2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Control Synthesis from Linear Temporal Logic Specifications using Model-Free Reinforcement Learning

Alper Kamil Bozkurt, Yu Wang, Michael M. Zavlanos, and Miroslav Pajic

Abstract— **We present a reinforcement learning (RL) framework to synthesize a control policy from a given linear temporal logic (LTL) specification in an unknown stochastic environment that can be modeled as a Markov Decision Process (MDP). Specifically, we learn a policy that maximizes the probability of satisfying the LTL formula without learning the transition probabilities. We introduce a novel rewarding and discounting mechanism based on the LTL formula such that (i) an optimal policy maximizing the total discounted reward effectively maximizes the probabilities of satisfying LTL objectives, and (ii) a model-free RL algorithm using these rewards and discount factors is guaranteed to converge to such a policy. Finally, we illustrate the applicability of our RL-based synthesis approach on two motion planning case studies.**

I. Introduction

Formal logics have been used to facilitate robot motion planning beyond its traditional focus on computing robot trajectories that, starting from an initial region, reach a desired goal without hitting any obstacles (e.g., [1], [2]). Linear Temporal Logic (LTL) is a widely used framework for formal specification of high-level robotic tasks on discrete models. Thus, control synthesis on discrete-transition systems for LTL objectives has attracted a lot of attention (e.g., [3]–[7]).

Another line of work considers motion planning for LTL objectives for systems that exhibit uncertainty coming from either robot dynamics or the environment, such as Markov Decision Processes (MDPs) [8]–[14]. By synthesizing control for an MDP, from a given LTL objective, the obtained controller maximizes the probability of satisfying the specification. Also, tools from probabilistic model checking [15] can be directly used for synthesis. Yet, when the MDP transition probabilities are not known a priori, the control policy needs to be synthesized through learning from samples.

Hence, there is a recent focus on learning for control (i.e., motion planning) synthesis for LTL objectives [16]–[26]. Most model-based reinforcement learning (RL) methods are based on detection of end components, and provide estimates of satisfaction probabilities with probably approximately correct bounds (e.g., [16], [17]). Such approaches, however, need to first learn and store the MDP transition probabilities, and thus have significant space requirements, restricting their use to systems with small and low-dimensional state spaces.

On the other hand, model-free RL methods derive the desired policies without storing a model of the MDP. The temporal logic tasks need to be represented by a reward function, possibly with a finite-memory, so that the optimal policy maximizing the discounted future reward, also maximizes the probability of satisfying the tasks. One approach is to use temporal logic specifications that are time-bounded or defined on finite traces so that they can be directly translated to a real-valued reward function [19]–[22]. Alternatively, unbounded LTL formulas can be transformed into an ω-automaton and the accepting condition of the automaton can be used to design the reward function.

Such reward functions based on Rabin conditions are introduced in [23], as part of a model-free RL method; the approach assigns a sufficiently small negative and a positive reward to the first and second sets of the Rabin pairs, respectively. A generalization to deep Q-learning, with a new optimization algorithm, is done in [24]. Yet, in the presence of rejecting end components or multiple Rabin pairs, optimal policies obtained by this method may not satisfy the LTL property almost surely, even if such policies exist [25].

A given LTL property can also be translated into a limit-deterministic Büchi automaton (LDBA), which can be used in quantitative analysis of MDPs [27], [28]. The first reward function based on LDBA accepting conditions is introduced in [29]. Yet, similar to [23], in the presence of non-accepting components, the algorithm might fail to converge to the policy that almost surely satisfies the LTL specification.

The problem of satisfying the Büchi condition of an LDBA can be reduced to a reachability problem by adding transitions with a positive reward from accepting states to a terminal state [25]. Then, as the transition probability from an accepting state to the terminal state goes to zero, in order to reach the terminal state and obtain a positive reward, an accepting state needs to be visited infinitely often, which captures the satisfaction of the Büchi condition. However, model-free RL algorithms such as Q-learning may fail to converge to the correct reachability probabilities without discounting (or improper discounting) in the presence of end components [17], as Q-learning might get stuck in one of the fixed-point solutions; e.g., if all the values are initialized to 1, Q-learning will not be able to decrease any value estimate.

Consequently, in this paper, we propose a model-free RL algorithm that is *guaranteed to find a control policy that maximizes the probability of satisfying a given LTL objective (i.e., specification) in an arbitrary unknown MDP*; for the MDP, not even which probabilities are nonzero (i.e., its graph/topology) is known. We use an automata-based approach that constructs a product MDP using an LDBA of a given LTL formula and assigns rewards based on the Büchi (repeated reachability) acceptance condition. Such optimal

This work is sponsored in part by the ONR under agreements N00014-17-1-2504, N00014-20-1-2745 and N00014-18-1-2374, AFOSR award number FA9550-19-1-0169, and the NSF CNS-1652544 and CNS-1932011 grants.

Alper Kamil Bozkurt, Yu Wang, Michael M. Zavlanos and Miroslav Pajic are with Duke University, Durham, NC 27708, USA, {alper.bozkurt, yu.wang094, michael.zavlanos, miroslav.pajic}@duke.edu.

policy can then be derived by learning a policy maximizing the satisfaction probability of the Büchi condition on the product. Unlike [25], our approach directly assigns positive rewards to the accepting states and discounts these rewards in such a way that the values of the optimal policy are *proved to converge to the maximal satisfaction probabilities* as the discount factor goes beyond a threshold that is less than 1.

The rest of the paper is organized as follows. We introduce preliminaries and formalize the problem in Sec. II. Sec. III presents our model-free RL algorithm that maximizes probabilities that LTL specifications are satisfied. Finally, we evaluate our approach on several motion planning problems for mobile robots (Sec. IV), before concluding in Sec. V.

II. Preliminaries and Problem Statement

We start with preliminaries on LTL, MDPs, and RL on MDPs, before problem formulation. We denote the sets of real and natural numbers by $\mathbb{R}$ and $\mathbb{N}$, respectively. For a set S, S^+ denotes the set of all finite sequences taken from S.

A. Markov Decision Processes and Reinforcement Learning

MDPs are common modeling formalism for systems that permit nondeterministic choices with probabilistic outcomes.

