2016 IEEE 55th Conference on Decision and Control (CDC)
ARIA Resort & Casino
December 12-14, 2016, Las Vegas, USA

Q-Learning for Robust Satisfaction of Signal Temporal Logic Specifications

Derya Aksaray, Austin Jones, Zhaodan Kong, Mac Schwager, and Calin Belta

***Abstract*— This paper addresses the problem of learning optimal policies for satisfying signal temporal logic (STL) specifications by agents with unknown stochastic dynamics. The system is modeled as a Markov decision process, in which the states represent partitions of a continuous space and the transition probabilities are unknown. We formulate two synthesis problems where the desired STL specification is enforced by maximizing the probability of satisfaction, and the expected robustness degree, that is, a measure quantifying the quality of satisfaction. We discuss that Q-learning is not directly applicable to these problems because, based on the quantitative semantics of STL, the probability of satisfaction and expected robustness degree are not in the standard objective form of Q-learning. To resolve this issue, we propose an approximation of STL synthesis problems that can be solved via Q-learning, and we derive some performance bounds for the policies obtained by the approximate approach. The performance of the proposed method is demonstrated via simulations.**

I. Introduction

This paper addresses the problem of controlling a system with unknown, stochastic dynamics to achieve a complex, time-sensitive task. We consider tasks given in terms of temporal logic (TL) [5] that can be used to reason about how the state of a system evolves over time. Recently, there has been a great interest in control synthesis with TL specifications (e.g., [3], [4], [9], [20], [17]). When a stochastic dynamical model is known, there exist algorithms to find control policies for maximizing the probability of achieving a given TL specification (e.g., [17], [15]) by planning over stochastic abstractions (e.g., [14], [1], [17]). However, only a few papers have considered the problem of enforcing TL specifications to a system with unknown dynamics. For example, reinforcement learning has been used to find a policy maximizing the probability of satisfying a given linear temporal logic (LTL) formula in [6], [20].

In contrast to existing works on reinforcement learning using propositional temporal logic, we consider signal temporal logic (STL), a rich predicate logic that can describe tasks involving bounds on physical parameters and time intervals [11]. STL is also endowed with a metric called *robustness degree* that quantifies how strongly a given trajectory satisfies an STL formula as a real number rather than just providing a *yes* or *no* answer [12], [11]. This measure enables the use of optimization methods to solve inference (e.g., [16]) or formal synthesis problems (e.g., [19]) involving STL.

In this paper, we formulate two problems that enforce a desired STL specification by maximizing 1) the probability of satisfaction and 2) the expected robustness degree. One of the difficulties in solving these problems is the history-dependence of the satisfaction. For instance, if the specification requires visiting region A before region B, whether or not the system should move towards region B depends on whether or not it has previously visited region A. For LTL formulae with time-abstract semantics, this history-dependence can be broken by translating the formula to a deterministic Rabin automaton, e.g., [20]. In the case of STL, such a construction is difficult due to the time-bounded semantics. We circumvent this issue by defining a fragment of STL such that the progress towards satisfaction is checked with a sufficient number of state measurements. We thus define a Markov decision process (MDP), called the τ-MDP, whose states correspond to the τ-step history of the system and the actions are from a finite set of motion primitives.

Even though the history dependence issue can be solved by defining a τ-MDP, a reinforcement learning strategy such as Q-learning [22] is still not applicable to maximize probability of satisfaction or expected robustness degree. In Q-learning, an agent tries an action, observes an immediate reward, and updates its policy to maximize the sum of rewards. However, based on the quantitative semantics of STL, the objective functions such as probability of satisfaction or expected robustness degree are not in the standard form of Q-learning. Thus, we propose an approximation of these functions such that the new synthesis problems can be solved via Q-learning. Moreover, we provide some performance bounds for the approximate solutions, which can be sufficiently close to the actual solutions with a proper selection of the approximation parameter. Finally, we demonstrate the performance of the proposed approach through simulation case studies.

II. Preliminaries: Signal Temporal Logic (STL)

In this paper, the desired system behavior is described by an STL fragment with the following *syntax* :

$$\begin{aligned} \Phi &:= F_{[a,b]}\phi | G_{[a,b]}\phi \\ \phi &:= F_{[c,d]}\varphi | G_{[c,d]}\varphi \\ \varphi &:= \psi | \neg\varphi | \varphi \wedge \varphi | \varphi \vee \varphi, \end{aligned} \tag{1}$$

*This work was partially supported at Boston University by ONR grant number N00014-14-1-0554 and by the NSF grant numbers CMMI-1400167, NSF NRI-1426907. D. Aksaray is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA. daksaray@mit.edu. A. Jones is with the Departments of Mechanical Engineering and Electrical Engineering, Georgia Institute of Technology, Atlanta, GA. austinjones@gatech.edu. Z. Kong is with the Department of Mechanical and Aerospace Engineering, University of California Davis, Davis, CA. zdkong@ucdavis.edu. M. Schwager is with the Department of Aeronautics and Astronautics, Stanford University, Stanford, CA. schwager@stanford.edu. C. Belta is with the Department of Mechanical Engineering, Boston University, Boston, MA. cbelta@bu.edu.

where $a,b,c,d \in \mathbb{R}_{\geq 0}$ are finite non-negative time bounds; Φ, ϕ, and φ are STL formulae; ψ is a predicate in the form of $f(\mathbf{s}) < d$ where $\mathbf{s} : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is a signal, $f : \mathbb{R}^n \to \mathbb{R}$ is a function, and $d \in \mathbb{R}$ is a constant. The Boolean operators $\neg$, $\wedge$, and $\vee$ are negation, conjunction (i.e., *and*), and disjunction (i.e., *or*), respectively. The temporal operators F and G refer to *Finally* (i.e., eventually) and *Globally* (i.e., always), respectively.

