Robotics: Science and Systems 2014
Berkeley, CA, USA, July 12-16, 2014

Probably Approximately Correct MDP Learning and Control With Temporal Logic Constraints

Jie Fu and Ufuk Topcu
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, Pennsylvania 19104
Email: jief, utopcu@seas.upenn.edu

***Abstract*—We consider synthesis of controllers that maximize the probability of satisfying given temporal logic specifications in unknown, stochastic environments. We model the interaction between the system and its environment as a Markov decision process (MDP) with initially unknown transition probabilities. The solution we develop builds on the so-called model-based probably approximately correct Markov decision process (PAC-MDP) method. The algorithm attains an ε-approximately optimal policy with probability $1-\delta$ using samples (i.e. observations), time and space that grow polynomially with the size of the MDP, the size of the automaton expressing the temporal logic specification, $\frac{1}{\varepsilon}$, $\frac{1}{\delta}$ and a finite time horizon. In this approach, the system maintains a model of the initially unknown MDP, and constructs a product MDP based on its *learned model* and the specification automaton that expresses the temporal logic constraints. During execution, the policy is iteratively updated using observation of the transitions taken by the system. The iteration terminates in finitely many execution steps. With high probability, the resulting policy is such that, for any state, the difference between the probability of satisfying the specification under this policy and the optimal one is within a predefined bound.**

I. Introduction

Integrating model-based learning into control allows an agent to complete its assigned mission by exploring its unknown environment, using the gained knowledge to gradually approach an (approximately) optimal policy. In this approach, learning and control complement each other. For the controller to be effective, there is a need for correct and sufficient knowledge of the system. Meanwhile, by exercising a control policy, the agent obtains new percepts, which are then used in learning to improve its model of the system. In this paper, we propose a method that extends model-based probably approximately correct Markov decision process (PAC-MDP) reinforcement learning to temporal logic-constrained control for *unknown*, *stochastic* systems.

A stochastic system can be modeled as an MDP. Control design methods for stochastic systems with temporal logic constraints have been developed [1, 2] and applied successfully to robot motion planning problems [3]. With incomplete knowledge of the system dynamics, a stochastic system can be modeled as an MDP in which the transition probabilities are unknown or only partially known. Take a robotic motion planning problem as an example. Different terrains where the robot operates affect its dynamics in a way that, for the same action of the robot, the probability distributions over the arrived positions differ depending on the level and coarseness of different grounds. The robot dynamics in an unknown terrain can be modeled as an MDP in which the transition probabilities are unknown. Acquiring such knowledge through observations of robot's movement requires possibly infinite number of samples. Yet, in practice, such a requirement may not be affordable or realizable. Alternatively, with finite amount of samples, we may be able to approximate the actual MDP and reason about the optimality and correctness (with respect to the underlying temporal logic specifications) of policies synthesized using this approximation.

We develop an algorithm that computational efficiently updates the controller subject to temporal logic constraints for an unknown MDP. To this end, we extend the PAC-MDP method [4, 5] to maximize the probability of satisfying a given temporal logic specification in an MDP with unknown transition probabilities. In the proposed method, the agent maintains a model of the MDP learned from observations (transitions between different states enabled by actions) and when the learning phase terminates, the learned MDP approximates the true MDP to a specified degree, with a pre-defined high probability. The algorithm balances the ratio of exploration and exploitation: Before the learning phase terminates, either the current policy is approximately optimal, or new information can be invoked by exercising this policy. Finally, at convergence, the policy is ensured to be *approximately optimal*, and the time, space, and sample complexity of achieving this policy is polynomial in the size of the MDP, in the size of the automaton expressing the temporal logic specification and other quantities that measure the accuracy of, and the confidence in, the learned MDP with respect to the true one.

Existing results in temporal logic-constrained verification and control synthesis with unknown systems are mainly in two categories: The first uses statistical model checking and hypothesis testing for Markov chains [6] and MDPs [7]. The second applies inference algorithms to identify the unknown factors and adapt the controller with the inferred model (a probabilistic automaton, or a two-player deterministic game) of the system and its environment [8, 9]. Statistical model checking for MDPs [7] relies on sampling of the trajectories of Markov chains induced from the underlying MDP and policies to verify whether the probability of satisfying a *bounded* linear temporal logic constraint is greater than some quantity for all

admissible policies. It is restricted to *bounded* linear temporal logic properties in order to make the sampling and checking for paths computationally feasible. For linear temporal logic specifications in general, computationally efficient algorithm has not been developed. Reference [10] employs inference algorithms for deterministic probabilistic finite-state automata to identify a subclass of MDPs, namely, deterministic MDPs. Yet, this method requires the data (the state-action sequences in the MDPs) to be independent and identically distributed. Such an assumption cannot hold in the paradigm where learning (exploration) and policy update (exploitation) are carried out in parallel and at run time, simply because that following a particular control policy introduces sampling bias for observations of the system. Reference [8] applies stochastic automata learning combined with probabilistic model checking for stochastic systems. However, it requires an *infinite amount* of experiences for the model to be identified and the policy to be optimal, and may not be affordable in practice.

The extension of the PAC-MDP method to control synthesis with temporal logic constraints shares many attractive features with the original method: First, though it applies to linear temporal logic constraints, it still guarantees efficient convergence to an approximately optimal policy within a *finite time horizon* and the number of policy updates is determined by the size of underlying MDP, independent from the specification. Second, it balances the ratio of exploration (for improving the knowledge of the model) and exploitation (for maximizing the probability of satisfying the specification) phases. Third, it does not require independent and identically distributed samples.

II. Preliminaries

Definition 1 (Labeled MDP)**.** A labeled MDP is a tuple $M = \langle Q, \Sigma, q_0, P, \mathcal{AP}, L\rangle$ where Q and Σ are finite state and action sets. $q_0 \in Q$ is the initial state. The transition probability function $P : Q \times \Sigma \times Q \to [0, 1]$ is defined such that $\sum_{q' \in Q} P(q, \sigma, q') \in \{0, 1\}$ for any state $q \in Q$ and any action $\sigma \in \Sigma$. $\mathcal{AP}$ is a finite set of atomic propositions and $L : Q \to 2^{\mathcal{AP}}$ is a labeling function which assigns to each state $q \in Q$ a set of atomic propositions $L(q) \subseteq \mathcal{AP}$ that are valid at the state q. L can be extended to state sequences in the usual way, i.e., $L(\rho_1\rho_2) = L(\rho_1)L(\rho_2)$ for $\rho_1, \rho_2 \in Q^*$.

The *structure* of the labeled MDP M is the underlying graph $\langle Q, \Sigma, E\rangle$ where $E \subseteq Q \times \Sigma \times Q$ is the set of labeled edges. $(q, \sigma, q') \in E$ if and only if $P(q, \sigma, q') \neq 0$. We say action σ is *enabled* at q if and only if there exists $q' \in Q$, $(q, \sigma, q') \in E$.

A. A specification language

We use linear temporal logic formula (LTL) to specify a set of desired system properties such as safety, liveness, persistence and stability. A formula in LTL is built from a finite set of atomic propositions $\mathcal{AP}$, true, false and the Boolean and temporal connectives $\wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$ and $\Box$ (always), $\mathcal{U}$ (until), $\Diamond$ (eventually), $\bigcirc$ (next). Given an LTL formula φ as the system specification, one can always represent it by a deterministic Rabin automaton (DRA) $\mathcal{A}_\varphi = \langle S, 2^{\mathcal{AP}}, T_s, I_s, \mathsf{Acc}\rangle$ where S is a finite state set, $2^{\mathcal{AP}}$ is the alphabet, $I_s \in S$ is the initial state, and $T_s : S \times 2^{\mathcal{AP}} \to S$ the transition function. The acceptance condition Acc is a set of tuples $\{(J_i, K_i) \in 2^S \times 2^S \mid i = 0, 1, \ldots, m\}$. The run for an infinite word $w = w[0]w[1] \ldots \in (2^{\mathcal{AP}})^\omega$ is the infinite sequence of states $s_0 s_1 \ldots \in S^\omega$ where $s_0 = I_s$ and $s_{i+1} = T_s(s_i, w[i])$. A run $\rho = s_0 s_1 \ldots$ is accepted in $\mathcal{A}_\varphi$ if there exists at least one pair $(J_i, K_i) \in \mathsf{Acc}$ such that $\mathsf{Inf}(\rho) \cap J_i = \emptyset$ and $\mathsf{Inf}(\rho) \cap K_i \neq \emptyset$ where $\mathsf{Inf}(\rho)$ is the set of states that appear infinitely often in ρ.