Definition 1. *A (labeled) MDP is a tuple $\mathcal{M} = (S, A, P, s_0, AP, L)$, where S is a finite set of states, A is a finite set of actions, $P : S \times A \times S \to [0, 1]$ is the transition probability function, $s_0 \in S$ is an initial state, AP is a finite set of atomic propositions, and $L : S \to 2^{AP}$ is a labeling function. For simplicity, let $A(s)$ denote the set of actions that can be taken in state s; then for all states $s \in S$, it holds that $\sum_{s' \in S} P(s, a, s') = 1$ if $a \in A(s)$, and 0 otherwise.*

A path is an infinite sequence of states $\sigma = s_0 s_1 s_2 \ldots$, with $s_i \in S$ such that for all $i \geq 0$, there exists $a_i \in A$ with $P(s_i, a_i, s_{i+1}) > 0$. We use $\sigma[i]$ to denote the state s_i, as well as $\sigma[:i]$ and $\sigma[i{+}1:]$ to denote the prefix $s_0 s_1 \ldots s_i$ and the suffix $s_{i+1} s_{i+2} \ldots$ of the path, respectively.

Definition 2. *A **policy** π for an MDP $\mathcal{M}$ is a function $\pi : S^+ \to A$ such that $\pi(\sigma[:n]) \in A(\sigma[n])$. A policy is **memoryless** if it only depends on the current state, i.e., $\pi(\sigma[:n]) = \pi(\sigma[n])$ for any σ, and thus can be defined as $\pi : S \to A$. A Markov chain (MC) of an MDP $\mathcal{M}$ induced by a memoryless policy π is a tuple $\mathcal{M}_\pi = (S, P_\pi, s_0, AP, L)$, where $P_\pi(s, s') = P(s, \pi(s), s')$ for all $s, s' \in S$. A **bottom strongly connected component** (BSCC) of an MC is a strongly connected component with no outgoing transitions.*

Let $R : S \to \mathbb{R}$ be a *reward function* of the MDP $\mathcal{M}$. Then, for a discount factor $\gamma \in (0, 1)$, the K-step *return* ($K \in \mathbb{N}$ or $K = \infty$) of a path σ from time $t \in \mathbb{N}$ is

$$G_{t:K}(\sigma) = \sum_{i=0}^{K} \gamma^i R(\sigma[t{+}i]),\ G_t(\sigma) = \lim_{K \to \infty} G_{t:K}(\sigma). \quad (1)$$

Under a policy π, the *value* of a state s is defined as the expected return of a path – i.e.,

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t(\sigma) \mid \sigma[t] = s \right], \quad (2)$$

for any fixed $t \in \mathbb{N}$ such that $Pr_\pi^{\mathcal{M}}(\sigma[t] = s) > 0$.

The RL objective is to find an optimal policy π^* for MDP $\mathcal{M}$ from samples, such that the return $v_\pi(s)$ is maximized for all $s \in S$; we denote the maximum by $v_*(s)$. Specifically, RL is *model-free*, if π^* is derived without explicitly estimating the transition probabilities, as done in model-based RL; hence, it scales significantly better in large applications [30].

B. LTL and Limit-Deterministic Büchi Automata

LTL provides a high-level language to describe specifications of a system. LTL formulas can be constructed inductively as combinations of Boolean operators, negation ($\neg$) and conjunction ($\wedge$), and two temporal operators, next ($\bigcirc$) and until (U), using the following syntax:

$$\varphi ::= \text{true} \mid a \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \mathsf{U} \varphi_2,\ a \in \text{AP}. \quad (3)$$

The satisfaction of an LTL formula φ for a path σ of an MDP from Def. 1 (denoted by $\sigma \models \varphi$) is defined as follows: σ satisfies an atomic proposition a, if the first state s_0 of the path is labeled with a, i.e., $a \in L(s_0)$; a path σ satisfies $\bigcirc\varphi$ if $\sigma[1:]$ satisfies the formula φ; and finally,

$$\sigma \models \varphi_1 \mathsf{U} \varphi_2, \quad \text{if } \exists i. \sigma[i] \models \varphi_2 \text{ and } \forall j < i. \sigma[j] \models \varphi_1. \quad (4)$$

Other common Boolean and temporal operators are derived as follows: (or) $\varphi_1 \vee \varphi_2 \equiv \neg(\neg\varphi_1 \wedge \neg\varphi_2)$; (implies) $\varphi_1 \to \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2$; (eventually) $\Diamond\varphi \equiv \text{true}\ \mathsf{U}\ \varphi$; and (always) $\Box\varphi \equiv \neg(\Diamond\neg\varphi)$ [15].

Satisfaction of an LTL formula can be evaluated on a Limit-Deterministic Büchi Automata (LDBA) that can be directly derived from the formula [27], [28].

Definition 3. *An LDBA is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, B)$, where Q is a finite set of states, Σ is a finite alphabet, $\delta : Q \times (\Sigma \cup \{\epsilon\}) \to 2^Q$ is a (partial) transition function, $q_0 \in Q$ is an initial state, and B is a set of* accepting states*, such that (i) δ is total except for the ϵ-moves, i.e., $|\delta(q, \alpha)| = 1$ for all $q \in Q, \alpha \in \Sigma$; and (ii) there exists a bipartition of Q to an initial and an accepting component, i.e., $Q_I \cup Q_A = Q$, where*

- *the ϵ-moves are not allowed in the accepting component, i.e., for any $q \in Q_A$, $\delta(q, \epsilon) = \emptyset$,*
- *outgoing transitions from the accepting component stay within it, i.e., for any $q \in Q_A, \nu \in \Sigma$, $\delta(q, \nu) \subseteq Q_A$,*
- *the accepting states are in the accepting component, i.e., $B \subseteq Q_A$.*

*An infinite path σ is accepted by the LDBA if it satisfies the **Büchi condition** – i.e., $inf(\sigma) \cap B \neq \emptyset$, where $inf(\sigma)$ denotes the set of states visited by σ infinitely many times.*

C. Problem Statement

In this work, we consider the problem of synthesizing a robot control policy in a stochastic environment such that ***the probability of satisfying the desired specification is maximized***. The robot environment is modeled as an MDP with *unknown transition probabilities* (i.e., not even which probabilities are nonzero is known), and the desired objective (i.e., specification) is given by an LTL formula. Our goal is to obtain such a policy by *learning the maximal probabilities that the LTL specification is satisfied*; this should be achieved by directly interacting with the environment – i.e., *without constructing a model of the MDP*.