For any signal $\mathbf{s}$, let s_t denote the value of $\mathbf{s}$ at time t and let $(\mathbf{s},t)$ be the part of the signal that is a sequence of $s_{t'}$ for $t' \in [t,\infty)$. Accordingly, the *Boolean semantics* of STL is recursively defined as follows:

$$
\begin{array}{lcl}
(\mathbf{s},t) \models (f(\mathbf{s}) < d) & \Leftrightarrow & f(s_t) < d, \\
(\mathbf{s},t) \models \neg(f(\mathbf{s}) < d) & \Leftrightarrow & \neg\big((\mathbf{s},t) \models (f(\mathbf{s}) < d)\big), \\
(\mathbf{s},t) \models \phi_1 \wedge \phi_2 & \Leftrightarrow & (\mathbf{s},t) \models \phi_1 \text{ and } (\mathbf{s},t) \models \phi_2, \\
(\mathbf{s},t) \models \phi_1 \vee \phi_2 & \Leftrightarrow & (\mathbf{s},t) \models \phi_1 \text{ or } (\mathbf{s},t) \models \phi_2, \\
(\mathbf{s},t) \models G_{[a,b]}\phi & \Leftrightarrow & (\mathbf{s},t') \models \phi \quad \forall t' \in [t+a,t+b], \\
(\mathbf{s},t) \models F_{[a,b]}\phi & \Leftrightarrow & \exists t' \in [t+a,t+b] \text{ s.t. } (\mathbf{s},t') \models \phi.
\end{array}
$$

For a signal $(\mathbf{s},0)$, i.e., the whole signal starting from time 0, satisfying $F_{[a,b]}\phi$ means that "there exists a time within $[a,b]$ such that ϕ will eventually be true", and satisfying $G_{[a,b]}\phi$ means that "ϕ is true for all times between $[a,b]$".

STL is endowed with a metric called *robustness degree* [12], [11] (also called "degree of satisfaction") that quantifies how well a given signal $\mathbf{s}$ satisfies a given formula Φ. The robustness degree is calculated recursively according to the *quantitative semantics:*

$$
\begin{array}{rcl}
r(\mathbf{s},(f(\mathbf{s}) < d),t) & = & d - f(s_t), \\
r(\mathbf{s},\neg(f(\mathbf{s}) < d),t) & = & -r(\mathbf{s},(f(\mathbf{s}) < d),t), \\
r(\mathbf{s},\phi_1 \wedge \phi_2,t) & = & \min\big(r(\mathbf{s},\phi_1,t), r(\mathbf{s},\phi_2,t)\big), \\
r(\mathbf{s},\phi_1 \vee \phi_2,t) & = & \max\big(r(\mathbf{s},\phi_1,t), r(\mathbf{s},\phi_2,t)\big), \\
r(\mathbf{s},G_{[a,b]}\phi,t) & = & \min\limits_{t' \in [t+a,t+b]} r(\mathbf{s},\phi,t'), \\
r(\mathbf{s},F_{[a,b]}\phi,t) & = & \max\limits_{t' \in [t+a,t+b]} r(\mathbf{s},\phi,t').
\end{array}
$$

As a short-hand notation, $r(\mathbf{s},\phi)$ refers to $r(\mathbf{s},\phi,0)$ throughout the paper. Let ε-perturbation be a sequence of disturbances such that any signal under ε-perturbation stays inside the ε-envelope. Note that $r(\mathbf{s},\phi) = \varepsilon > 0$ means that $\mathbf{s}$ satisfies ϕ. Moreover, the signal $\mathbf{s}$ under ε-perturbation still satisfies ϕ. Similarly, $r(\mathbf{s},\phi) = \varepsilon < 0$ means that $\mathbf{s}$ violates ϕ, and $\mathbf{s}$ under ε-perturbation still violates ϕ.

As in [10], let $hrz(\phi)$ denote the *horizon length* of an STL formula ϕ, which is the required number of samples to resolve any (future or past) requirements of ϕ. The horizon length can be computed recursively as

$$
\begin{array}{rcl}
hrz(\psi) & = & 0, \\
hrz(\phi) & = & b \qquad \text{if } \phi = G_{[a,b]}\psi \text{ or } F_{[a,b]}\psi, \\
hrz(F_{[a,b]}\phi) & = & b + hrz(\phi), \\
hrz(G_{[a,b]}\phi) & = & b + hrz(\phi), \\
hrz(\neg\phi) & = & hrz(\phi), \\
hrz(\phi_1 \wedge \phi_2) & = & \max(hrz(\phi_1), hrz(\phi_2)), \\
hrz(\phi_1 \vee \phi_2) & = & \max(hrz(\phi_1), hrz(\phi_2)),
\end{array}
$$

where $a,b \in \mathbb{R}_{\geq 0}$, ψ is a predicate, and ϕ, ϕ_1, ϕ_2 are STL formulae.

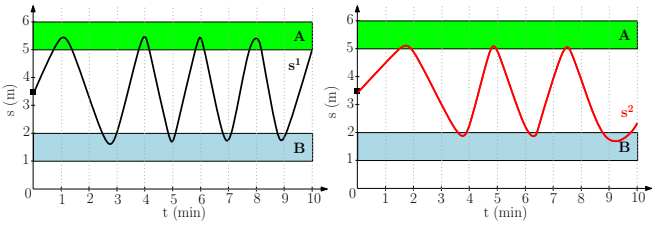


Fig. 1. The specification is "visit regions A and B every 3 minutes along a mission horizon of 10 minutes", i.e., $\Phi = G_{[0,7]}(F_{[0,3]}(s > 5 \wedge s < 6) \wedge F_{[0,3]}(s > 1 \wedge s < 2))$, which is satisfied by signal $\mathbf{s^1}$ and violated by signal $\mathbf{s^2}$.