B. Policy synthesis in a known MDP

Given an MDP and an LTL specification φ, we aim to maximize the probability of satisfying φ from a given state. We now present a standard quantitative synthesis method in a known MDP with LTL specifications, following [11, 12].

Definition 2 (Product MDP)**.** Given an MDP $M = \langle Q, \Sigma, P, q_0, \mathcal{AP}, L\rangle$ and the DRA $\mathcal{A}_\varphi = \langle S, 2^{\mathcal{AP}}, T_s, I_s, \{(J_i, K_i) \mid i = 1, \ldots, m\}\rangle$, the *product MDP* is $\mathcal{M} = M \ltimes \mathcal{A}_\varphi = \langle V, \Sigma, \Delta, v_0, \mathsf{Acc}\rangle$, with components defined as follows: $V = Q \times S$ is the set of states. Σ is the set of actions. The initial state is $v_0 = (q_0, s_0)$ where $s_0 = T_s(I_s, L(q_0))$. $\Delta : V \times \Sigma \times V \to [0, 1]$ is the transition probability function. Given $v = (q, s)$, σ, $v' = (q', s')$ and $s' = T_s(s, L(q'))$, let $\Delta(v, \sigma, v') = P(q, \sigma, q')$. The acceptance condition is $\mathsf{Acc} = \{(\hat{J}_i, \hat{K}_i) \mid \hat{J}_i = Q \times J_i, \hat{K}_i = Q \times K_i, i = 1, \ldots, m\}$.

A *memoryless, deterministic policy* for a product MDP $\mathcal{M} = \langle V, \Sigma, \Delta, v_0, \mathsf{Acc}\rangle$ is a function $f : V \to \Sigma$. A memoryless policy f in $\mathcal{M}$ is in fact a finite-memory policy f' in the underlying MDP M. Given a state $(q, s) \in V$, we can consider s to be a memory state, and define $f'(\rho) = f((q, s))$ where the run $\rho = q_0 q_1 \ldots q_n$ satisfies $q_n = q$ and $T_s(I_s, L(\rho)) = s$. For the type of MDPs in Def. 1 and LTL specifications, memoryless, deterministic policies in the product MDP are sufficient [13] for maximizing the probability of satisfying the specifications.

Definition 3 (Markov chain induced by a policy)**.** Given an MDP $\mathcal{M} = \langle V, \Sigma, \Delta, v_0, \mathsf{Acc}\rangle$ and a policy $f : V \to \Sigma$, the *Markov chain induced by policy* f is a tuple $\mathcal{M}^f = \langle V, \Sigma, \Delta^f, v_0, \mathsf{Acc}\rangle$ where $\Delta^f(v, v') = \Delta(v, f(v), v')$.

A *path* in a Markov chain is a (finite or infinite) sequence of states $x \in V^*$ (or V^ω). Given a Markov chain $\mathcal{M}^f$, starting from the initial state v_0, the state visited at the step t is a random variable X_t. The probability of reaching state v' from state v in one step, denoted $\Pr(X_{t+1} = v' \mid X_t = v)$, equals $\Delta^f(v, v')$. This is extended to a unique measure $\Pr$ over a set of (infinite) paths of $\mathcal{M}^f$, $\Pr(v_0 v_1 \ldots v_n) = \Pr(X_n = v_n \mid X_{n-1} = v_{n-1}) \cdot \Pr(v_0 v_1 \ldots v_{n-1})$.

The following notations are used in the rest of the paper: For a Markov chain $\mathcal{M}^f$, induced by policy f from MDP $\mathcal{M}$, let $h^{\leq i}(v, X)$ (resp. $h^i(v, X)$) be the probability of a path starts from state v and hits the set X for the first time within i steps (resp. at the exact i-th step). By definition, $h^{\leq i}(v, X) =$

$\sum_{k=0}^{i} h^i(q, X)$. In addition, let $h(v, X) = \sum_{k=0}^{\infty} h^k(v, X)$, which is the probability of a path that starts from state v and enters the set X eventually. When multiple Markov chains are involved, we write $h_{\mathcal{M}^f}$ and $\Pr_{\mathcal{M}^f}$ to distinguish the hitting probability h and the probability measure $\Pr$ in $\mathcal{M}^f$.

Definition 4 (Accepting end components)**.** The *end component* for the product MDP $\mathcal{M}$ denotes a pair (W, f) where $W \subseteq V$ is non-empty and $f : W \to \Sigma$ is defined such that for any $v \in W$, $\sum_{v' \in W} \Delta(v, f(v), v') = 1$; and the induced directed graph $(W, \to_f)$ is strongly connected. Here, $v \to_f v'$ is an edge in the directed graph if $\Delta(v, f(v), v') > 0$. An accepting end component (AEC) is an end component such that $W \cap \hat{J}_i = \emptyset$ and $W \cap \hat{K}_i \neq \emptyset$ for some $i \in \{1, \ldots, m\}$.

Let the set of AECs in $\mathcal{M}$ be $\mathsf{AEC}(\mathcal{M})$ and the set of *accepting end states* be $\mathcal{C} = \{v \mid \exists (W, f) \in \mathsf{AEC}(\mathcal{M}), v \in W\}$. Due to the property of AECs, once we enter some state $v \in \mathcal{C}$, we can find an AEC (W, f) such that $v \in W$, and initiate the policy f such that for some $i \in \{1, \ldots, m\}$, all states in $\hat{J}_i$ will be visited only finite number of times and some state in $\hat{K}_i$ will be visited infinitely often. Given the structure of $\mathcal{M}$, the set $\mathsf{AEC}(\mathcal{M})$ can be computed by algorithms in [14, 15]. Therefore, given the system MDP M and its specification automaton $\mathcal{A}_\varphi$, to maximize the probability of satisfying the specification, we want to synthesize a policy f that maximizes the probability of hitting the set of accepting end states $\mathcal{C}$, and after hitting the set, a policy in the accepting end component will be followed. The set of accepting end components can be computed in time polynomial in the size of the product MDP. For a reachability specification φ, for any state, a policy that maximizes the probability of satisfying φ indeed maximizes the probability of hitting a set of final states in the product MDP from that state (see [11]), our result is still applicable and there is no need to compute the end components.

C. Problem statement

The synthesis method in Section II produces the optimal policy only if the MDP model is known. However, in practice, such knowledge may not be available. For an MDP with unknown transition probabilities, model-based reinforcement learning approach suggests the system learns a model of the true MDP at run time, and uses the knowledge to iteratively update its policy. Moreover, the learning and policy update shall be efficient and eventually the policy converges to one which meets a certain criterion of success. Tasked with maximizing the probability of satisfying the specification, we define, for a given policy, the state value in the product MDP is the probability satisfying the specification from that state onwards and the optimal policy is the one that maximizes the state value for each state in the product MDP. The probability of satisfying an LTL specification is indeed the probability of entering the set of accepting end states in the product MDP (see Section II). We introduce the following definition.

Definition 5 (T-step state values and state values)**.** Let $\mathcal{M}$ be the product MDP, $\mathsf{AEC}(\mathcal{M})$ be the set of accepting end components, and f be a policy in $\mathcal{M}$. For each state $v \in V$, given a finite horizon $T \in \mathbb{N}$, the T-step state value is $U^f_{\mathcal{M}}(v, T) = h^{\leq T}_{\mathcal{M}^f}(v, \mathcal{C})$, where $\mathcal{C}$ is the set of accepting end states obtained from $\mathsf{AEC}(\mathcal{M})$. The *optimal T-step state value* is $U^*_{\mathcal{M}}(v, T) = \max_f \{U^f_{\mathcal{M}}(v, T)\}$, and the *optimal T-step policy* is $f^*_T = \arg\max_f \{U^f_{\mathcal{M}}(v, T)\}$. Similarly, We define the state value $U^f_{\mathcal{M}}(v) = h_{\mathcal{M}^f}(v, \mathcal{C})$. The *optimal state value* is $U^*_{\mathcal{M}}(v) = \max_f \{U^f_{\mathcal{M}}(v)\}$ and the *optimal policy* is $f^* = \arg\max_f \{U^f_{\mathcal{M}}(v)\}$.