For any policy π, $Pr_\pi(s \models \varphi)$ denotes the probability of all paths from the state s to satisfy formula φ under the policy

$$Pr_\pi^{\mathcal{M}}(s \models \varphi) := Pr_\pi^{\mathcal{M}}\{\sigma \mid \sigma[0] = s \text{ and } \sigma \models \varphi\}. \quad (5)$$

We omit the superscript $\mathcal{M}$ when it is clear from the context. We now formally state the problem considered in this work.

Problem 1. *Given an MDP $\mathcal{M} = (S, A, P, s_0, AP, L)$ where P is fully unknown and an LTL specification φ, design a model-free RL algorithm that finds a finite-memory objective policy π^φ that satisfies*

$$Pr_{\pi^\varphi}(s \models \varphi) = Pr_{max}(s \models \varphi), \quad (6)$$

where $Pr_{max}(s \models \varphi) := \max_\pi Pr_\pi(s \models \varphi)$ for all $s \in S$.

III. RL-Based Synthesis from LTL Specifications

In this section, we introduce a framework to solve Problem 1. We start by exploiting the fact that any LTL formula can be transformed into an LDBA that can be used in quantitative analysis of MDPs [27], [28]; in such LDBAs, the only nondeterministic transitions are the ϵ-moves from the initial component to the accepting component (e.g., see Fig. 1a). Therefore, we reduce the problem of satisfying a given LTL objective φ in an MDP $\mathcal{M}$ to the problem satisfying a repeated reachability (Büchi) objective $\varphi_B = \Box\Diamond B$ in the product MDP, computed from the MDP $\mathcal{M}$ and the obtained LBDA. We then exploit a new discounting and rewarding mechanism that enables the use of model-free reinforcement learning, to find an objective policy with strong performance guarantees (i.e., probability maximization). Specifically, we use Q-learning [31] in this work, but other reinforcement learning methods can be applied similarly. Our overall approach is captured in Algorithm 1, and we now describe each step in detail.

A. Design of Product MDP

Given an LTL formula φ with atomic propositions 2^{AP}, the product MDP is constructed by composing $\mathcal{M}$ with an LDBA $\mathcal{A}_\varphi$ with the alphabet 2^{AP}, which can be automatically derived from φ [27], [28]. LDBAs, similarly to deterministic Rabin automata [15], can be used in quantitative analysis of MDPs if they are constructed in a certain way [25].

Definition 4. *A product MDP $\mathcal{M}^\times = (S^\times, A^\times, P^\times, s_0^\times, B^\times)$ of an MDP $\mathcal{M} = (S, A, P, s_0, AP, L)$ and an LDBA $\mathcal{A} = (Q, 2^{AP}, \delta, q_0, B)$ is defined as: $S^\times = S \times Q$ is the set of states, $A^\times = A \cup A^\epsilon$, $A^\epsilon{:=}\{\epsilon_q | q \in Q\}$ is the action set, $P^\times : S^\times \times A^\times \times S^\times \to [0, 1]$ is the transition function*

$$P^\times(\langle s, q\rangle, a, \langle s', q'\rangle) = \begin{cases} P(s, a, s') & q' = \delta(q, L(s)) \text{ and } a \notin A^\epsilon \\ 1 & a = \epsilon_{q'} \text{ and } q' \in \delta(q, \epsilon) \text{ and } s = s' \\ 0 & \text{otherwise} \end{cases}, \quad (7)$$

$s_0^\times$ is $\langle s_0, q_0\rangle$, and $B^\times = \{\langle s, q\rangle \in S^\times \mid q \in B\}$ is the set of accepting states. We say that a path σ of the product MDP $\mathcal{M}^\times$ satisfies the Büchi condition φ_B if $inf(\sigma) \cap B^\times \neq \emptyset$.

The nondeterministic ϵ-moves in the LDBA are represented by ϵ-actions in the product MDP. When an ϵ-action is taken, only the state of the LDBA is updated according to the corresponding ϵ-move. When an MDP action is taken, the next MDP state will be determined by the transition probabilities and the LDBA makes a transition by consuming the label of the current MDP state. Intuitively, an ϵ-action can be considered as guessing the possible paths generated in the future. If, as part of iterative learning, the guess is wrong the agent cannot change its guess; however, in the next RL episode, the agent can make the correct one.

Algorithm 1: Model-free RL-based synthesis on MDPs that maximizes the satisfaction probability of LTL specifications.

Input: LTL formula φ, MDP $\mathcal{M}$
Translate φ to an LDBA $\mathcal{A}_\varphi$
Construct the product $\mathcal{M}^\times$ of $\mathcal{M}$ and $\mathcal{A}_\varphi$
Initialize $Q(\langle s, q\rangle, a)$ on $\mathcal{M}^\times$
for $t = 0, 1, \ldots, T$ **do**
  Derive a policy π from Q
  Take the action $a_t \leftarrow \pi(\langle s, q\rangle_t)$
  Observe the next state $\langle s, q\rangle_{t+1}$
  $Q(\langle s, q\rangle_t, a_t) \leftarrow (1-\alpha)\cdot Q(\langle s, q\rangle_t, a_t) + \alpha \cdot R_B(\langle s, q\rangle_t) + \alpha \cdot \Gamma_B(\langle s, q\rangle_t) \cdot \max_{a'} Q(\langle s, q\rangle_{t+1}, a')$
end for
Get a greedy policy π_B^φ from Q
return π_B^φ and $\mathcal{A}_\varphi$

An example product MDP is illustrated in Fig. 1. In the MDP (Fig. 1b), states s_0 and s_1 are labeled by atomic propositions a and b, respectively. In the LDBA (Fig. 1a), for simplicity, the transitions are labeled by Boolean formulas of the atomic propositions of a and b or an ϵ label, with 1 standing for "true"; this is equivalent to labeling the transitions using sets of atomic propositions, as in Def. 3. A transition labeled by a Boolean formula is triggered upon receiving a set of atomic propositions satisfying that formula, and a transition labeled by an ϵ label can be (but does not have to be) triggered automatically. The product MDP is shown in Fig. 1c.