Example 1: Consider the regions A and B illustrated in Fig. 1 and a specification as "visit regions A and B every 3 minutes along a mission horizon of 10 minutes". Note that the desired specification can be formulated in STL as

$$
\begin{array}{rl}
\Phi = & G_{[0,7]}\phi \\
\phi = & F_{[0,3]}(s > 5 \wedge s < 6) \wedge F_{[0,3]}(s > 1 \wedge s < 2).
\end{array} \tag{2}
$$

The horizon lengths of Φ and ϕ are $hrz(\Phi) = 10$ and $hrz(\phi) = 3$, respectively. Let $\psi_1 = (s > 5 \wedge s < 6)$ and $\psi_2 = (s > 1 \wedge s < 2)$. Then satisfying Φ implies satisfying $\bigwedge\limits_{t \in [0,7]} (F_{[t,t+3]}\psi_1 \wedge F_{[t,t+3]}\psi_2)$. Let $\mathbf{s^1}$ and $\mathbf{s^2}$ be two signals as illustrated in Fig. 1. The signal $\mathbf{s^1}$ satisfies Φ because A and B are visited within $[t,t+3]$ for every $t \in [0,7]$. However, the signal $\mathbf{s^2}$ violates Φ because region B is not visited within $[0,3]$. Moreover, the robustness degree of $\mathbf{s}$ with respect to Φ can be computed via the quantitative semantics as follows:

$$
\min_{t \in [0,7]} \min \left\{ \max_{t' \in [t,t+3]} r(\mathbf{s},\psi_1,t'), \max_{t' \in [t,t+3]} r(\mathbf{s},\psi_2,t') \right\} \tag{3}
$$

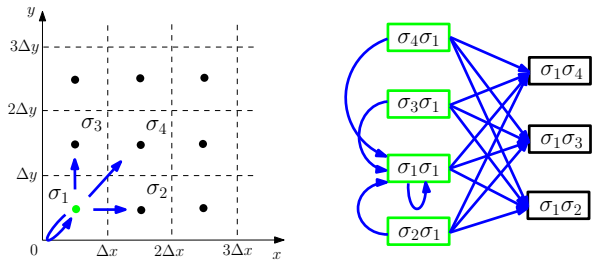
Based on (3), the robustness degrees of $\mathbf{s^1}$ and $\mathbf{s^2}$ with respect to Φ are computed as $r(\mathbf{s^1},\Phi) = 0.35$ and $r(\mathbf{s^2},\Phi) = -1$ indicating that $\mathbf{s^1}$ satisfies Φ while $\mathbf{s^2}$ does not.

III. Problem Formulation

A. System Model

We consider a system as a Markov decision process (MDP) $M = \langle \Sigma, A, P, R \rangle$, where Σ denotes the state-space, A is a finite set of motion primitives, $P : \Sigma \times A \times \Sigma \to [0,1]$ is a probabilistic transition relation, and $R : \Sigma \to \mathbb{R}$ is a reward function. We assume that Σ comprises a set of partitions and each $\sigma_i \in \Sigma$ corresponds to the centroid of a partition, e.g., $\sigma_1 = (\Delta x/2, \Delta y/2)$ in Fig. 2(a). Moreover, each motion primitive $a \in A$ drives the system from the current state σ_i to an adjacent state σ_j. Let $s_t \in \Sigma$ denote the state of a system at time t, and let $s_{t_1:t_2}$ denote the state trajectory of the system within $[t_1,t_2]$. Suppose that a system moves in an environment shown in Fig. 2(a), and its initial state is $s_0 = \sigma_1$. If the system visits σ_3 and returns to σ_1, its state trajectory can be written as $s_{0:2\Delta t} = \sigma_1\sigma_3\sigma_1$ where $\Delta t > 0$ is the discrete time step.

In this paper, we assume that the MDP model is already available. For more generic cases, abstractions of stochastic systems can be constructed via several methods (e.g., [14], [1], [17]). Moreover, if the system satisfies certain conditions, a discrete-time signal can be used to reason about whether or



(a) (b)

Fig. 2. (a) Discretized state-space, (b) Representation of σ_1 over $2-MDP$.

not the continuous-time signal satisfies a TL formula (e.g., [13], [8]).

B. Problem Definition

In real-world applications, many systems (e.g., robotic systems) contain uncertainty in their dynamics due to mechanical, environmental, or sensing issues. In this aspect, we consider an MDP M, for which the transition probability function P is unknown. This means that when the system executes a motion primitive a at state s_t, it is not certain where it will be in $t+1$, i.e., the probability distribution for s_{t+1} is unknown. Then, the question becomes how to enforce an STL specification Φ to a system with unknown dynamics.

In this paper, we formulate two problems that have different objective functions to find a control policy π enforcing the desired specification Φ. In the first problem, we maximize the probability of satisfying Φ, which is a commonly used objective in formal synthesis problems (e.g., [15], [17], [9]). In the second problem, we maximize the expected robustness degree with respect to Φ, which has recently been used in model predictive control framework (e.g., [19]).

Problem 1 (Maximizing Probability of Satisfaction): Let Φ be an STL specification with $hrz(\Phi) = T$. Given a stochastic model $M = \langle \Sigma, A, P, R \rangle$ with unknown P, known reward function R, and an initial partial state trajectory $s_{0:\tau}$ for some $\tau \in [0,T)$, find a control policy

$$\pi_1^* = \arg\max_{\pi} Pr^{\pi}[s_{0:T} \models \Phi] \tag{4}$$

where $Pr^{\pi}[s_{0:T} \models \Phi]$ is the probability of $s_{0:T}$ satisfying Φ under policy π.

Problem 2 (Maximizing Expected Robustness Degree): Let Φ be an STL specification with $hrz(\Phi) = T$. Given a stochastic model $M = \langle \Sigma, A, P, R \rangle$ with unknown P, known reward function R, and an initial partial state trajectory $s_{0:\tau}$ for some $\tau \in [0,T)$, find a control policy

$$\pi_2^* = \arg\max_{\pi} E^{\pi}[r(s_{0:T}, \Phi)] \tag{5}$$

where $E^{\pi}[r(s_{0:T}, \Phi)]$ is the expected robustness degree of $s_{0:T}$ with respect to Φ under policy π.

