof of Lemma 2 closely follows that in [6] (Vol. II, Section 2.4) where the environment policy is fixed. More specifically, the proof in [6] is modified to allow minimization over environment policies. This modification holds due to Assumptions 1 and 2.

First, partition the state space $S = \{1, 2, \ldots, n, t\}$. Let $S_1 = \{t\}$ and for $k = 2, 3, \ldots$, define
$S_k = \{i | i \notin S_1 \cup \cdots \cup S_{k-1}, \min_{a \in A(i)} \max_{j \in S_1 \cup \cdots \cup S_{k-1}} \min_{P \in \mathcal{P}_i^a} P_{ij}^a > 0\}.$

Let m be the largest integer such that S_m is nonempty. By Assumptions 1 and 2, $\cup_{k=1}^m S_k = S$.

Choose a vector $w > 0$ so that T is a contraction with respect to $\| \cdot \|_w$. Take the ith component w_i to be the same for states i in the same set S_k. Choose the components w_i of the vector w by $w_i = y_k$ if $i \in S_k$, where $y_1, \ldots, y_m$ are appropriately chosen scalars satisfying $1 = y_1 < y_2 < \cdots < y_m$.

Let $\epsilon := \min_{k=2,\ldots,m} \min_{\mu \in \Pi_s} \min_{i \in S_k} \min_{P \in \mathcal{P}} \sum_{j \in S_1 \cup \cdots \cup S_{k-1}} P_{ij}^{\mu(i)}$,

where $\mathcal{P}$ is the set of *all* transition matrices that satisfy Assumption 1. Note that $0 < \epsilon \le 1$. We will show that one can choose $y_2, \ldots, y_m$ so that for some $\beta < 1$, $\frac{y_m}{y_k}(1 - \epsilon) + \frac{y_{k-1}}{y_k}\epsilon \le \beta < 1$ for $k = 2, \ldots, m$ and then show that such a choice exists.

For all vectors $u, v \in \mathbb{R}^n$, select μ such that $T_\mu u = Tu$. Then, for all i,

$$\begin{aligned}(Tv)(i) - (Tu)(i) &= (Tv)(i) - (T_\mu u)(i) \\ &\le (T_\mu v)(i) - (T_\mu u)(i) \\ &= \sum_{j=1}^n V_{ij}^{\mu(i)} v(j) - U_{ij}^{\mu(i)} u(j) \\ &\le \sum_{j=1}^n P_{ij}^{\mu(i)} (v(j) - u(j)),\end{aligned}$$

where $U_i^{\mu(i)} = \arg\min_{p \in \mathcal{P}_i^{\mu(i)}} p^T u$, $V_i^{\mu(i)} = \arg\min_{p \in \mathcal{P}_i^{\mu(i)}} p^T v$, and $P_{ij}^{\mu(i)} := \arg\max\{U_{ij}^{\mu(i)}(v(j) - u(j)), V_{ij}^{\mu(i)}(v(j) - u(j))\}$ over $U_{ij}^{\mu(i)}$ and $V_{ij}^{\mu(i)}$ for each j.

Let $k(j)$ be such that state j belongs to the set $S_{k(j)}$. Then, for any constant c, $\| v - u \|_w \le c \implies v(j) - u(j) \le c y_{k(j)}, j = 1, \ldots, n$, and thus for all i,

$$\begin{aligned}\frac{(Tv)(i) - (Tu)(i)}{c y_{k(i)}} &\le \frac{1}{y_{k(i)}} \sum_{j=1}^n P_{ij}^{\mu(i)} y_{k(j)} \\ &\le \frac{y_{k(i)-1}}{y_{k(i)}} \sum_{j \in S_1 \cup \cdots \cup S_{k(i)-1}} P_{ij}^{\mu(i)} \\ &+ \frac{y_m}{y_{k(i)}} \sum_{j \in S_{k(i)} \cup \cdots \cup S_m} P_{ij}^{\mu(i)} \\ &= \left(\frac{y_{k(i)-1}}{y_{k(i)}} - \frac{y_m}{y_{k(i)}} \right) \sum_{j \in S_1 \cup \cdots \cup S_{k(i)-1}} P_{ij}^{\mu(i)} \\ &+ \frac{y_m}{y_{k(i)}} \le \left(\frac{y_{k(i)-1}}{y_{k(i)}} - \frac{y_m}{y_{k(i)}} \right) \epsilon + \frac{y_m}{y_{k(i)}} \le \beta.\end{aligned}$$

Then, $\frac{(Tv)(i) - (Tu)(i)}{w_i} \le c\beta$, $i = 1, \ldots, n$, which taking the max over i gives $\| Tv - Tu \|_w \le c\beta$ for all $u, v \in \mathbb{R}^n$ with $\| u - v \|_w \le c$. Thus, T is a contraction under the $\| \cdot \|_w$ norm.