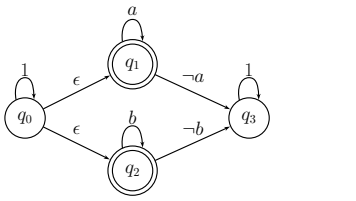
To distinguish the two ϵ transitions from q_0 to q_1 and from q_0 to q_2 in Fig. 1a, we denote them by ϵ_1 and ϵ_2 in Fig. 1c, respectively. Notice that choosing ϵ_2 before choosing β does not satisfy the Büchi condition although the generated paths by this policy satisfy the LTL formula. Yet, this does not constitute a problem because in such cases, there always exists a corresponding policy that generates the same paths and satisfies the Büchi condition (e.g. choosing ϵ_2 after β).

Now, the satisfaction of the LTL objective φ on the original MDP $\mathcal{M}$ is related to the satisfaction of the Büchi objective φ_B on the product MDP $\mathcal{M}^\times$, as formalized below.

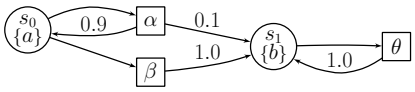
Lemma 1. *A memoryless policy π_B^φ that maximizes the satisfaction probability of φ_B on $\mathcal{M}^\times$ induces a finite-memory policy π^φ that maximizes the satisfaction probability of φ on $\mathcal{M}$ in Problem 1.*

Proof. Follows from the proof of Theorem 3 in [28]. □

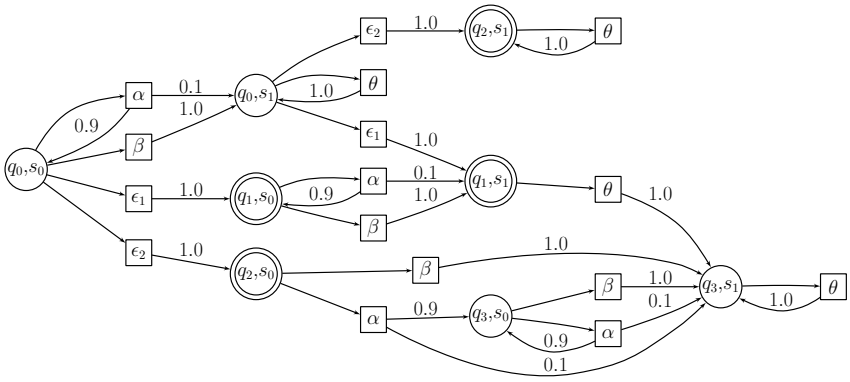
Therefore, the behavior of the induced policy π^φ can be described by the policy π_B^φ and the LDBA $\mathcal{A}_\varphi$ derived directly from the LTL formula φ. Initially, $\mathcal{A}_\varphi$ is reset to its start state q_0 and whenever the MDP $\mathcal{M}$ makes a transition



(a) A derived LDBA $\mathcal{A}$ for the LTL formula $\varphi = \Diamond\Box a \vee \Diamond\Box b$



(b) An example MDP $\mathcal{M}$; the circles denote MDP states, rectangles denote actions, and numbers transition probabilities



(c) The obtained product MDP

Fig. 1: Product MDP $\mathcal{M}^\times$ obtained from an MDP $\mathcal{M}$ and an LDBA $\mathcal{A}$ that is automatically derived from an LTL formula φ.

from s to s', $\mathcal{A}_\varphi$ updates its current state from q to $\delta(q, L(s))$. The action to be selected in an MDP state s when $\mathcal{A}_\varphi$ is in a state q is determined by π_B^φ as follows: if $\pi_B^\varphi(\langle s,q\rangle)$ is an ϵ-action $\epsilon_{q'}$, $\mathcal{A}_\varphi$ changes its state to q' and the action $\pi_B^\varphi(\langle s,q'\rangle)$ is selected; otherwise, $\pi_B^\varphi(\langle s,q\rangle)$ is selected.

B. Learning for Büchi Conditions with Discounted Rewards

Our goal is to learn a policy that maximizes the probability of satisfying a given Büchi objective. By Lemma 1, in what follows, we assume policies are memoryless since they are sufficient for Büchi objectives. For simplicity, we omit the superscript $^\times$ and write $\mathcal{M} = (S, A, P, s_0, B)$ and $s \in S$ instead of $\mathcal{M}^\times = (S^\times, A^\times, P^\times, s_0^\times, B^\times)$ and $\langle s,q\rangle \in S^\times$.

We propose a model-free learning method that uses carefully crafted rewards and state-dependent discounting based on the Büchi condition such that an optimal policy π^* maximizing the expected return is also an objective policy π_B^φ maximizing the satisfaction probabilities. Specifically, we define the return of a path as a function of these rewards and discount factors in such a way that the value of a state, the expected return from that state, approaches the probability of satisfying the objective as the discount factor γ goes to 1.

Theorem 1. *For a given MDP $\mathcal{M}$ with $B \subseteq S$, the value function v_π^γ for the policy π and the discount factor γ satisfies*

$$\lim_{\gamma\to 1^-} v_\pi^\gamma(s) = Pr_\pi(s \models \Box\Diamond B) \tag{8}$$

for all states $s \in S$, if the return of a path is defined as

$$G_t(\sigma) := \sum\nolimits_{i=0}^{\infty} R_B(\sigma[t{+}i]) \cdot \prod\nolimits_{j=0}^{i-1} \Gamma_B(\sigma[t{+}j]) \tag{9}$$

where $\prod_{j=0}^{-1} := 1$, $R_B : S \to [0,1)$ and $\Gamma_B : S \to (0,1)$ are the reward and the discount functions defined as:

$$R_B(s) := \begin{cases} 1-\gamma_B & s \in B \\ 0 & s \notin B \end{cases},\ \Gamma_B(s) := \begin{cases} \gamma_B & s \in B \\ \gamma & s \notin B \end{cases} \tag{10}$$

Here, we set $\gamma_B = \gamma_B(\gamma)$ as a function of γ such that

$$\lim_{\gamma\to 1^-} \frac{1-\gamma}{1-\gamma_B(\gamma)} = 0. \tag{11}$$

Before proving Theorem 1, we develop bounds on $G_t(\sigma)$.