IV. Control Synthesis via Q-learning

For systems with unknown stochastic dynamics, reinforcement learning can be used to design optimal control policies, that is, the system learns how to take actions by trial and error interactions with the environment. In this paper, we use the Q-learning algorithm that is briefly presented in the first subsection. Then, we discuss that Problems 1 and 2 are not in the standard form to apply this algorithm. Finally, we present the main contribution of this paper, i.e., the approximation of STL synthesis problems that can be solved via Q-learning.

A. Q-learning

Q-learning is a model-free reinforcement learning method [22], which can be used to find the optimal policy for a finite MDP. In particular, the objective of an agent at state s_t is to maximize $V(s_t)$, its expected (discounted) reward in finite or infinite horizon, i.e.,

$$E\Big[\sum_{k=0}^{T} r(s_{k+t+1})\Big] \quad \text{or} \quad E\Big[\sum_{k=0}^{\infty} \gamma^k r(s_{k+t+1})\Big], \tag{6}$$

where $r(s)$ is the reward obtained at state s, and γ is the discount factor. Also, $V^*(s) = \max_a Q^*(s,a)$, where $Q^*(s,a)$ is the optimal Q-function for every state-action pair (s,a).

Starting from state s, the system chooses an action a, which takes it to state s' and results in a reward r. Then, the Q-learning rule is defined as follows:

$$Q(s,a) := (1-\alpha)Q(s,a) + \alpha[r + \gamma \max_{a^* \in A} Q(s', a^*)], \tag{7}$$

where $\gamma \in (0,1)$ is the discount factor and $\alpha \in (0,1]$ is the learning rate. Accordingly, if each action $a \in A$ is repetitively implemented in each state $s \in \Sigma$ for infinite number of times and α decays appropriately, then Q converges to Q^* with probability 1 (see Theorem 4.1). Thus, we can find the optimal policy $\pi^* : \Sigma \to A$ as $\pi^* = \arg\max_a Q^*(s,a)$.

Theorem 4.1: [21] Given a finite MDP, $M = \langle S, A, P, R \rangle$, let $Q^*(s,a)$ be the optimal Q-function for every pair of (s,a). Consider the Q-learning algorithm with the update rule $Q_{k+1}(s,a) = (1-\alpha_k)Q_k(s,a) + \alpha_k[r + \gamma \max_{a^* \in A} Q_k(s', a^*)]$, where the discount factor $\gamma \in (0,1)$, α_k satisfies $\sum_k \alpha_k = \infty$ and $\sum_k \alpha_k^2 < \infty$. Then $Q_k(s,a)$ converges to $Q^*(s,a)$ with probability 1 as $k \to \infty$.

B. Q-learning and Formal Synthesis

There are several reasons why one cannot directly use Q-learning in Problems 1 and 2. First, the action selection at each time cannot depend only on the current state as in Q-learning. For example, consider a specification $\Phi_1 = F_{[0,T]} x > 3$. Satisfying Φ_1 implies that visiting the desired region at least one time in $[0,T]$. Let the current state be $s_t := x = 2$ and assume that the desired region is not visited before t. Note that if $t = T-1$, then the action selection via the optimal policy leads the agent to maximally approach the desired region. However, if $t = 0$, then the optimal policy may result in an action that drives the agent further away from the desired region (while ensuring to eventually satisfy $x > 3$). Thus, the optimal policies may not necessarily be the same if the same state is occupied but the remaining mission horizons are different. Moreover, if Φ involves a nested temporal operator, the policy should also take into account a sufficient length of state history in addition to the current

state and the remaining mission horizon. Thus, the policies in Problems 1 and 2 should be defined as $\pi : \Sigma^\tau \times \mathbb{N}_{\geq 0} \to A$ where $\Sigma^\tau = \Sigma \times \cdots \times \Sigma$ for τ times.

Second, one can not directly apply Q-learning because an agent trying to optimize (4) or (5) does not have an immediate reward after taking an action. Consider a specification Φ such that $hrz(\Phi) = T$. Accordingly, both satisfaction and the robustness degree can be computed over a T-length trajectory (i.e., these measures are undefined for partial trajectories having a length smaller than T). For example, consider an agent trying to satisfy $\Phi_1 = F_{[0,T]}\psi$. Then, the objective function in Problem 2 can be written as $\max_\pi E^\pi\Big[\max\big(r(s_{0:T},\psi,0),\ldots,r(s_{0:T},\psi,T)\big)\Big]$. Hence, the objective functions in (4) or (5) are not in the standard form of Q-learning as in (6).

C. Proposed Approach

In this paper, we approximate the synthesis problems in (4) and (5) to find the optimal policy via Q-learning. The overview of the proposed approach is: 1) for any STL formula (i.e., $G_{[0,T]}\phi$ or $F_{[0,T]}\phi$), redefine the state-space as Σ^τ where τ is a function of $hrz(\phi)$; 2) redefine the objective function such that an agent observes an immediate reward after taking each action and the remaining mission horizon can be eliminated from the policy as $\pi^* : \Sigma^\tau \to A$.

Let Φ be $G_{[0,T]}\phi$ or $F_{[0,T]}\phi$, where Φ and ϕ are STL formulae with the syntax in (1). Let the horizon length of ϕ be $hrz(\phi) = \tau$. Then, we denote the τ-state of the agent at time t by s_t^τ, which is the τ-horizon trajectory involving the current state and the most recent $\tau-1$ past states, i.e., $s_t^\tau = s_{t-\tau+1:t}$. By considering all τ-states of the agent, we remodel the agent as a τ-MDP.