We can now state the main problem of the paper.

Problem 1. Given an MDP $M = \langle Q, \Sigma, q_0, P, \mathcal{AP}, L\rangle$ with unknown transition probability function P, and an LTL specification automaton $\mathcal{A}_\varphi = \langle S, 2^{\mathcal{AP}}, T_s, I_s, \mathsf{Acc}\rangle$, two parameters ε and δ, design an algorithm which, with probability at least $1 - \delta$, outputs a policy $f : Q \times S \to \Sigma$ such that for any state (q, s), the T-step state value of policy f is ε-close to the optimal state value in $\mathcal{M}$, and the sample, space and time complexity required for this algorithm is less than some polynomial in the relevant quantities $(|Q|, |S|, |\Sigma|, \frac{1}{\varepsilon}, T, \frac{1}{\delta})$.

III. Main result

A. Overview

First we provide an overview of our solution to Problem 1. Assume that the system has full observations over the state and action spaces, in the underlying MDP M, the set of states are partitioned into *known* and *unknown* states (see Definition 8). Informally, a state becomes known if it has been visited sufficiently many times, which is determined by some confidence level $1 - \delta$ and a parameter ϵ, the number of states and the number of actions in M, and a finite-time horizon T.

Since the true MDP is unknown, we maintain and update a learned MDP $\overline{M}$, as well as the product MDP $\overline{\mathcal{M}}$. Based on the partition of known and unknown states, and the estimates of transition probabilities for the set of known states $H \subseteq Q$, we consider the set of states $\hat{H} = H \times S$ in $\overline{\mathcal{M}}$ to be known and construct a sub-MDP $\overline{\mathcal{M}}_{\hat{H}}$ of $\overline{\mathcal{M}}$ that only includes the set $\hat{H}$ of known states, with an additional sink state that groups together the set of unknown states $V \setminus \hat{H}$. A policy is computed in order to maximize the probability of hitting some target set in $\overline{\mathcal{M}}_{\hat{H}}$ within a *finite*-time horizon T. We show that by following this policy, in T steps, either there is a high probability of hitting a state in the accepting end states of $\mathcal{M}$, or some unknown state will be explored. Then, we show that at some finite point all states will become known and the learning phase terminates.

Once all states become known, the structure of M must have been identified and the set of accepting end components in the learned product MDP $\overline{\mathcal{M}}$ is exactly these in the true product MDP $\mathcal{M}$. As a result, with probability at least $1 - \delta$, the policy obtained in $\overline{\mathcal{M}}$ is *near optimal*. Informally, a policy f is near optimal, if, from any initial state, the probability of satisfying the specification with f in T steps is no less than the probability of eventually satisfying the specification with the optimal policy, minus a small quantity.

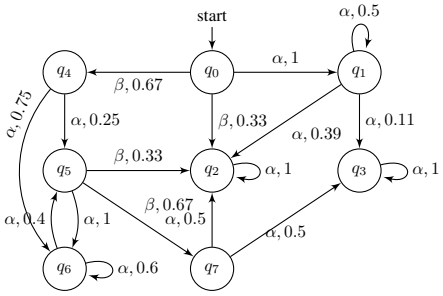


Fig. 1. Example of an MDP with states $Q = \{q_i, i = 0, \ldots, 7\}$, actions $\Sigma = \{\alpha, \beta\}$, and transition probability function P as indicated.

Example: Consider the MDP taken from [11, page 855], as a running example. The objective is to always eventually visiting the state q_3, i.e., $\varphi = \square\lozenge q_3$. In [11], the MDP is known and the algorithm for computing the optimal policy is given. As the MDP has already encoded the information of the specification, the atomic propositions are omitted and we can use the MDP M as the product MDP $\mathcal{M}$ with acceptance condition $\{(\emptyset, \{q_3\})\}$ and the accepting end component is $(\{q_3\}, f(q_3) = \alpha)$. For this known MDP, with respect to φ, the optimal policy f^* and the probability of satisfying the specification under f^* is obtained in Table I.

TABLE I
THE OPTIMAL POLICY AND STATE VALUES IN THE MDP OF FIG. 1.

	q_0	q_1	q_2	q_3	q_4	q_5	q_6	q_7
$f^*(\cdot)$	β	α	α	α	α	β	α	α
$U^*_{\mathcal{M}}(\cdot)$	0.22445	0.22	0	1	0.335	0.335	0.335	0.5

B. Maximum likelihood estimation of transition probabilities

For the MDP M, we assume that for each state-action pair, the probability distribution $\mathrm{Dist}(q, a) : Q \to [0, 1]$, defined by $\mathrm{Dist}(q, a)(q') = P(q, a, q')$, is an independent Dirichlet distribution (as in [16, 17, 18]). For each transition $(q, a, q') \in Q \times \Sigma \times Q$ and each discrete time t for $t \geq 0$, we associate an integer $\theta^t(q, a, q')$ with (q, a, q') to represent the number of observations of such a transition. Let $\theta^t(q, a) = \sum_{q' \in Q} \theta^t(q, a, q')$. At time t, with $\theta^t(q, a, q')$ large enough, the *maximum likelihood estimator* [19] of the transition probability $P(q, \sigma, q')$ is a random variable of normal distribution with mean and variance, respectively, $\overline{P}(q, \sigma, q') = \frac{\theta^t(q,a,q')}{\theta^t(q,a)}$, and $\mathrm{Var} = \frac{\theta^t(q,\sigma,q')(\theta^t(q,a)-\theta^t(q,\sigma,q'))}{\theta^t(q,a)^2(\theta^t(q,a)+1)}$.

C. Approximating the underlying MDP

We extend the definition of α-approximation in MDPs [4], to labeled MDPs.

Definition 6 (α-approximation in labeled MDPs)**.** Let M and $\overline{M}$ be two labeled MDPs over the *same state and action spaces* and let $0 < \alpha < 1$. $\overline{M}$ is an α-approximation of M if $\overline{M}$ and M share the same labeling function and the same structure, and for any state q_1 and q_2, and any action $a \in \Sigma$, it holds that $\left|P(q_1, a, q_2) - \overline{P}(q_1, a, q_2)\right| \leq \alpha$.

By construction of the product MDP, it is easy to prove that if $\overline{M}$ α-approximates M, then $\overline{\mathcal{M}} = \overline{M} \ltimes \mathcal{A}_\varphi$ is an α-approximation of $\mathcal{M} = M \ltimes \mathcal{A}_\varphi$. In the following, we denote the true MDP (and its product MDP) by M (and $\mathcal{M}$), the learned MDP (and the learned product MDP) by $\overline{M}$ (and $\overline{\mathcal{M}}$).

Since the true MDP is unknown in Problem 1, at each time instance, we can only compute a policy f using our hypothesis for the true model. Thus, we need a method for evaluating the performance of the synthesized policy. For this purpose, based on the simulation lemma in [5, 4], the following lemma is derived. It provides a way of estimating the T-step state values under the synthesized policy in the unknown MDP $\mathcal{M}$, using the MDP learned from observations and the approximation error between the true MDP and our hypothesis.