Lemma 2. *For all paths and $G_t(\sigma)$ from* (9)*, it holds that*

$$0 \le \gamma G_{t+1}(\sigma) \le G_t(\sigma) \le 1 - \gamma_B + \gamma_B G_{t+1}(\sigma) \le 1 \tag{12}$$

Proof. Since there is no negative reward, $G_t \ge 0$ holds. By the return definition, replacing γ with 1 yields a larger or equal return, which constitutes the following upper bound on the return: $G_t(\sigma) \le 1 - \gamma_B^b \le 1$, where b is the number of B states visited. Return $G_t(\sigma)$ from (9) satisfies

$$G_t(\sigma) = \begin{cases} 1 + \gamma_B(G_{t+1}(\sigma) - 1) & \sigma[t] \in B \\ \gamma G_{t+1}(\sigma) & \sigma[t] \notin B \end{cases} \tag{13}$$

From $G_t(\sigma) \le 1$ it follows that $1 + \gamma_B(G_{t+1}(\sigma) - 1) \ge \gamma G_{t+1}(\sigma)$, which with (13) proves the other inequalities. □

Lemma 2 implies that replacing a prefix of a path with states belonging to B never decreases the return of a path and similarly replacing with states that do not belong to B never increases the return. The result is particularly useful when we establish upper and lower bounds on the value of a state.

The next lemma shows that under a policy, the values of states in the accepting BSCCs of the induced Markov chain approach 1 in the limit; thus, is the key to proving Theorem 1.

Lemma 3. *Let BSCC($\mathcal{M}_\pi$) denote the set of all BSCCs of an induced Markov chain $\mathcal{M}_\pi$ and let B_π denote the set of B states that belong to a BSCC of $\mathcal{M}_\pi$ – i.e.,*

$$B_\pi := \{s \mid s \in B, s \in T, T \in \mathit{BSCC}(\mathcal{M}_\pi)\}. \tag{14}$$

Then, for any state s in B_π

$$\lim_{\gamma\to 1^-} v_\pi^\gamma(s) = 1. \tag{15}$$

Proof. For any fixed $t \in \mathbb{N}$, let N_t be the stopping time of first returning to the state $s \in S$ after leaving it at t,

$$N_t = \min\{\tau \mid \sigma[t{+}\tau] = s, \tau > 0\}. \tag{16}$$

Then by (2), it holds that

$$\begin{aligned} v_\pi^\gamma(s) &= 1 - \gamma_B + \gamma_B \mathbb{E}_\pi[G_{t+1}(\sigma) \mid \sigma[t]{=}s] \\ &= 1 - \gamma_B + \gamma_B \mathbb{E}_\pi\Big[G_{t+1:t+N_t-1}(\sigma) \\ &\quad + \Big(\prod\nolimits_{i=1}^{N_t-1} \Gamma(\sigma[t{+}i])\Big) \cdot G_{t+N_t}(\sigma) \mid \sigma[t]{=}s\Big], \end{aligned} \tag{17}$$

since once a state $s \in B_\pi$ is visited, almost surely it is visited again [15]. Using that $G_t(\sigma) \geq \gamma G_{t+1}(\sigma)$, we obtain

$$\begin{aligned} v_\pi^\gamma(s) &\geq 1-\gamma_B+\gamma_B\mathbb{E}_\pi\left[\gamma^{N_t-1}G_{t+N_t}(\sigma) \mid \sigma[t]{=}s\right] \\ &\overset{①}{\geq} 1-\gamma_B+\gamma_B\mathbb{E}_\pi\left[\gamma^{N_t-1} \mid \sigma[t]{=}s\right] v_\pi(s) \\ &\overset{②}{\geq} 1-\gamma_B+\gamma_B\gamma^{\mathbb{E}_\pi[N_t-1|\sigma[t]=s]}v_\pi(s) \\ &\geq 1-\gamma_B+\gamma_B\gamma^n v_\pi(s) \end{aligned} \tag{18}$$

where ① holds by the Markov property, ② holds by the Jensen's inequality, and $n \geq 1$ is a constant. From (18),

$$\begin{aligned} v_\pi^\gamma(s) &\geq \frac{1-\gamma_B}{1-\gamma_B\gamma^n} \geq \frac{1-\gamma_B}{1-\gamma_B(1-n(1-\gamma))} \\ &= \frac{1}{1+n\frac{1-\gamma}{1-\gamma_B}-n(1-\gamma)}. \end{aligned} \tag{19}$$

where the second "$\geq$" holds by $(1-(1-\gamma))^n \geq 1-n(1-\gamma)$ for $\gamma \in (0,1)$. Finally, since $v_\pi^\gamma(s) \leq 1$ by Lemma 2, letting $\gamma, \gamma_B \to 1^-$ under the condition (11) results in (15). □

We now prove Theorem 1.

Proof of Theorem 1. First, we divide the expected return of a random path σ from a state $s \in S$ by whether it visits the states $B \subseteq S$ infinitely often:

$$\begin{aligned} v_\pi^\gamma(s) = \mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t]{=}s, \sigma \models \square\lozenge B]Pr_\pi(s \models \square\lozenge B) \\ + \mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t]{=}s, \sigma \not\models \square\lozenge B]Pr_\pi(s \not\models \square\lozenge B) \end{aligned} \tag{20}$$

for some fixed $t \in \mathbb{N}$. let M_t be the stopping time of first reaching a state in B_π after leaving s at t,

$$M_t = \min\{\tau \mid \sigma[t{+}\tau] \in B_\pi, \tau > 0\} \tag{21}$$

where B_π is defined as in (14). Then, it holds that

$$\begin{aligned} &\mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t]{=}s, \sigma \models \square\lozenge B] \\ &\overset{①}{=} \mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t]{=}s, \sigma \models \lozenge B_\pi] \\ &\overset{②}{\geq} \mathbb{E}_\pi\left[\gamma^{M_t}G_{t+M_t}(\sigma) \mid \sigma[t]{=}s, \sigma \models \lozenge B_\pi\right] \\ &\overset{③}{\geq} \mathbb{E}_\pi\left[\gamma^{M_t} \mid \sigma[t]{=}s, \sigma \models \lozenge B_\pi\right] v_{\pi,\min}^\gamma(B_\pi) \\ &\overset{④}{\geq} \gamma^{\mathbb{E}_\pi[M_t|\sigma[t]=s,\sigma\models\lozenge B_\pi]}v_{\pi,\min}^\gamma(B_\pi) \\ &= \gamma^m v_{\pi,\min}^\gamma(B_\pi), \end{aligned} \tag{22, 23}$$