Definition 1 (τ-MDP): Given an MDP $M = (\Sigma, A, P, R)$ and $\tau \in \mathbb{N}_{>0}$, a τ-MDP is a tuple $M^\tau = (\Sigma^\tau, A, P^\tau, R^\tau)$, where

- $\Sigma^\tau \subseteq (\Sigma \cup \varepsilon)^\tau$ is the set of finite states, where ε is the empty string. Each state $\sigma^\tau \in \Sigma^\tau$ corresponds to a $\tau-$horizon (or shorter) path on Σ. Shorter paths of length $n < \tau$ (i.e., the system has not yet evolved for τ time steps) have ε prepended $\tau - n$ times.
- $P^\tau : \Sigma^\tau \times A \times \Sigma^\tau \to [0,1]$ is a probabilistic transition relation. Let $\sigma_i^\tau = \sigma_a\sigma_b\ldots\sigma_c\sigma_d$ and $\sigma_j^\tau = \sigma_e\ldots\sigma_f\sigma_g$. $P^\tau(\sigma_i^\tau, a, \sigma_j^\tau) > 0$ if and only if $P(\sigma_d, a, \sigma_g) \in [0,1]$ and for $\tau > 1$ the first $\tau-1$ elements of σ_j^τ are equal to the last $\tau-1$ elements of σ_i^τ (i.e., $\sigma_e\ldots\sigma_f = \sigma_b\ldots\sigma_d$).
- $R^\tau : \Sigma^\tau \to \mathbb{R}$ is a reward function.

For instance, the highlighted state σ_1 in Fig. 2(a) corresponds to four τ-states for $\tau = 2$ as illustrated in Fig. 2(b).

For any $\Phi = F_{[0,T]}\phi$ or $\Phi = G_{[0,T]}\phi$, τ can be computed as follows:

$$\tau = \left\lceil \frac{hrz(\phi)}{\Delta t} \right\rceil + 1 \tag{8}$$

where Δt is the time step and $\lceil . \rceil$ is the ceiling function, i.e., $\lceil x \rceil$ is the smallest integer not less than $x \in \mathbb{R}$. Note that if Φ does not have nested temporal operators, then $hrz(\phi) = 0$ and $\tau = 1$ as a consequence. As such, $M^\tau = M$ for $\tau = 1$.

For any state trajectory $s_{0:T}$, we can write the corresponding τ-state trajectory as $s_{\tau-1:T}^\tau = s_{\tau-1}^\tau \ldots s_T^\tau$ where each $s_t^\tau := s_{t-\tau+1:t}$ for $\tau - 1 \leq t \leq T$. Moreover, for each τ-state s_t^τ, we can compute the corresponding robustness degree with respect to ϕ. Accordingly, the robustness degree of $s_{0:T}$ with respect to Φ can be written in terms of τ-states as

$$r(s_{0:T},\Phi) = \begin{cases} \max\big(r(s_{\tau-1}^\tau,\phi),\ldots,r(s_T^\tau,\phi)\big), & \text{if } \Phi = F_{[0,T]}\phi \\ \min\big(r(s_{\tau-1}^\tau,\phi),\ldots,r(s_T^\tau,\phi)\big), & \text{if } \Phi = G_{[0,T]}\phi \end{cases} \tag{9}$$

Note that plugging (9) into (5) makes the objective in Problem 2 as follows:

$$\max_\pi E^\pi[r(s_{0:T},\Phi)] = \begin{cases} \max_\pi E^\pi\big[\max_{\tau-1\leq t\leq T}\big(r(s_t^\tau,\phi)\big)\big], & \text{if } \Phi = F_{[0,T]}\phi \\ \max_\pi E^\pi\big[\min_{\tau-1\leq t\leq T}\big(r(s_t^\tau,\phi)\big)\big], & \text{if } \Phi = G_{[0,T]}\phi \end{cases} \tag{10}$$

Since Q-learning cannot be used for cases like (10), we propose to use the *log-sum-exp* [7] approximation of the max function to represent the objective as a sum of rewards, i.e.,

$$\max(x_1,\ldots,x_n) \sim \frac{1}{\beta}\log\sum_{i=1}^n e^{\beta x_i}, \tag{11}$$

where $\beta > 0$ is a constant and

$$\max(x_1,\ldots,x_n) \leq \frac{1}{\beta}\log\sum_{i=1}^n e^{\beta x_i} \leq \max(x_1,\ldots,x_n) + \frac{1}{\beta}\log n. \tag{12}$$

Based on (11), the equation in (10) can be approximated as

$$\max_\pi E^\pi[r(s_{0:T},\Phi)] \sim \begin{cases} \max_\pi E^\pi\Big[\frac{1}{\beta}\log\sum_{t=\tau-1}^T e^{\beta r(s_t^\tau,\phi)}\Big], & \text{if } \Phi = F_{[0,T]}\phi \\ \max_\pi E^\pi\Big[-\frac{1}{\beta}\log\sum_{t=\tau-1}^T e^{-\beta r(s_t^\tau,\phi)}\Big], & \text{if } \Phi = G_{[0,T]}\phi \end{cases} \tag{13}$$

Similarly, maximizing the probability of satisfying Φ can be written as

$$\max_\pi Pr^\pi\big[s_{0:T} \models \Phi\big] = \max_\pi E^\pi\Big[I\big(r(s_{0:T},\Phi)\big)\Big] \tag{14}$$

where $I(.)$ is the indicator function such that $I(x) = 1$ if $x \geq 0$ and $I(x) = 0$ otherwise. Since $I\big(\max(x_1,\ldots,x_n)\big) = \max\big(I(x_1),\ldots,I(x_n)\big)$ (or $I\big(\min(x_1,\ldots,x_n)\big) = \min\big(I(x_1),\ldots,I(x_n)\big)$), plugging (9) into (14) makes the objective in Problem 1 as follows:

$$\max_\pi Pr^\pi\big[s_{0:T} \models \Phi\big] = \begin{cases} \max_\pi E^\pi\Big[\max_{\tau-1\leq t\leq T} I\big(r(s_t^\tau,\phi)\big)\Big], & \text{if } \Phi = F_{[0,T]}\phi \\ \max_\pi E^\pi\Big[\min_{\tau-1\leq t\leq T} I\big(r(s_t^\tau,\phi)\big)\Big], & \text{if } \Phi = G_{[0,T]}\phi \end{cases} \tag{15}$$

