Model-Free Reinforcement Learning for Stochastic Games with Linear Temporal Logic Objectives

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

Abstract— **We study synthesis of control strategies from linear temporal logic (LTL) objectives in unknown environments. We model this problem as a turn-based zero-sum stochastic game between the controller and the environment, where the transition probabilities and the model topology are fully unknown. The winning condition for the controller in this game is the satisfaction of the given LTL specification, which can be captured by the acceptance condition of a deterministic Rabin automaton (DRA) directly derived from the LTL specification. We introduce a model-free reinforcement learning (RL) methodology to find a strategy that maximizes the probability of satisfying a given LTL specification when the Rabin condition of the derived DRA has a single accepting pair. We then generalize this approach to any LTL formulas, for which the Rabin accepting condition may have more than one pairs, providing a lower bound on the satisfaction probability. Finally, we show applicability of our RL method on two planning case studies.**

I. Introduction

Linear temporal logic (LTL) offers a formal language to capture high level requirements of robot planning and control tasks, such as jobs and motion sequencing, obstacle avoidance, and surveillance. Hence, there is a growing interest in using LTL in robotics [1]–[16]. To synthesize a controller strategy from a given LTL task, most of these methods require a model of the environment a-priori, limiting their use in scenarios where the environment is unknown. For such environments, reinforcement learning (RL) is commonly utilized to search for a strategy that performs the task [17].

Recently, control synthesis using RL from LTL specifications is considered for Markov decision processes (MDPs). Model-based techniques (e.g., [18], [19]) are usually based on detection of the MDP end components; yet, with these methods, it is necessary to learn and store the transition probabilities of the MDP, which may result in very large memory requirements. Model-free RL mitigates this problem; however, the given task needs to be represented by a reward function such that a strategy maximizing the discounted cumulative reward satisfies the given LTL task specifications.

Accordingly, the problem of reward shaping from LTL specifications in MDP has been recently considered [20]–[22]. These model-free methods generally translate the given LTL specification into a limit-deterministic Büchi automaton (LDBA), which is then composed with the initial MDP, and design a reward function based on the acceptance condition of the automaton. State-space augmentation using the LDBA solves the memory requirements of the task, while the Büchi acceptance condition (i.e., repeated reachability) enables the use of simple reward functions.

However, such LDBA-based rewarding approaches are not well-suited for stochastic games, since LDBAs and many other nondeterministic automata, in general, cannot be used in solving games [23]. Hence, there are only few studies on learning-based synthesis from temporal objectives for stochastic games. One approach, [24] proposed a model-based probably approximately correct (PAC) learning algorithm for stochastic games with LTL and discounted sums of rewards objectives. Yet, the method requires that (i) the transition graph (i.e., topology) is known a-priori, (ii) the LTL objective must belong to a limited subset of LTL formulas that can be translated into a deterministic Büchi automaton, and (iii) there exists a strategy that almost surely satisfies the LTL objective. This allows pre-computation of the winning regions before learning. Also, [25] introduced a model-based learning method with PAC guarantees for reachability objectives only (i.e., very limited fragment of LTL formulas). The method uses on-the-fly detection of (simple) end components of the games, and careful construction of the confidence intervals on the transition probabilities. However, as model-based methods, both methods are inefficient in terms of space requirements when the number of possible successors of actions is not small.

Consequently, in this work we introduce a *model-free* RL approach to synthesize controllers for stochastic games, such that the obtained control policies maximize the (worst-case) probabilities of satisfying the given LTL task objectives. We start by translating the LTL objective into a Deterministic Rabin Automaton (DRA) and introduce a reward and discount function based on the Rabin acceptance condition. We first consider DRAs with a single accepting pair and prove that any model-free RL algorithm using these functions converges to a desired strategy for a sufficiently large discount factor. We then generalize our method to any LTL specification, for which the DRA may have an arbitrary number of accepting pairs; for such specifications, we establish a lower bound on the satisfaction probability. Lastly, we show the applicability of our RL approach on two robot planning case studies.

II. Preliminaries and Problem Statement

A. Stochastic (Turn-Based) Two-Player Games

We use turn-based stochastic games to model the interaction between the controller (i.e., Player 1) and unpredictable environment (i.e., Player 2), where actions have probabilistic outcomes. The controller can only choose actions in certain states; the rest of the states are in control of the environment.

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 Zavlanos, and Miroslav Pajic are with Duke University, Durham, NC 27708, USA, {alper.bozkurt, yu.wang094, michael.zavlanos, miroslav.pajic}@duke.edu

Definition 1 (Stochastic Games): A (labeled turn-based) two-player stochastic game is a tuple $\mathcal{G} = (S, (S_\mu, S_\nu), A, P, s_0, \mathrm{AP}, L)$, where S is a finite set of states; $S_\mu \subseteq S$ is the set of states where the controller chooses actions; $S_\nu = S \backslash S_\mu$ is the set of states where the environment chooses actions; s_0 is the initial state; A is a finite set of actions and $A(s)$ denotes the set of actions that can be taken in state $s \in S$; $P{:}S \times A \times S \to [0,1]$ is the transition probability function such that for all $s \in S$, $\sum_{s' \in S} P(s,a,s') = 1$ if $a \in A(s)$, and 0 otherwise; $s_0 \in S$ is an initial state; AP is a finite set of atomic propositions; and $L{:}S \to 2^{\mathrm{AP}}$ is a labeling function.