where $v_{\pi,\min}^\gamma(B_\pi) = \min_{s\in B_\pi} v_\pi^\gamma(s)$ and m is constant. Here, ① holds because a path $\sigma \models \square\lozenge B$ almost surely eventually enters an accepting BSCC, it eventually reaches a state $s \in B_\pi$ almost surely, ②, ③ and ④ hold due to Lemma 2, the Markov property and Jensen's inequality. From (20), we have

$$v_\pi^\gamma(s) \geq \gamma^m v_\pi(B_\pi) Pr_\pi(s \models \square\lozenge B). \tag{24}$$

Similarly, let M'_t be the stopping time of first reaching a rejecting BSCC of $\mathcal{M}_\pi$ after leaving s at t. Then

$$\begin{aligned} M'_\pi = \min\big\{\tau \mid \sigma[t{+}\tau] \in T, T \cap B = \varnothing, \\ T \in BSCC(\mathcal{M}_\pi), \tau > 0\big\} \end{aligned} \tag{25}$$

denoting the number of time steps before a rejecting BSCC is reached. Thus, from Lemma 2 and the Markov property

$$\begin{aligned} &\mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t]{=}s, \sigma \not\models \square\lozenge B] \\ &\leq \mathbb{E}_\pi\left[1-\gamma_B^{M'_\pi} \mid \sigma[t]{=}s, \sigma \not\models \square\lozenge B\right] \\ &\leq 1-\gamma_B^{\mathbb{E}_\pi\left[M'_\pi|\sigma[t]=s,\sigma\not\models\square\lozenge B\right]} = 1-\gamma_B^{m'} \end{aligned} \tag{26}$$

where m' is also constant. From this upper bound and (20)

$$v_\pi^\gamma(s) \leq Pr_\pi(s \models \square\lozenge B) + (1-\gamma_B^{m'})Pr_\pi(s \not\models \square\lozenge B).$$

Both the above upper bound and the lower bound from (24) go to the probability of satisfying the formula as γ approaches 1 from below, thus concluding the proof. □

Theorem 1 suggests that the limit of the optimal state values is equal to the maximal probabilities as γ goes to 1; this is captured by the next corollary whose proof follows from the definition of the optimal policies and maximal probabilities.

Corollary 1. *For all states $s \in S$ the following holds:*

$$\lim_{\gamma\to 1^-} v_*^\gamma(s) = Pr_{max}(s \models \square\lozenge B). \tag{27}$$

Remark 1. *From Theorem 1 of [32], $\gamma < 1$ ensures convergence of the model-free learning to the unique solution. With $\gamma = 1$, the result may converge to a non-optimal policy [17] as there might exist multiple fixed-point solutions.*

Finally, as the policies are discrete, the convergence of (8) and (27) is achieved after some threshold γ', as stated below.

Corollary 2. *There exists a γ' such that for all $\gamma > \gamma'$ and for all states $s \in S$, the optimal policy π^* satisfies*

$$Pr_{\pi^*}(s \models \square\lozenge B) = Pr_{max}(s \models \square\lozenge B). \tag{28}$$

Proof. Let $d_{\min}$ be the minimum positive difference between the satisfaction probabilities of two policies:

$$\begin{aligned} d_{\min} := \min\big\{&|Pr_{\pi_1}(s \models \square\lozenge B) - Pr_{\pi_2}(s \models \square\lozenge B)| \\ &\mid s \in S, Pr_{\pi_1}(s \models \square\lozenge B) \neq Pr_{\pi_2}(s \models \square\lozenge B)\big\} \end{aligned}$$

and let γ' be the discount factor such that

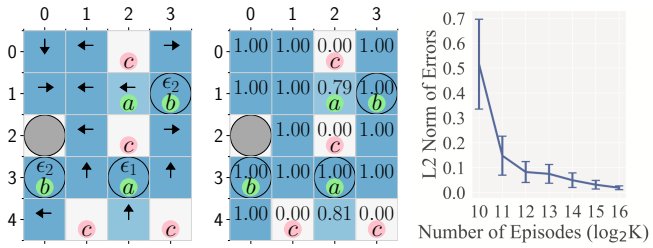
$$\max\big\{|v_\pi^\gamma(s) - Pr_\pi(s \models \square\lozenge B)| \mid s \in S\big\} < d_{\min}/2. \tag{29}$$

Now, suppose a policy π' that maximizes the satisfaction probability is not optimal for γ', then the optimal value of all states must be larger than $Pr_{\max}(s \models \square\lozenge B) - d_{\min}/2$, which is not possible due to the definition of $d_{\min}$. □

IV. Implementation and Case Studies

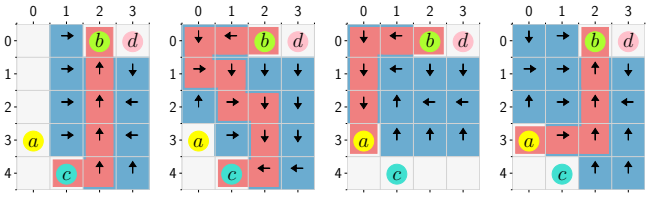
We implemented our RL-based synthesis framework in Python; we used Rabinizer 4 [33] to map LTL formulas into LDBAs, and Q-learning for the proposed discounting rewards. The code and videos are available at [34]. We evaluated our framework on two motion planning case studies. We consider two scenarios in a grid-world where a mobile robot can take four actions *top, left, down* and *right* (Fig. 2 and 3). The robot moves in the intended direction with probability 0.8 and it can go sideways with probability 0.2 (0.1 each). If the robot hits a wall or an obstacle it stays in the same state.