Lemma 1. *Given two MDPs $M = \langle Q, \Sigma, P, \mathcal{AP}, L\rangle$ and $\overline{M} = \langle Q, \Sigma, \overline{P}, \mathcal{AP}, L\rangle$, consider $\overline{M}$ is an $\frac{\epsilon}{NT}$ -approximation of M where N is the number of states in M (and $\overline{M}$), T is a finite time horizon, and $0 < \epsilon < 1$. Then for any specification automaton $\mathcal{A}_\varphi = \langle S, 2^{\mathcal{AP}}, T_s, I_s, \mathsf{Acc}\rangle$, for any state v in the product MDP $\mathcal{M} = M \ltimes \mathcal{A}_\varphi = \langle V, \Sigma, \Delta, v_0, \mathsf{Acc}\rangle$, for any policy $f : V \to \Sigma$, we have that $\left|U^f_{\mathcal{M}}(v, T) - U^f_{\overline{\mathcal{M}}}(v, T)\right| \leq \epsilon$.*

The proof of Lemma is given in Appendix. It is worth mentioning that though the confidence level $1 - \delta$ is achieved for the estimation of each transition probability, the confidence level on the bound between $U^f_{\mathcal{M}}(v, T)$ and $U^f_{\overline{\mathcal{M}}}(v, T)$ for T steps is not $(1 - \delta)^T$. See the proof for more details.

Lemma 1 is important in two aspects. First, for any policy, it allows us to estimate the ranges of T-step state values in the true MDP using its approximation. We will show in Section III-D that the learned MDP approximates the true MDP for some $0 < \alpha < 1$. Second, it shows that for a given finite time horizon T, the size of the specification automaton will not affect the accuracy requirement on the learned MDP for achieving an ϵ-close T-step state value for any policy and any initial state. Thus, even if the size of the specification automaton is exponential in the size of the temporal logic specification, this exponential blow-up will not lead to any exponential increase of the required number of samples for achieving a desired approximation through learning. Yet, the specification influences the choice of T potentially. In the following we will discuss how to choose such a finite time horizon T and the potential influence.

Lemma 2. *Let $\overline{M}$ be an $\frac{\epsilon}{NT}$-approximation of M. For any specification automaton $\mathcal{A}_\varphi$, suppose $f : V \to \Sigma$ and $g : V \to \Sigma$ be the T-step optimal policy in $\overline{\mathcal{M}} = \overline{M} \ltimes \mathcal{A}_\varphi$ and $\mathcal{M} = M \ltimes \mathcal{A}_\varphi$ respectively. For any state $v \in V$, it holds that $\left|U^f_{\mathcal{M}}(v, T) - U^g_{\mathcal{M}}(v, T)\right| \leq 2\epsilon$.*

Proof: The result directly follows from $U^g_{\overline{\mathcal{M}}}(v, T) \leq U^f_{\overline{\mathcal{M}}}(v, T)$, $\left|U^f_{\mathcal{M}}(v, T) - U^f_{\overline{\mathcal{M}}}(v, T)\right| \leq \epsilon$ and $\left|U^g_{\overline{\mathcal{M}}}(v, T) - U^g_{\mathcal{M}}(v, T)\right| \leq \epsilon$, which can be derived from Lemma 1. ∎

The finite time horizon T is chosen in a way that for the optimal policy f, the state-value $U^f_{\mathcal{M}}(v, T)$ has to be sufficiently close to the probability of satisfying the specification eventually (an infinite horizon), that is, $U^f_{\mathcal{M}}(v)$.

Definition 7 (ϵ-state value mixing time)**.** Given the product MDP $\mathcal{M}$ and a policy f, let $d^f(t) = \max_{v\in V}\left|U^f_{\mathcal{M}}(v,t) - U^f_{\mathcal{M}}(v)\right|$, and the *$\epsilon$-state value mixing time* is defined by $t^f_{\text{mix}}(\epsilon) := \min\{t :\ d^f(t) \le \epsilon\}$.

Given some $0 < \epsilon < 1$, we can use an (estimated) upper bound of the ϵ-state value mixing time $t^f_{\text{mix}}(\epsilon)$ for the optimal policy f as the finit time horizon T. The estimated upper bound may vary for different temporal logic specifications.

D. Exploration and exploitation

In this section, we use an exploration-exploitation strategy similar to that of the R-max [5]. The basic idea is that the system always exercises a T-step optimal policy in some MDP constructed from its current knowledge (exploitation). Here T is chosen to be the ϵ-state value mixing time of the optimal policy. It is guaranteed that if there exists a state for which the system does not know enough due to insufficient observations, the probability of hitting this unknown state is non-zero within T steps, which encourages the agent to explore the unknown state. Once all states are known, based on Lemma 1 and 2, the T-step optimal policy synthesized with our hypothesis performs nearly as optimal as the true optimal policy.

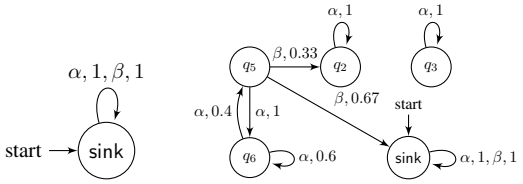
We now formally introduce the notions of known states and known MDP following [4].

Definition 8 (Known states)**.** Let M be an MDP and $\mathcal{A}_\varphi$ be a specification automaton. Let q be a state of M and $\sigma \in \Sigma$ be an action enabled from q. Let T be the ϵ-state-value mixing time of the optimal policy in $\mathcal{M} = M \ltimes \mathcal{A}_\varphi$. A probabilistic transition (q,σ,q') is *known* if with probability at least $1-\delta$, we have for any $q' \in Q$, $\text{Var}\cdot k \le \frac{\epsilon}{NT}$, where k is the critical value for the $1-\delta$ confidence interval [20], Var is the variance of the maximum likelihood estimator for the transition probability $P(q,\sigma,q')$, N is the number of states in M. A state q is *known* if and only if for any action σ enabled from q, and for any state q' that can be reached by action σ, the probabilistic transition (q,σ,q') is known.

Definition 9 (Known product MDP)**.** Given $H \subseteq Q$ the set of known states in an MDP M, let $\hat{H} \times S \subseteq V$ be the set of known states in the product MDP $\mathcal{M}$. The *known* product MDP is $\mathcal{M}_{\hat{H}} = \langle \hat{H} \cup \{\mathsf{sink}\}, \Sigma, \Delta_{\hat{H}}, v_0, \mathsf{Acc}_{\hat{H}}\rangle$ where $\hat{H} \cup \{\mathsf{sink}\}$ is the set of states and sink is the sink state. $\Delta_{\hat{H}}$ is the transition probability function and is defined as follows: If both $v, v' \in \hat{H}$, $\Delta_{\hat{H}}(v,\sigma,v') = \Delta(v,\sigma,v')$. Else if $v \in \hat{H}$ and there exists $\sigma \in \Sigma$ such that $\Delta(v,\sigma,v') > 0$ for some $v' \notin \hat{H}$, then let $\Delta_{\hat{H}}(v,\sigma,\mathsf{sink}) = \sum_{v'\notin\hat{H}} \Delta(v,\sigma,v')$. For any $\sigma \in \Sigma$, $\Delta_{\hat{H}}(\mathsf{sink},\sigma,\mathsf{sink}) = 1$. The acceptance condition $\mathsf{Acc}_{\hat{H}}$ in $\mathcal{M}_{\hat{H}}$ is a set of pairs $(\{(\hat{J}_i \cap \hat{H}, \hat{K}_i \cap \hat{H}) \mid i = 0,\dots,m\} \cup \{(\emptyset,\{\mathsf{sink}\})\}) \setminus \{(\emptyset,\emptyset)\}$.

Intuitively, by including $(\emptyset,\{\mathsf{sink}\})$ in $\mathsf{Acc}_{\hat{H}}$, we encourage the exploration of unknown states aggregated in sink.

Example (cont.): Initially, all states in the product MDP (Fig. 1) are unknown, and thus the known product MDP has only state sink, see Fig. 2(a). Figure 2(b) shows the known product MDP M_H where $H = \{q_2, q_3, q_5, q_6\}$.



(a) $M_\emptyset$ (b) M_H, $H = \{q_2, q_3, q_5, q_6\}$

Fig. 2. Two known product MDPs constructed from the example MDP with the sets of known states $\emptyset$ and $\{q_2, q_3, q_5, q_6\}$ respectively.

The following lemma shows that the optimal T-step policy in $\mathcal{M}_{\hat{H}}$ either will be near optimal in the product MDP $\mathcal{M}$, or will allow a rapid exploration of an unknown state in M.