Based on (11), the equation in (15) can be approximated as

$$\max_\pi Pr^\pi\big[s_{0:T} \models \Phi\big] \sim \begin{cases} \max_\pi E^\pi\Big[\frac{1}{\beta}\log\sum_{t=\tau-1}^T e^{\beta I\left(r(s_t^\tau,\phi)\right)}\Big], & \text{if } \Phi = F_{[0,T]}\phi \\ \max_\pi E^\pi\Big[-\frac{1}{\beta}\log\sum_{t=\tau-1}^T e^{-\beta I\left(r(s_t^\tau,\phi)\right)}\Big], & \text{if } \Phi = G_{[0,T]}\phi \end{cases} \tag{16}$$

Problem 1A (Max. Approx. Probability of Satisfaction): Let Φ and ϕ be STL formulae with the syntax in (1)

such that $\Phi = F_{[0,.]}\phi$ or $\Phi = G_{[0,.]}\phi$. Let $hrz(\Phi) = T$. Given an unknown MDP M, let $M^\tau = \langle \Sigma^\tau, A, P^\tau, R^\tau \rangle$ be the τ-MDP where τ is computed as in (8). Assume that the initial τ-state $s^\tau_{\tau-1} = s_{0:\tau-1}$ is given and $\beta > 0$. Find a control policy $\pi^*_{1A} : \Sigma^\tau \to A$ such that

$$\pi^*_{1A} = \begin{cases} \arg\max_\pi E^\pi \Big[\sum_{t=\tau-1}^{T} e^{\beta I(r(s^\tau_t,\phi))} \Big], & \text{if } \Phi = F_{[0,T]}\phi \\ \arg\max_\pi E^\pi \Big[-\sum_{t=\tau-1}^{T} e^{-\beta I(r(s^\tau_t,\phi))} \Big], & \text{if } \Phi = G_{[0,T]}\phi \end{cases} \tag{17}$$

Problem 2A (Max. Expected Approx. Robustness Degree): Let Φ and ϕ be STL formulae with the syntax in (1) such that $\Phi = F_{[0,.]}\phi$ or $\Phi = G_{[0,.]}\phi$. Let $hrz(\Phi) = T$. Given an unknown MDP M, let $M^\tau = \langle \Sigma^\tau, A, P^\tau, R^\tau \rangle$ be the τ-MDP where τ is computed as in (8). Assume that the initial τ-state $s^\tau_{\tau-1} = s_{0:\tau-1}$ is given and $\beta > 0$. Find a control policy $\pi^*_{2A} : \Sigma^\tau \to A$ such that

$$\pi^*_{2A} = \begin{cases} \arg\max_\pi E^\pi \Big[\sum_{t=\tau-1}^{T} e^{\beta r(s^\tau_t,\phi)} \Big], & \text{if } \Phi = F_{[0,T]}\phi \\ \arg\max_\pi E^\pi \Big[-\sum_{t=\tau-1}^{T} e^{-\beta r(s^\tau_t,\phi)} \Big], & \text{if } \Phi = G_{[0,T]}\phi \end{cases} \tag{18}$$

Theorem 4.2: Let Φ be an STL formula with the syntax in (1) and $hrz(\Phi) = T$. Assume that a partial state trajectory $s_{0:\tau-1}$ is initially given where τ is computed as in (8). For some $\beta > 0$ and $\Delta t = 1$, let π^*_1, π^*_2, π^*_{1A}, π^*_{2A} be the optimal policies obtained by solving Problems 1, 2, 1A, 2A, respectively. Then,

$$\begin{aligned} Pr^{\pi^*_1}[s_{0:T} \models \Phi] - \frac{1}{\beta}\log(T-\tau+2) &\leq Pr^{\pi^*_{1A}}[s_{0:T} \models \Phi] \\ E^{\pi^*_2}[r(s_{0:T},\Phi)] - \frac{1}{\beta}\log(T-\tau+2) &\leq E^{\pi^*_{2A}}[r(s_{0:T},\Phi)] \end{aligned}$$

Proof: The proof can be found in [2]. ∎

In the following proposition, we show that Q-learning can be used to solve Problems 1A and 2A.

Proposition 4.3: Let Φ and ϕ be STL formulae such that $\Phi = F_{[0,.]}\phi$ or $\Phi = G_{[0,.]}\phi$. Given a finite MDP M, let $M^\tau = \langle \Sigma^\tau, A, P^\tau, R^\tau \rangle$ be the τ-MDP where τ is computed as in (8) and $Q^*(s^\tau, a)$ is the optimal Q-function for every pair of (s^τ, a). Consider the Q-learning algorithm with the update rule $Q_{k+1}(s^\tau_i, a) = (1-\alpha_k)Q_k(s^\tau_i, a) + \alpha_k[R + \gamma \max_{a^* \in A} Q_k(s^\tau_j, a^*)]$, where s^τ_j is the resulting state by taking action a at s^τ_i, $\gamma \in (0,1)$, α_k satisfies $\sum_k \alpha_k = \infty$ and $\sum_k \alpha_k^2 < \infty$, and for some $\beta > 0$, the immediate reward R obtained at s^τ_j is defined as

$$R = \begin{cases} e^{\beta I(r(s^\tau_j,\phi))}, & \text{if Problem 1A with } \Phi = F_{[0,T]}\phi \\ -e^{-\beta I(r(s^\tau_j,\phi))}, & \text{if Problem 1A with } \Phi = G_{[0,T]}\phi \\ e^{\beta r(s^\tau_j,\phi)}, & \text{if Problem 2A with } \Phi = F_{[0,T]}\phi \\ -e^{-\beta r(s^\tau_j,\phi)}, & \text{if Problem 2A with } \Phi = G_{[0,T]}\phi \end{cases}$$

Then $Q_k(s^\tau, a)$ converges to $Q^*(s^\tau, a)$ w.p.1 as $k \to \infty$.