For Q-learning, we used ε-greedy policy to choose the optimal actions, and discount factors $\gamma_B = 0.99$ and $\gamma =$



(a) Policy (b) Value Estimates (c) Convergence

Fig. 2: The objective policy and the estimated maximal probabilities of satisfying φ_1 from (30). Empty circles: absorbing states; Filled circles: obstacles; Arrows: actions *top, left, down* and *right* and ϵ_1, ϵ_2 are ϵ-actions. State labels: encircled letters in the lower part of the cells. The values are rounded to the closest hundredth.



(a) Policy c to b (b) Policy b to c (c) Policy b to a (d) Policy a to b

Fig. 3: A summary of the synthesized policy for the nursery scenario. Arrows: actions *top, left, down*, and *right*; encircled characters: state labels. The actions in states that are not reachable or lead to another LDBA state are not displayed. In all subfigures, the most likely paths are highlighted in red.

0.99999. The probability that a random action is taken, ε, and the learning rate, α, were gradually decreased from 1.0 to 0.1 and then 0.001. The objective policies and estimates of the maximal probabilities were obtained using 100 000 episodes.

A. Motion Planning with Safe Absorbing States

In this example, the robot tries to reach a safe absorbing state (states a or b in circle), while avoiding unsafe states (states c). This is formally specified in LTL as

$$\varphi_1 = (\Diamond\Box a \vee \Diamond\Box b) \wedge \Box\neg c. \tag{30}$$

The LDBA computed from φ_1 has 4 states and the product MDP has 80 states. All episodes started in a random state and were terminated after $T = 100$ steps.

The optimal policy obtained for an MDP is illustrated in Fig. 2a. The shortest way to enter a safe absorbing state from $(0, 0)$ is reaching $(1, 3)$ via $(1, 2)$; yet, in that case, the robot visits an unsafe state with probability 0.2. Thus, the optimal policy tries to enter one of $(3, 0)$ and $(3, 2)$ by choosing *up* in $(3, 1)$. Under this policy, the robot eventually reaches a safe absorbing state without visiting an unsafe state almost surely. Once the robot enters an absorbing state, it chooses an ϵ-action depending on the state label, and thus the LDBA transitions to an accepting state, with positive rewards.

Fig. 2b shows the estimates of the maximal probabilities. Note that the approximation errors in $(1, 2)$ and $(4, 2)$ are due to the variance of the return caused by the unsafe states. When the robot visits an unsafe state, the LDBA makes a transition to a trap state, making it impossible for the robot to receive a positive reward. Hence, the return that can be obtained from $(1, 2)$ and $(4, 2)$ is either 1 or 0 with probability 0.8 and 0.2, respectively. In addition, this type of non-0 or non-1 probability guarantees cannot be provided with existing learning-based methods for LTL specifications.

While the values from Fig. 2a and 2b were obtained from a single run over K=100 000 episodes, we investigated the impact of the number of episodes. Fig. 2c shows the L2 norm of the errors averaged over 100 repetitions for different numbers of episodes (the error bars show standard deviation).

B. Mobile Robot in Nursery Scenario

In this scenario (inspired by [35]), the robot's objective is to repeatedly check a baby (at state b) and go back to its charger (at state c), while avoiding the danger zone (at state d). Near the baby b, the only allowed action is *left* and when taken the following situations can happen: (i) the robot hits the wall with probability 0.1 and wakes the baby up; (ii) the robot moves left with probability 0.8 or moves down with probability 0.1. If the baby has been woken up, which means the robot could not leave in a single time step (represented by LTL as $b \wedge \bigcirc b$), the robot should notify the adult (at state a); otherwise, the robot should directly go back to the charger (at state c). The full objective is specified in LTL as

$$\varphi_2 = \Box\Big(\underbrace{\neg d}_{(1)} \wedge \underbrace{(b \wedge \neg \bigcirc b) \rightarrow \bigcirc(\neg b \,\mathsf{U}\, (a \vee c))}_{(2)} \wedge \underbrace{a \rightarrow \bigcirc(\neg a \,\mathsf{U}\, b)}_{(3)}$$
$$\wedge \underbrace{(\neg b \wedge \bigcirc b \wedge \neg \bigcirc \bigcirc b) \rightarrow (\neg a \,\mathsf{U}\, c)}_{(4)} \wedge \underbrace{c \rightarrow (\neg a \,\mathsf{U}\, b)}_{(5)} \wedge \underbrace{(b \wedge \bigcirc b) \rightarrow \Diamond a}_{(6)}\Big).$$

Here, the sub-formulas mean (1) avoid the danger state; (2) if the baby is left, do not return before visiting the adult or the charger; (3) after notifying the adult, leave immediately and go for the baby; (4) after leaving the baby sleeping, go for the charger and do not notify the adult; (5) after charging, return to the baby first without visiting the adult; and (6) notify the adult if the baby has woken up.

The LDBA for this specification has 47 states and the product MDP has 940 states. The episodes were terminated after 1000 steps and the robot position was reset to charging.

Fig. 3 depicts the optimal policy for the four most visited LDBA states during the simulation. The robot follows the policy in Fig. 3a after it leaves the charger dock $(4, 1)$. Under this policy, the robot almost surely reaches the baby in $(0, 2)$, while successfully avoiding visiting a. Similarly, the policy in Fig. 3b is followed by the robot to go back to the charger while the baby is sleeping. If the baby is awake, the robot takes the shortest path to reach a (Fig. 3c).

V. Conclusion

In this work, we present a model-free learning-based method to synthesize a control policy that *maximizes* probability that an LTL specification is satisfied in *unknown* stochastic environments that can be modeled by an MDP. We first show that synthesizing controllers from an LTL specification on the MDP can be converted to synthesizing a memoryless policy of a Büchi objective on the product MDP. Then, we design a novel discounting and reward scheme, and show that the memoryless policy optimizing this reward, also optimizes the satisfaction probability of the Büchi objective (and thus the initial LTL specification). Finally, we evaluate our synthesis method on motion planning case studies.

REFERENCES

[1] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[2] Sertac Karaman, Matthew R Walter, Alejandro Perez, Emilio Frazzoli, and Seth Teller. Anytime motion planning using the RRT. In *2011 IEEE International Conference on Robotics and Automation*, pages 1478–1483. IEEE, 2011.

[3] C. I. Vasile and C. Belta. Sampling-based temporal logic path planning. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4817–4822, Nov 2013.