Lemma 3. *Given a product MDP $\mathcal{M}$ and a set of known states $\hat{H} \subset V$, for any $v \in \hat{H}$, for $0 < \alpha < 1$, let f be the optimal T-step policy in $\mathcal{M}_{\hat{H}}$. Then, one of the following two statements holds: 1) $U^f_{\mathcal{M}}(v,T) \ge U^*_{\mathcal{M}}(v,T) - \alpha$. 2) An unknown state which is not in the accepting end state set $\mathcal{C}$ will be visited in the course of running f for T steps with a probability at least α.*

Proof: Suppose that $U^f_{\mathcal{M}}(v,T) < U^*_{\mathcal{M}}(v,T) - \alpha$ (otherwise, f witnesses the claim). First, we show, for any policy $g : V \to \Sigma$ and any $v \in \hat{H}$, it holds that

$$U^g_{\mathcal{M}_{\hat{H}}}(v,T) \ge U^g_{\mathcal{M}}(v,T). \tag{1}$$

For notation simplicity, let $\Pr = \Pr_{\mathcal{M}^g}$ be the probability measure over the paths in $\mathcal{M}^g$ and $\Pr' = \Pr_{\mathcal{M}^g_{\hat{H}}}$ be the probability measure over the paths in $\mathcal{M}^g_{\hat{H}}$.

Let $X \subseteq V^*$ be a set of paths in $\mathcal{M}^g$ such that each $x \in X$, with $|x| \le T$, starts in v, ends in $\mathcal{C}$ and has every state in $\hat{H}$; $Y \subseteq V^*$ be the set of paths in $\mathcal{M}^g$ such that each $y \in Y$, with $|y| \le T$, starts in v, ends in $\mathcal{C}$ and has at least one state not in $\hat{H}$; and Y' be the set of paths y in $\mathcal{M}^g_{\hat{H}}$ which starts in v, ends with sink and has length $|y| \le T$. We can write

$$U^g_{\mathcal{M}}(v,T) = \sum_{x\in X}\Pr(x) + \sum_{y\in Y}\Pr(y), \text{ and}$$
$$U^g_{\mathcal{M}_{\hat{H}}}(v,T) = \sum_{x\in X}\Pr'(x) + \sum_{y\in Y'}\Pr'(y).$$

Since the transition probabilities in $\mathcal{M}$ and $\mathcal{M}_{\hat{H}}$ are same for the set of known states, and X is the set of paths which only visit known states, we infer that $\sum_{x\in X}\Pr(x) = \sum_{x\in X}\Pr'(x)$. Moreover, since $y \in Y$ contains an unknown state, it leads to sink in $\mathcal{M}_{\hat{H}}$, and thus is in Y'. We infer that $Y \subseteq Y'$, $\sum_{y\in Y}\Pr(y) \le \sum_{y\in Y'}\Pr'(y)$ and thus $U^g_{\mathcal{M}_{\hat{H}}}(v,T) \ge U^g_{\mathcal{M}}(v,T)$.

Next, let f be the optimal T-step policy in $\mathcal{M}_{\hat{H}}$ and ℓ be the optimal T-step policy in $\mathcal{M}$. From Eq. (1), we obtain an inequality: $U^\ell_{\mathcal{M}_{\hat{H}}}(v,T) \ge U^\ell_{\mathcal{M}}(v,T)$.

By the T-step optimality of f in $\mathcal{M}_{\hat{H}}$ and ℓ in $\mathcal{M}$, it also holds that $U^f_{\mathcal{M}_{\hat{H}}}(v,T) \ge U^\ell_{\mathcal{M}_{\hat{H}}}(v,T)$ and $U^*_{\mathcal{M}}(v,T) =$

$U^{\ell}_{\mathcal{M}}(v,T) \geq U^{f}_{\mathcal{M}}(v,T)$. Hence,

$$U^{f}_{\mathcal{M}_{\hat{H}}}(v,T) \geq U^{\ell}_{\mathcal{M}_{\hat{H}}}(v,T) \geq U^{*}_{\mathcal{M}}(v,T) \geq U^{f}_{\mathcal{M}}(v,T)$$
$$\implies U^{f}_{\mathcal{M}_{\hat{H}}}(v,T) - U^{f}_{\mathcal{M}}(v,T) \geq U^{*}_{\mathcal{M}}(v,T) - U^{f}_{\mathcal{M}}(v,T).$$

Given the fact that $U^{*}_{\mathcal{M}}(v,T) - U^{f}_{\mathcal{M}}(v,T) > \alpha$, we infer that $U^{f}_{\mathcal{M}_{\hat{H}}}(v,T) - U^{f}_{\mathcal{M}}(v,T) = \sum_{z \in Z} \Pr(z) > \alpha$, where Z is the set of paths such that each $z \in Z$ starts from v, and ends in some *unknown* state which is *not* an accepting end state in $\mathcal{M}$. Therefore, we reach at the conclusion that if $U^{f}_{\mathcal{M}}(v,T) < U^{*}_{\mathcal{M}}(v,T) - \alpha$, then the probability of visiting an unknown state which is not in $\mathcal{C}$ must be at least α. ∎

Note that, for any unknown state which is in $\mathcal{C}$, one can apply the policy in its corresponding accepting end component to visit such a state infinitely often, and after a sufficient number of visits, it will become known. Also, though we use the product MDP $\mathcal{M}$, Lemma 3 can also be applied to the learned product MDP $\overline{\mathcal{M}}$.

IV. PAC-MDP ALGORITHM IN CONTROL WITH TEMPORAL LOGIC CONSTRAINTS.

Theorem 1. *Let $M = \langle Q, \Sigma, P, \mathcal{AP}, L\rangle$ be an MDP with P unknown, and φ be an LTL formula. Let $0 < \delta < 1$, and $\epsilon > 0$ be input parameters. Let $\mathcal{M} = M \ltimes \mathcal{A}_{\varphi}$ be the product MDP and T be the ϵ-state value mixing time of the optimal policy in $\mathcal{M}$. Let $F_{\mathcal{M}}(\epsilon, T)$ be the set of policies in $\mathcal{M}$ whose ϵ-state value mixing time is T. With probability no less than $1-\delta$, Algorithm 1 will return a policy $f \in F_{\mathcal{M}}(\epsilon, T)$ such that $\left|U^{f}_{\mathcal{M}}(v,T) - U^{*}_{\mathcal{M}}(v)\right| \leq 3\epsilon$ within a number of steps polynomial in $|Q|$, $|\Sigma|$, $|S|$, T, $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$.*

Proof: Firstly, applying the Chernoff bound [20], the upper bound on the number of visits to a state for it to be known is polynomial in $|\Sigma|, T, \frac{1}{\epsilon}$ and $\frac{1}{\delta}$. Before all states are known, the current policy f exercised by the system is T-step optimal in $\overline{\mathcal{M}}_{\hat{H}}$ induced from the set of known states $\hat{H}$. Then, by Lemma 3, either for each state, policy f attains a state value α-close to the optimal T-step state value in $\overline{\mathcal{M}}$, or an unknown state will be visited with probability at least α. However, because $\overline{\mathcal{M}}_{\hat{H}}$ is $\frac{\epsilon}{NT}$-approximation of $\mathcal{M}_{\hat{H}}$, Lemma 1 and Lemma 2 guarantee that policy f either attains a state value $(2\epsilon+\alpha)$-close to the optimal T-step state value in $\mathcal{M}$ for any state, or explores efficiently. If it is always not the first case, then after a finite number of steps, which is polynomial in $|Q|, |\Sigma|, T, \frac{1}{\epsilon}, \frac{1}{\delta}$, all states will be known, and the learned MDP $\overline{M}$ (resp. $\overline{\mathcal{M}}$) $\frac{\epsilon}{NT}$-approximates the true MDP M (resp. $\mathcal{M}$). Since T is the ϵ-state value mixing time of the optimal policy in $\mathcal{M}$, the T-step optimal policy $g: V \to \Sigma$ in $\mathcal{M}$ satisfies $|U^{g}_{\mathcal{M}}(v,T) - U^{*}_{\mathcal{M}}(v)| \leq \epsilon$. From Lemma 2, it holds that $\left|U^{f}_{\mathcal{M}}(v,T) - U^{g}_{\mathcal{M}}(v,T)\right| \leq 2\epsilon$ and thus we infer that $\left|U^{f}_{\mathcal{M}}(v,T) - U^{*}_{\mathcal{M}}(v)\right| \leq 3\epsilon$. ∎

Note that, the sample complexity of the algorithm is polynomial in $|Q|, |\Sigma|, T, \frac{1}{\epsilon}$, and $\frac{1}{\delta}$, and is independent from the size of $\mathcal{A}_{\varphi}$. However, in the value iteration step, the space and time complexity of policy synthesis is polynomial in $|Q|, |S|, |\Sigma|, T, \frac{1}{\epsilon}, \frac{1}{\delta}$.