Proof: The proof follows from Theorem 4.1. ∎

Remark 1: When Q-learning is used to maximize the discounted versions of (17) and (18) as in Proposition 4.3, the approximation bound can be derived from (12) as

$$\max(x_1,\ldots,x_n) + \frac{1}{\beta}\log\gamma^n \leq \frac{1}{\beta}\log\sum_{i=1}^{n}\gamma^i e^{\beta x_i} \leq \max(x_1,\ldots,x_n) + \frac{1}{\beta}\log n.$$

Accordingly, in light of Theorem 4.2, the performances of the policies obtained via Q-learning are bounded as

$$\begin{aligned} Pr^{\pi^*_1}[s_{0:T} \models \Phi] - \frac{1}{\beta}\max\big(\log(T-\tau+2) - \log\gamma^n\big) &\leq Pr^{\pi^*_{1A}}[s_{0:T} \models \Phi] \\ E^{\pi^*_2}[r(s_{0:T},\Phi)] - \frac{1}{\beta}\max\big(\log(T-\tau+2) - \log\gamma^n\big) &\leq E^{\pi^*_{2A}}[r(s_{0:T},\Phi)]. \end{aligned}$$

Consequently, selecting γ close to 1 and arbitrarily large selection of β significantly reduce the performance gap between the solutions obtained via Problems 1 and 1A (2 and 2A). However, larger values of β would increase the maximum reward hence would reduce the convergence rate in Q-learning [18].

V. Simulation Results

We consider an agent moving in an environment illustrated in Fig. 3(b). The set of motion primitives at each state is $A = \{N, NW, W, SW, S, SE, E, NE, stay\}$. We model the motion uncertainty as in Fig. 3(a) where, for any selected feasible action, the agent follows the corresponding blue arrow with probability 0.93 or a red arrow with probability 0.023. Moreover, taking an infeasible action (i.e., moving towards a boundary while next to it) leads the agent to stay at its current state. We consider an STL formula defined over the environment as $\Phi_2 = G_{[0,12]}\big(F_{[0,2]}(region\ A) \wedge F_{[0,2]}(region\ B)\big)$ where *region A* represents $x > 1 \wedge x < 2 \wedge y > 3 \wedge y < 4$ and *region B* represents $x > 2 \wedge x < 3 \wedge y > 2 \wedge y < 3$. Note that Φ_2 expresses the following: "for all $t \in [0,12]$, eventually visit *region A* every $[t, t+2]$ and eventually visit *region B* every $[t, t+2]$". Note that $\Phi_2 = G_{[0,12]}\phi$ where $\phi = F_{[0,2]}(region\ A) \wedge F_{[0,2]}(region\ B)$ and $hrz(\phi) = 2$. Assuming that $\Delta t = 1$, $\tau = 3$ based on (8).

The sizes of the state-spaces are $|\Sigma| = 16$ and $|\Sigma^\tau| = 676$[1] for $\tau = 3$. To implement the Q-learning algorithm, the number of episodes is chosen as 2000 (i.e., $1 \leq k \leq 2000$), and we use the parameters $\beta = 50$, $\gamma = 0.9999$, and $\alpha_k = 0.95^k$. After 2000 trainings, the resulting policies π^*_{1A} and π^*_{2A} are used to generate 500 trajectories, which lead to

$$\begin{aligned} E^{\pi^*_{1A}}[r(s_{0:14},\Phi_2)] = 0.084, \quad Pr^{\pi^*_{1A}}[s_{0:14} \models \Phi_2] = 0.732, \\ E^{\pi^*_{2A}}[r(s_{0:14},\Phi_2)] = 0.422, \quad Pr^{\pi^*_{2A}}[s_{0:14} \models \Phi_2] = 0.936. \end{aligned}$$

Sample trajectories generated by π^*_{1A} and π^*_{2A} are displayed in Fig. 3 (c) and (d), respectively.

The performance difference between two solutions can be explained by what happens when trajectories almost satisfy Φ_2. While solving Problem 1A, if a τ-state slightly violating or strongly violating ϕ is encountered, the overall reward in both cases will be the same. On the other hand, while solving

[1]This indicates that there are 676 partial trajectories with 3 states in the scenario illustrated in in Fig. 3(a), and it is computed by taking into account the admissible 2 transitions at each state in Fig. 3(a).

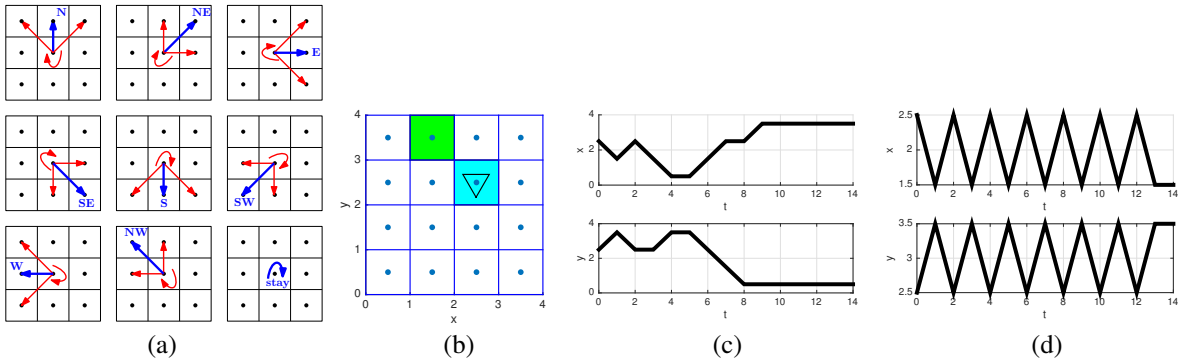


Fig. 3. (a) The motion uncertainty (as red arrows) for a particular action (blue arrow), (b) the initial state and the desired regions for which a sample trajectory by (c) π^*_{1A} and (d) π^*_{2A}.