[4] Stephen L Smith, Jana Tmov, Calin Belta, and Daniela Rus. Optimal path planning for surveillance with temporal-logic constraints. *The International Journal of Robotics Research*, 30(14):1695–1708, 2011.

[5] Y. Chen, X. C. Ding, A. Stefanescu, and C. Belta. Formal approach to the deployment of distributed robotic teams. *IEEE Transactions on Robotics*, 28(1):158–171, Feb 2012.

[6] Y. Kantaros and M. M. Zavlanos. Sampling-based control synthesis for multi-robot systems under global temporal specifications. In *2017 ACM/IEEE 8th International Conference on Cyber-Physical Systems (ICCPS)*, pages 3–14, April 2017.

[7] E. M. Wolff, U. Topcu, and R. M. Murray. Optimization-based trajectory generation with linear temporal logic specifications. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5319–5325, May 2014.

[8] M. Guo and M. M. Zavlanos. Probabilistic motion planning under temporal tasks and soft constraints. *IEEE Transactions on Automatic Control*, 63(12):4051–4066, Dec 2018.

[9] Meng Guo and Dimos V Dimarogonas. Multi-agent plan reconfiguration under local LTL specifications. *The International Journal of Robotics Research*, 34(2):218–235, 2015.

[10] Y. Kantaros and M. M. Zavlanos. Sampling-based optimal control synthesis for multirobot systems under global temporal tasks. *IEEE Transactions on Automatic Control*, 64(5):1916–1931, May 2019.

[11] M. Lahijanian, S. B. Andersson, and C. Belta. Temporal logic motion planning and control with probabilistic satisfaction guarantees. *IEEE Transactions on Robotics*, 28(2):396–409, April 2012.

[12] E. M. Wolff, U. Topcu, and R. M. Murray. Robust control of uncertain Markov decision processes with temporal logic specifications. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 3372–3379, Dec 2012.

[13] Marta Kwiatkowska and David Parker. Automated verification and strategy synthesis for probabilistic systems. In Dang Van Hung and Mizuhito Ogawa, editors, *Automated Technology for Verification and Analysis*, pages 5–22, Cham, 2013. Springer International Publishing.

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

[15] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. MIT Press, Cambridge, MA, USA, 2008.

[16] Jie Fu and Ufuk Topcu. Probably approximately correct MDP learning and control with temporal logic constraints, 2014. arXiv:1404.7073 [cs.SY].

[17] Tomáš Brázdil, Krishnendu Chatterjee, Martin Chmelík, Vojtěch Forejt, Jan Křetínský, Marta Kwiatkowska, David Parker, and Mateusz Ujma. Verification of Markov decision processes using learning algorithms. In Franck Cassez and Jean-François Raskin, editors, *Automated Technology for Verification and Analysis*, pages 98–114, Cham, 2014. Springer International Publishing.

[18] Min Wen, Rüdiger Ehlers, and Ufuk Topcu. Correct-by-synthesis reinforcement learning with temporal logic constraints. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4983–4990. IEEE, 2015.

[19] D. Aksaray, A. Jones, Z. Kong, M. Schwager, and C. Belta. Q-learning for robust satisfaction of signal temporal logic specifications. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6565–6570, Dec 2016.

[20] X. Li, C. Vasile, and C. Belta. Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3834–3839, Sep. 2017.

[21] Rodrigo Toro Icarte, Toryn Q Klassen, Richard Valenzano, and Sheila A McIlraith. Teaching multiple tasks to an RL agent using LTL. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 452–461. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[22] Giuseppe De Giacomo, Luca Iocchi, Marco Favorito, and Fabio Patrizi. Foundations for restraining bolts: Reinforcement learning with LTL_f/LDL_f restraining specifications. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pages 128–136, 2019.

[23] 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 *53rd IEEE Conference on Decision and Control*, pages 1091–1096, Dec 2014.

[24] Qitong Gao, Davood Hajinezhad, Yan Zhang, Yiannis Kantaros, and Michael M. Zavlanos. Reduced variance deep reinforcement learning with temporal logic specifications. In *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, ICCPS '19, pages 237–248, New York, NY, USA, 2019. ACM.

[25] Ernst Moritz Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. Omega-regular objectives in model-free reinforcement learning. In Tomáš Vojnar and Lijun Zhang, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 395–412, Cham, 2019. Springer International Publishing.

[26] Xiao Li, Yao Ma, and Calin Belta. A policy search method for temporal logic specified reinforcement learning tasks. In *2018 Annual American Control Conference (ACC)*, pages 240–245. IEEE, 2018.

[27] Ernst Moritz Hahn, Guangyuan Li, Sven Schewe, Andrea Turrini, and Lijun Zhang. Lazy Probabilistic Model Checking without Determinisation. In Luca Aceto and David de Frutos Escrig, editors, *26th International Conference on Concurrency Theory (CONCUR 2015)*, volume 42 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 354–367, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[28] Salomon Sickert, Javier Esparza, Stefan Jaax, and Jan Křetínský. Limit-deterministic Büchi automata for linear temporal logic. In Swarat Chaudhuri and Azadeh Farzan, editors, *Computer Aided Verification*, pages 312–332, Cham, 2016. Springer International Publishing.

[29] Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Logically-constrained reinforcement learning. *arXiv:1801.08099 [cs.LG]*, 2018.

[30] Alexander L. Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L. Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML 06, page 881888, New York, NY, USA, 2006. Association for Computing Machinery.

[31] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 2nd edition, 2018.

[32] Tommi Jaakkola, Michael I. Jordan, and Satinder P. Singh. Convergence of stochastic iterative dynamic programming algorithms. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 703–710. Morgan-Kaufmann, 1994.

[33] Jan Křetínský, Tobias Meggendorfer, Salomon Sickert, and Christopher Ziegler. Rabinizer 4: From LTL to your favourite deterministic automaton. In Hana Chockler and Georg Weissenbacher, editors, *Computer Aided Verification*, pages 567–577, Cham, 2018. Springer International Publishing.

[34] A. K. Bozkurt, Y. Wang, M. M. Zavlanos, and M. Pajic. CSRL, 2019. `https://gitlab.oit.duke.edu/cpsl/csrl`.

[35] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas. Where's Waldo? sensor-based temporal logic motion planning. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3116–3121, April 2007.