In problem 1, we aim to obtain a policy f which is ε-optimal in $\mathcal{M}$. This can be achieved by setting $\epsilon = \frac{\varepsilon}{3}$ (see Theorem 1).

In Algorithm 1, the policy is updated at most $|Q|$ times as there is no need to update once all states become known. Given the fact that for LTL specifications, the time complexity of synthesis in a known MDP is $\mathcal{O}(n \cdot m)$ where n is the number of states and m is the number of transitions in the product MDP, Algorithm 1 is a provably efficient algorithm for learning and policy update. Moreover, the input T can be eliminated by letting $T = 1$ and iteratively increase T by 1 (see [4, 5] for more details on the elimination technique).

Algorithm 1: `LearnAndSynthesize`

Input: The state and action sets Q ,Σ, the set of atomic propositions $\mathcal{AP}$ and the labeling function $L: Q \to 2^{\mathcal{AP}}$, the specification DRA $\mathcal{A}_{\varphi}$, parameters ϵ and δ, the (upper bound of) ϵ-state mixing time T for the optimal policy in $M \ltimes \mathcal{A}_{\varphi}$.

Output: A policy $f: Q \times S \to \Sigma$.

begin

- $H := \emptyset$, $q := q_0$, $s := T_s(I_s, L(q))$,
- recompute=True;
- $\overline{M} = \langle Q, \Sigma, \overline{P}, \mathcal{AP}, L\rangle$, `/*` $\overline{P}(q,a,q') = 0$ for any $(q,a,q') \in Q \times \Sigma \times Q$. `*/`
- **while** *True* **do**
  - **if** *recompute=True* **then**
    - $\hat{H} = H \times S$, $\overline{\mathcal{M}} = \overline{M} \ltimes \mathcal{A}_{\varphi}$,
    - $\overline{\mathcal{M}}_{\hat{H}} :=$ `KnownMDP`$(\overline{\mathcal{M}}, \hat{H})$, `/* Obtain the known product MDP in Def. 9. */`
    - $f :=$ `ValueIteration`$(\overline{\mathcal{M}}_{\hat{H}}, T)$, `/* Standard value iteration to compute` T`-step optimal policy. */`
  - $q', a =$ `Exploit`$((q,s), f)$, `/* Apply policy` f `to state` (q,s). `*/`
  - $H_p = H$,
  - $\overline{M}, H :=$ `Update`$(\overline{M}, H, q, a, q', \epsilon, \delta, |Q|, T)$ `/* Update the estimates for transition probabilities. */`
  - **if** $H_p \neq H$ **then** recompute=True;
  - **else** recompute=False;
  - **if** $\overline{P}(q',a,q') = 1$ *or* $(q,s) \in \overline{\mathcal{C}}$; `/*` $\overline{\mathcal{C}}$ `is the accepting end states in` $\overline{M}$. `*/` **then** With probability $0 \leq p \leq 1$, restart with a random state $(q,s) \in Q \times S$;
  - **else** $q := q'$, $s := T_s(s, L(q))$;
  - `/* With prob` p`, random sample` $Q \times S$ `to explore all state space. */`
  - **if** $H = Q$ **then**
    - **return** $f :=$ `ValueIteration`$(\overline{\mathcal{M}}, T)$.

During learning, it is possible that for state $q \in Q$ and for action $a \in \Sigma$, we estimate that $\overline{P}(q,a,q) = 1$. Then either in the true MDP, $P(q,a,q) = 1$, or, $P(q,a,q) < 1$ yet we have not observed a transition (q,a,q') for any $q' \neq q$. In this case, with some probability p, we restart with a random initial state of MDP. With probability $1-p$, we keep exploring state q. The probability p is a tuning parameter in Algorithm 1.

V. Examples

We apply Algorithm 1 to the running example MDP (Fig. 1) and a robotic motion planning problem in an unknown terrain. The implementations are in Python on a desktop with Intel(R) Core(TM) processor and 16 GB of memory.

A. The running example

We consider different assignments for ε and 95% of confidence level, i.e., $\delta = 0.05$, and $T = 15$ as the (estimated upper bound of) $\frac{\varepsilon}{3}$-state value mixing time of the optimal policy, for all assignments of ε. A *step* means that the system takes an action and arrives at a new state.

For $\varepsilon = 0.01$, after 274968 steps and 35.12 seconds, all states become known and the policy is optimal. Let the policy at step t be f^t. We evaluate the policy f^t in the true product-MDP and plot the state value (the probability of satisfying the specification from that state under policy f^t) $U^{f^t}_{\mathcal{M}}(q_i) : i = 0, \ldots, 7$, for the finite horizon in Fig. 3(a). Note that, even though the policy computed at step $t = 204468$ has already converged to the optimal policy, it is only after $t = 274968$ that in the system's hypothesis, the T-step state value computed using the known MDP under its T-step optimal policy is ε-close to the optimal state value in the true product MDP.

For $\varepsilon = 0.02$, in 136403 steps with 17.73 seconds, all states become known and the policy is optimal. For $\varepsilon = 0.05$, in 55321 steps with 7.18 seconds all states are known. However, the policy f outputs α for all states except q_5, at which it outputs β. Comparing to the optimal policy which outputs β for both q_0 and q_5, f is sub-optimal in the true MDP $\mathcal{M}$: With f, $U^f_{\mathcal{M}}(q_0) = 0.22$, comparing to 0.22445 with the optimal policy. For the remaining states, we obtain the same state values with policy f as the optimal one.

Finally, in three experiments, it is observed that the actual maximal error (0.00445 with $\varepsilon = 0.05$) never exceeds 0.01, because we use the loose upper bound on the error between the T-step state value with any policy in $\mathcal{M}$ and its approximation $\overline{\mathcal{M}}$ in Lemma 1, to guarantee the correctness of the solution.

B. A motion planning example

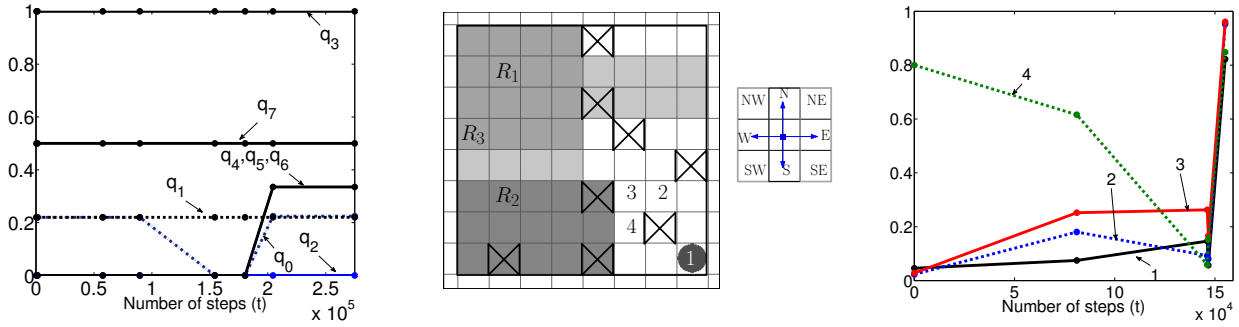
We apply the algorithm to a robot motion planning problem (see Fig. 3(b)). The environment consists of four different unknown terrains: Pavement, grass, gravel and sand. In each terrain and for robot's different action (heading north ('N'), south ('S'), west ('W') and east ('E')), the probability of arriving at the correct cell is in certain ranges: $[0.9, 0.95]$ for pavement, $[0.85, 0.9]$ for grass, $[0.8, 0.85]$ for gravel and $[0.75, 0.80]$ for sand. With a relatively small probability, the robot will arrive at the cell adjacent to the intended one. For example, with action 'N', the intended cell is the one to the north ('N'), whose the adjacent ones are the northeast ('NE')and northwest cells ('NW') (see Fig. 3(b)). The objective of the robot is to maximize the probability of satisfying a temporal logic specification $\varphi = \square\lozenge(R_1 \wedge \lozenge(R_2 \wedge \lozenge R_3)) \wedge \square\neg R_4$ where R_1, R_2, R_3 are critical surveillance cells and R_4 includes a set of unsafe cells to be avoided. For illustrating the effectiveness of the algorithm, we mark a subset of cells labeled by $1, 2, 3, 4$ and evaluate the performance of iteratively updated policies given that a cell in the set is the initial location of the robot.