We now show that scalars $y_1, y_2, \ldots, y_m$ exist such that the above assumptions hold. Let $y_0 = 0, y_1 = 1$, and suppose that $y_1, y_2, \ldots, y_k$ have been chosen. If $\epsilon = 1$, we choose $y_{k+1} = y_k + 1$. If $\epsilon < 1$, we choose y_{k+1} to be $y_{k+1} = \frac{1}{2}(y_k + M_k)$ where $M_k = \min_{1 \le i \le k} \left\{ y_i + \frac{\epsilon}{1-\epsilon}(y_i - y_{i-1}) \right\}$.

Since $M_{k+1} = \min \left\{ M_k, \ y_{k+1} + \frac{\epsilon}{1-\epsilon}(y_{k+1} - y_k) \right\}$, by induction we have that for all k, $y_k < y_{k+1} < M_{k+1}$, and thus one can construct the required sequence. ■

References

[1] A. Abate, A. D'Innocenzo, M. Di Benedetto, and S. Sastry. Markov set-chains as abstractions of stochastic hybrid systems. In *11th International Conference on Hybrid Systems: Computation and Control*, 2008.

[2] R. Alur and S. La Torre. Deterministic generators and games for LTL fragments. *ACM Trans. Comput. Logic*, 5(1):1–25, 2004.

[3] S. Azuma and G. Pappas. Discrete abstraction of stochastic nonlinear systems: A bisimulation function approach. In *American Control Conference*, 2010.

[4] J. A. Bagnell, A. Y. Ng, and J. G. Schneider. Solving uncertain Markov decision processes. Technical report, Carnegie Mellon University, 2001.

[5] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[6] D. P. Bertsekas. *Dynamic Programming and Optimal Control (Vol. I and II)*. Athena Scientific, 2001.

[7] O. Buffet. Reachability analysis for uncertain ssps. In *Proceedings of the IEEE International Conf. on Tools with Artificial Intelligence*, 2005.

[8] K. Chatterjee, K. Sen, and T. Henzinger. Model-checking omega-regular properties of interval Markov chains. In R. M. Amadio, editor, *Foundations of Software Science and Computation Structure (FoSSaCS)*, pages 302–317, March 2008.

[9] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the Association for Computing Machinery*, 42:857–907, 1995.

[10] L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Stanford University, 1997.

[11] X. C. Ding, S. L. Smith, C. Belta, and D. Rus. LTL control in uncertain environments with probabilistic satisfaction guarantees. In *Proceedings of 18th IFAC World Congress*, 2011.

[12] R. Givan, S. Leach, and T. Dean. Bounded-parameter Markov decision processes. *Artificial Intelligence*, 122:71–109, 2000.

[13] J. Klein and C. Baier. Experiments with deterministic omega-automata for formulas of linear temporal logic. *Theoretical Computer Science*, 363:182–195, 2006.

[14] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: verification of probabilistic real-time systems. In *Proceedings of 23rd International Conference on Computer Aided Verification*, 2011.

[15] M. Lahijanian, S. B. Andersson, and C. Belta. Control of Markov decision processes from PCTL specifications. In *Proceedings of the American Control Conference*, 2011.

[16] E. L. Lehmann and J. P. Romano. *Testing Statistical Hypotheses*. Springer, 2005.

[17] R. Majumdar, E. Render, and P. Tabuada. Robust discrete synthesis against unspecified disturbances. In *Proc. Hybrid Systems: Computation and Control*, pages 211–220, 2011.

[18] A. W. Naylor and G. R. Sell. *Linear Operator Theory in Engineering and Science*. Springer-Verlag, 1982.

[19] A. Nilim and L. El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53:780–798, 2005.

[20] J. K. Satia and R. E. L. Jr. Markovian decision processes with uncertain transition probabilities. *Operations Research*, 21(3):pp. 728–740, 1973.

[21] D. Wu and X. Koutsoukos. Reachability analysis of uncertain systems using bounded-parameter Markov decision processes. *Artif. Intell.*, 172:945–954, 2008.