Problem 2A, the policy producing the slightly violating τ-state will be reinforced much more strongly than an arbitrary policy as the resulting robustness degree is larger. Since the robustness degree gives "partial credit" for trajectories that are close to satisfaction, the Q-learning performs a directed search to find policies that satisfy the formula. Since probability maximization gives no partial credit, the Q-learning is essentially performing a random search until it encounters a trajectory that satisfies the given formula. Therefore, if the family of policies that satisfy the formula with positive probability is small, it will on average take the Q-learning algorithm solving Problem 1A a longer time to converge to a solution that enforces formula satisfaction.

VI. Conclusions and Future Work

In this paper, we proposed an approximation of STL synthesis problems that can be solved via Q-learning. The proposed approach is based on 1) remodeling the system as a τ-MDP where each state corresponds to a τ-length trajectory and τ is computed based on the given STL formula, 2) approximating the probability of satisfaction and expected robustness degree such that the new objective functions are in the form of sum of rewards. We showed that the polices computed by the proposed approach can perform arbitrarily close to the optimal policies of the original problems. Finally, we demonstrated the performance of the proposed method on a case study and observed that after the same number of training, the resulting policy by maximizing the expected robustness degree performs better than the resulting policy by maximizing the probability of satisfaction. Future research includes incorporating complexity reduction techniques for faster convergence to optimal policies and extending this work for multi-agent systems.

References

[1] A. Abate, A. D'Innocenzo, and M. Di Benedetto. Approximate abstractions of stochastic hybrid systems. *IEEE Trans. on Automatic Control*, 56(11):2688–2694, Nov 2011.

[2] D. Aksaray, A. Jones, Z. Kong, M. Schwager, and C. Belta. Q-learning for robust satisfaction of signal temporal logic specifications. *arXiv preprint*, 2016.

[3] D. Aksaray, K. Leahy, and C. Belta. Distributed multi-agent persistent surveillance under temporal logic constraints. *IFAC-PapersOnLine*, 48(22):174–179, 2015.

[4] D. Aksaray, C.-I. Vasile, and C. Belta. Dynamic routing of energy-aware vehicles with temporal logic constraints. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3141–3146. IEEE, 2016.

[5] C. Baier and J.-P. Katoen. *Principles of model checking*, volume 26202649. MIT press Cambridge, 2008.

[6] T. Brazdil, K. Chatterjee, M. Chmelik, M.k, V. Forejt, J. Kretinsky, M. Kwiatkowska, D. Parker, and M. Ujma. Verification of markov decision processes using learning algorithms. In F. Cassez and J.-F. Raskin, editors, *Automated Technology for Verification and Analysis*, volume 8837 of *Lecture Notes in Computer Science*, pages 98–114. Springer International Publishing, 2014.

[7] M. Chen and M. Chiang. Distributed optimization in networking: Recent advances in combinatorial and robust formulations. In *Modeling and Optimization: Theory and Applications*, pages 25–52. Springer, 2012.

[8] L. De Alfaro and Z. Manna. Verification in continuous time by discrete reasoning. In *Algebraic Methodology and Software Technology*, pages 292–306. Springer, 1995.

[9] X. C. Ding, S. L. Smith, C. Belta, and D. Rus. Optimal control of markov decision processes with linear temporal logic constraints. *IEEE Trans. on Automatic Control*, 59(5):1244–1257, 2014.

[10] A. Dokhanchi, B. Hoxha, and G. Fainekos. On-line monitoring for temporal logic robustness. In *Runtime Verification*, pages 231–246. Springer, 2014.

[11] A. Donzé and O. Maler. *Robust satisfaction of temporal logic over real-valued signals*. Springer, 2010.

[12] G. E. Fainekos and G. J. Pappas. Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science*, 410(42):4262–4291, 2009.

[13] C. A. Furia and M. Rossi. Integrating discrete-and continuous-time metric temporal logics through sampling. In *Formal Modeling and Analysis of Timed Systems*, pages 215–229. Springer, 2006.

[14] A. Julius and G. Pappas. Approximations of stochastic hybrid systems. *IEEE Trans. on Automatic Control*, 54(6):1193–1203, June 2009.

[15] M. Kamgarpour, J. Ding, S. Summers, A. Abate, J. Lygeros, and C. Tomlin. Discrete time stochastic hybrid dynamic games: Verification and controller synthesis. In *IEEE Conf. on Decision and Control and European Control Conference*, pages 6122–6127, 2011.

[16] Z. Kong, A. Jones, and C. Belta. Temporal logics for learning and detection of anomalous behavior. *IEEE Trans. on Automatic Control*, PP(99):1–1, 2016.

[17] M. Lahijanian, S. B. Andersson, and C. Belta. Formal verification and synthesis for discrete-time stochastic systems. *IEEE Trans. on Automatic Control*, 6(8):2031–2045, 2015.

[18] S. H. Lim and G. DeJong. Towards finite-sample convergence of direct reinforcement learning. In *Machine Learning: ECML 2005*, pages 230–241. Springer, 2005.

[19] V. Raman, A. Donze, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia. Model predictive control with signal temporal logic specifications. In *IEEE Conf. on Decision and Control (CDC)*, pages 81–87, 2014.

[20] D. Sadigh, E. S. Kim, S. Coogan, S. S. Sastry, and S. A. Seshia. A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications. In *IEEE Conf. on Decision and Control*, pages 1091–1096. IEEE, 2014.

[21] J. N. Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine Learning*, 16(3):185–202, 1994.

[22] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.