Given $\varepsilon = 0.01$, $\delta = 0.05$, and $T = 50$, all states become known in 155089 steps and 1593.45 seconds, and the policy updated four times (one for each terrain type). It is worth mentioning that most of the computation time is spent on computing the set of bottom strongly connected components using the algorithm in [15] in the structure of the learned MDP, which is then used to determine the set of accepting end components in $\overline{\mathcal{M}}$. In Fig. 3, we plot the state value $U^{f^t}_{\mathcal{M}}((q_0, s_0))$ where $q_0 \in \{1, 2, 3, 4\}$ and $s_0 = T_s(I_s, q_0)$ for a finite time horizon. The policy output by Algorithm 1 is the optimal policy in the true MDP. The video demonstration for this example is available at http://goo.gl/rVMkrT.

VI. Conclusion and future work

We presented a PAC-MDP method for synthesis with temporal logic constraints in unknown MDPs and developed an algorithm that integrates learning and control for obtaining approximately optimal policies for temporal logic constraints with polynomial time, space and sample complexity. Our current work focuses on other examples (e.g. multi-vehicle motion planning), comparison to alternative, possibly ad hoc methods, and implementing a version of Algorithm 1 that circumvents the need for the input T following [5].

There are a number of interesting future extensions. First, although here we only considered one-player stochastic games, it is also possible to extend to two-player stochastic games, similar to the R-MAX algorithm [5]. Second, for safety critical robotics application, we might need to design the strategy that does not violate some safety constraints during exploration and exploitation. A critical problem is that by enforcing the safety constraints, one might limit the data acquired with exploration and thus only converge to a sub-optimal policy. In this case, we should strike a balance between safety and the optimality in the learned strategy. Third, besides the objective of maximizing the probability of satisfying a temporal logic constraint, other objectives can be considered, for example, minimizing the weighted average costs [21]. Fourth, the method is model-based in the sense that a hypothesis for the underlying MDP is maintained. The advantage in such a model-based approach is that when the control objective is changed, the knowledge gained in the past can be re-used in the policy synthesis for the new objective. However, model-free PAC-MDP approach [22], in which information on the policy is retained directly instead of the transition probabilities, can be of interests as its space-complexity is asymptotically less than the space requirement for model-based approaches.



(a) (b) (c)

Fig. 3. (a) The state value $U_{\mathcal{M}}^{f^t}(q_i)$, $i = 0, \ldots, 7$ v.s. step t, with $\varepsilon = 0.01$, $\delta = 0.05$ and $T = 15$. The markers represents the steps when the policy is recomputed. Note that, for states q_0 and q_1, the state values when all states become known are 0.22445 and 0.22 respectively, which are indiscernible from the figure. (b) Left: A 10×10 gridworld, where the disk represents the robot, the cells R_1, R_2, and R_3 are the interested regions, the crossed cells are the obstacles, labeled with R_4, the cells on the edges are walls, and we assume that if the robot hits the wall, it will be bounced back to the previous cell. Different grey scales represents different terrains: From the darkest to the lightest, these are "grass," " pavement," "sand" and "gravel." Right: The transitions of the robot, in which the center cell is the current location of the robot. (c) The state value $U_{\mathcal{M}}^{f^t}((q_0, s_0))$ v.s. step t, where $q_0 \in \{1, 2, 3, 4\}$ is the initial cell and $s_0 = T_s(I_s, L(q_0))$, under $\epsilon = 0.01$, $\delta = 0.05$ and $T = 50$. The markers represents the steps when the policy is recomputed.

Acknowledgements: The authors would like to thank Laura R. Humphrey of the AFRL for helpful comments on improving the paper. This work is supported by ONR award number 14-13-1-0778.

APPENDIX

Proof of Lemma 1: By Definition 6, M and $\overline{M}$ share the same structure. Thus, for any DRA $\mathcal{A}_\varphi$, the product MDPs $\mathcal{M} = M \ltimes \mathcal{A}_\varphi$ and $\overline{\mathcal{M}} = \overline{M} \ltimes \mathcal{A}_\varphi$ share the same structure and the same set of accepting end states $\mathcal{C} \subseteq V$.

For any policy f, let M_i be the Markov chains obtained from the induced Markov chains $\mathcal{M}^f$ and $\overline{\mathcal{M}}^f$ in the following way: Start at v and for the first i transitions, the transition probabilities are the same as in $\overline{\mathcal{M}}^f$, and for the rest of steps, the transition probabilities are the same as in $\mathcal{M}^f$. Clearly, $\mathcal{M}^f = M_0$ and $\overline{\mathcal{M}}^f = M_T$. For notational simplicity, we denote $h_{M_i}(\cdot) = h_i(\cdot)$, $\Pr_{M_i}(\cdot) = \Pr_i(\cdot)$. Then, we have that

$$\begin{aligned}
&\left|U_{\mathcal{M}}^f(v,T) - U_{\overline{\mathcal{M}}}^f(v,T)\right| = \left|h_0^{\leq T}(v,\mathcal{C}) - h_T^{\leq T}(v,\mathcal{C})\right| \\
&= \Big|h_0^{\leq T}(v,\mathcal{C}) - h_1^{\leq T}(v,\mathcal{C}) + h_1^{\leq T}(v,\mathcal{C}) - h_2^{\leq T}(v,\mathcal{C}) + \ldots + \\
&\quad h_{T-1}^{\leq T}(v,\mathcal{C}) - h_T^{\leq T}(v,\mathcal{C})\Big| = \sum_{i=0}^{T-1}\left|h_i^{\leq T}(v,\mathcal{C}) - h_{i+1}^{\leq T}(v,\mathcal{C})\right| \\
&\leq T \cdot \max_{i\in\{0,\ldots T-1\}}\left|h_i^{\leq T}(v,\mathcal{C}) - h_{i+1}^{\leq T}(v,\mathcal{C})\right|. \qquad (2)
\end{aligned}$$

For any $i = 0, \ldots, T-1$, we have that

$$\begin{aligned}
\mathsf{Diff1} &= \left|h_i^{\leq T}(v,\mathcal{C}) - h_{i+1}^{\leq T}(v,\mathcal{C})\right| = \Big|h_i^{\leq i}(v,\mathcal{C}) \\
&\quad + \sum_{k=i+1}^{T} h_i^k(v,\mathcal{C}) - h_{i+1}^{\leq i}(v,\mathcal{C}) - \sum_{k=i+1}^{T} h_{i+1}^k(v,\mathcal{C})\Big|.
\end{aligned}$$

Since for the first i steps, the transition probabilities in M_i and M_{i+1} are the same, then the probabilities of hitting the set $\mathcal{C}$ in M_i and M_{i+1} equal to the probability of hitting $\mathcal{C}$ in $\mathcal{M}^f = M_0$, i.e., $h_i^{\leq i}(v,\mathcal{C}) = h_{i+1}^{\leq i}(v,\mathcal{C}) = h_0^{\leq i}(v,\mathcal{C})$. Remind that $\Pr_i(x)$, for some $x \in V^*$, is the probability of path x occurring in M_i, as a consequence,

$$\begin{aligned}
\mathsf{Diff1} &= \left|\sum_{k=i+1}^{T} h_i^k(v,\mathcal{C}) - \sum_{k=i+1}^{T} h_{i+1}^k(v,\mathcal{C})\right| \\
&= \Bigg|\sum_{v'\notin\mathcal{C}} \Pr_i(xv') \sum_{v''\notin\mathcal{C}} \left(\Pr_i(v'v'') \cdot h_i^{\leq T-i-1}(v'',\mathcal{C})\right) - \\
&\quad \sum_{v'\notin\mathcal{C}} \Pr_{i+1}(xv') \sum_{v''\notin\mathcal{C}} \left(\Pr_{i+1}(v'v'') \cdot h_{i+1}^{\leq T-i-1}(v'',\mathcal{C})\right)\Bigg|,
\end{aligned}$$

where $x \in V^*$ is a path of length $i-1$ that starts in v and does not contain any state in $\mathcal{C}$. Note that for the first i (resp. the last $T-i-1$) transitions, the transition probabilities in M_i and M_{i+1} are the same as these in $\mathcal{M}^f = M_0$ (resp. $\overline{\mathcal{M}}^f = M_T$), thus we have $\Pr_i(xv') = \Pr_{i+1}(xv') = \Pr_0(xv')$ and $h_i^{\leq T-i-1}(v'',\mathcal{C}) = h_{i+1}^{\leq T-i-1}(v'',\mathcal{C}) = h_T^{\leq T-i-1}(v'',\mathcal{C})$.

Let $v' = (q', s')$, $v'' = (q'', s'')$ and $a = f(v')$. It is also noted that $\Pr_i(v'v'') = \Pr_i((q',s')(q'',s'')) = P(q',a,q'')$ with $s'' = T_s(s', L(q''))$ and $\Pr_{i+1}(v'v'') = \Pr_{i+1}((q',s')(q'',s'')) = \overline{P}(q',a,q'')$. Thus, as $\overline{M}$ approximate M, we have

$$\begin{aligned}
\mathsf{Diff1} &= \sum_{v'\notin\mathcal{C}} \Pr_0(xv') \sum_{v''\notin\mathcal{C}} \big(\left|P(q',a,q'') - \overline{P}(q',a,q'')\right| \\
&\quad \cdot h_T^{\leq T-i-1}(v'',\mathcal{C})\big) \\
&\leq \sum_{v'\notin\mathcal{C}} \Pr_0(xv') \cdot \frac{\epsilon}{NT} \cdot \sum_{v''\notin\mathcal{C}} h_T^{\leq T-i-1}(v'',\mathcal{C}) = \mathsf{Diff2},
\end{aligned}$$

The first term $\sum_{v'\notin\mathcal{C}} \Pr_0(xv') \leq 1$ and the last term is the sum of the probabilities of visiting $\mathcal{C}$ from different states in $V \setminus \mathcal{C}$ within $T-i-1$ steps, each of which is bounded by 1. Moreover, since $v'' = (q'', s'')$ where s'' is determined by the previous state s' in $\mathcal{A}_\varphi$ and the current state q'', i.e., $s'' = T_s(s', L(q''))$, the sum is bounded by the number N of states (choices for q'') in the underlying MDP M. Thus, $\mathsf{Diff1} \leq \mathsf{Diff2} \leq \frac{\epsilon}{NT} \cdot N = \frac{\epsilon}{T}$. Finally, from (2), we have $\left|U_{\mathcal{M}}^f(v,T) - U_{\overline{\mathcal{M}}}^f(v,T)\right| \leq \frac{\epsilon}{T} \cdot T = \epsilon$. ■

References

[1] M. Lahijanian, S. B. Andersson, and C. Belta, "A probabilistic approach for control of a stochastic system from LTL specifications," in *Proceedings of the Conference on Decision and Control and Chinese Control Conference*, pp. 2236–2241, 2009.

[2] A. Medina Ayala, S. B. Andersson, and C. Belta, "Probabilistic control from time-bounded temporal logic specifications in dynamic environments," in *Proceedings of the International Conference on Robotics and Automation*, pp. 4705–4710, 2012.

[3] B. Johnson and H. Kress-Gazit, "Analyzing and revising high-level robot behaviors under actuator error," in *Proceedings of the Intelligent Robots and Systems*, pp. 741–748, 2013.

[4] M. Kearns and S. Singh, "Near-optimal reinforcement learning in polynomial time," *Machine Learning*, vol. 49, no. 2-3, pp. 209–232, 2002.

[5] R. Brafman and M. Tennenholtz, "R-MAX-a general polynomial time algorithm for near-optimal reinforcement learning," *The Journal of Machine Learning*, vol. 3, pp. 213–231, 2003.

[6] A. Legay, B. Delahaye, and S. Bensalem, "Statistical model checking: An overview," in *Runtime Verification*, pp. 122–135, Springer, 2010.

[7] D. Henriques, J. G. Martins, P. Zuliani, A. Platzer, and E. M. Clarke, "Statistical model checking for Markov decision processes," *Proceedings of the International Conference on Quantitative Evaluation of Systems*, pp. 84–93, 2012.

[8] Y. Chen, J. Tumova, and C. Belta, "LTL robot motion control based on automata learning of environmental dynamics," in *Proceedings of the International Conference on Robotics and Automation*, pp. 5177–5182, 2012.

[9] J. Fu, H. Tanner, and J. Heinz, "Adaptive planning in unknown environments using grammatical inference," in *Proceedings of the Decision and Control Conference*, pp. 5357–5363, 2013.

[10] H. Mao, Y. Chen, M. Jaeger, T. D. Nielsen, K. G. Larsen, and B. Nielsen, "Learning Markov decision processes for model checking," in *Proceedings of Quantities in Formal Methods* (U. Fahrenberg, A. Legay, and C. R. Thrane, eds.), Electronic Proceedings in Theoretical Computer Science, pp. 49–63, 2012.

[11] C. Baier, M. Größer, M. Leucker, B. Bollig, and F. Ciesinski, "Controller synthesis for probabilistic systems (extended abstract)," in *Exploring New Frontiers of Theoretical Informatics* (J.-J. Levy, E. Mayr, and J. Mitchell, eds.), vol. 155 of *International Federation for Information Processing*, pp. 493–506, Springer US, 2004.

[12] J. Rutten, M. Kwiatkowska, G. Norman, and D. Parker, *Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems,* P. Panangaden and F. van Breugel (eds.), vol. 23 of *CRM Monograph Series*. American Mathematical Society, 2004.

[13] A. Bianco and L. De Alfaro, "Model checking of probabilistic and nondeterministic systems," in *Foundations of Software Technology and Theoretical Computer Science*, pp. 499–513, Springer, 1995.

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

[15] K. Chatterjee and M. Henzinger, "Faster and dynamic algorithms for maximal end-component decomposition and related graph problems in probabilistic verification," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pp. 1318–1336, 2011.

[16] M. O. Duff, *Optimal Learning: Computational Procedures for Bayes-adaptive Markov Decision Processes*. PhD thesis, University of Massachusetts Amherst, 2002.

[17] T. Wang, D. Lizotte, M. Bowling, and D. Schuurmans, "Bayesian sparse sampling for on-line reward optimization," in *Proceedings of the International Conference on Machine Learning*, pp. 956–963, 2005.

[18] P. S. Castro and D. Precup, "Using linear programming for Bayesian exploration in Markov decision processes," in *Proceedings of the International Joint Conferences on Artificial Intelligence* (M. M. Veloso, ed.), pp. 2437–2442, 2007.

[19] N. Balakrishnan and V. B. Nevzorov, *A Primer on Statistical Distributions*. Wiley, 2004.

[20] B. L. Mark and W. Turin, *Probability, Random Processes, and Statistical Analysis*. Cambridge University Press Textbooks, 2011.

[21] E. M. Wolff, U. Topcu, and R. M. Murray, "Optimal control with weighted average costs and temporal logic specifications.," in *Robotics: Science and Systems*, 2012.

[22] A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman, "PAC model-free reinforcement learning," in *Proceedings of the International Conference on Machine Learning*, pp. 881–888, 2006.