A *path* is an infinite sequence of game states $\sigma = s_0 s_1, \ldots$ such that for all $t \geq 0$, there exists an action $a \in A(s_t)$ where $P(s_t, a, s_{t+1}) > 0$. We use $\sigma[t]$, $\sigma[{:}t]$ and $\sigma[t{:}]$ to denote s_t, the prefix $s_0 \ldots s_t$ and the suffix $s_t s_{t+1} \ldots$ of the path, respectively. Strategies capture the players' behaviors, mapping the visited states to the actions available in the current state.

Definition 2 (Strategies): For a game $\mathcal{G}$, let $S_\mu^+ (S_\nu^+)$ denote the set of all **finite** prefixes $\sigma_{\mathrm{f}}^{s_\mu} (\sigma_{\mathrm{f}}^{s_\nu})$ ending with a state $s_\mu \in S_\mu (s_\nu \in S_\nu)$ of paths in the game. Then, a (pure) **control strategy** μ is a function $\mu{:}S_\mu^+ \to A$ such that $\mu(\sigma_{\mathrm{f}}^{s_\mu}) \in A(s_\mu)$ for all $\sigma_{\mathrm{f}}^{s_\mu} \in S_\mu^+$; a (pure) **environment strategy** ν is a function $\nu{:}S_\nu^+ \to A$ such that $\nu(\sigma_{\mathrm{f}}^{s_\nu}) \in A(s_\nu)$ for all $\sigma_{\mathrm{f}}^{s_\nu} \in S_\nu^+$; a strategy π is **memoryless**, if it only depends on the current state, i.e., $\pi(\sigma_f^s) = \pi(\sigma_f^{s'})$ if $s = s'$ for any σ_f^s and $\sigma_f^{s'}$, and thus can be defined as $\pi{:}S \to A$.

The induced Markov chain (MC) of game $\mathcal{G}$ under a strategy pair (μ, ν) is tuple $\mathcal{G}_{\mu,\nu} = (S, P_{\mu,\nu}, s_0, \mathrm{AP}, L)$, where

$$P_{\mu,\nu}(s,s') = \begin{cases} P(s,\mu(s),s') & \text{if } s \in S_\mu \\ P(s,\nu(s),s') & \text{if } s \in S_\nu \end{cases}.$$

We denote by $\mathcal{G}_{\mu,\nu}^s$ the MC resulting from changing the initial state from s_0 to $s \in S$ in $\mathcal{G}_{\mu,\nu}$, and use $\sigma \sim \mathcal{G}_{\mu,\nu}^s$ to denote a random path sampled from $\mathcal{G}_{\mu,\nu}^s$. Finally, a ***bottom strongly connected component*** (BSCC) of the (induced) MC $\mathcal{G}_{\mu,\nu}$ is a strongly connected component with no outgoing transitions; we use $\mathcal{B}(\mathcal{G}_{\mu,\nu})$ to denote the set of all BSCCs of $\mathcal{G}_{\mu,\nu}$.

B. LTL and Deterministic Rabin Automata

The desired behavior of a labeled stochastic game $\mathcal{G}$ are captured by LTL specifications, imposing requirements on the label sequences from infinite paths of the game [26]. In addition to the standard Boolean operators, LTL formulas can include two temporal operators, next ($\bigcirc$) and until (U), and any recursive combinations of the operators captured by the syntax $\varphi := \text{true} \mid a \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \mathsf{U} \varphi_2,\ a \in \mathrm{AP}$.

Satisfaction of an LTL formula φ for a path σ of the game $\mathcal{G}$, denoted by $\sigma \models \varphi$, is defined as follows: σ satisfies an atomic proposition a, if $a \in L(\sigma[0])$; σ satisfies $\bigcirc\varphi$ if $\sigma[1{:}]$ satisfies φ; and finally, $\sigma \models \varphi_1 \mathsf{U} \varphi_2$, if $\exists i . \sigma[i] \models \varphi_2$ and $\forall j < i . \sigma[j] \models \varphi_1$. Other temporal operators are derived as: (eventually) $\Diamond\varphi := \text{true } \mathsf{U}\ \varphi$; and (always) $\Box\varphi := \neg(\Diamond\neg\varphi)$.

Any LTL formula can be transformed into a DRA that accepts the language of all paths satisfying the formula [26].

Definition 3 (Deterministic Rabin Automata): A DRA is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathrm{Acc})$ where Q is a finite set of states; Σ is a finite alphabet; $\delta : Q \times \Sigma \to Q$ is the transition function; $q_0 \in Q$ is an initial state; and Acc is a set of k accepting pairs $\{(C_i, B_i)\}_{i=1}^k$ such that $C_i, B_i \subseteq Q$.

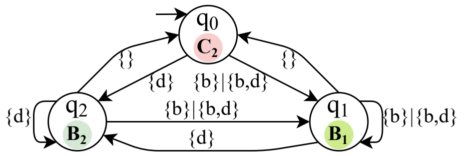


Fig. 1: A DRA derived from the LTL formula $\varphi = \Box\Diamond b \vee \Diamond\Box d$. Here, $B_1 = \{q_1\}$, $C_2 = \{q_0\}$ and $B_2 = \{q_2\}$ are the states in the Rabin acceptance condition $\mathrm{Acc} = \{(\varnothing, B_1), (C_2, B_2)\}$.

DRAs' acceptance criteria are defined based on infinite visits of some states. An infinite path σ is accepted by the DRA if it satisfies the ***Rabin condition***: there exists a pair $(C_i, B_i) \in \mathrm{Acc}$ such that states in C_i are visited finitely many times and at least one state in B_i visited is infinitely often – i.e., $\exists i$: $\mathrm{inf}(\sigma) \cap C_i = \varnothing \ \wedge\ \mathrm{inf}(\sigma) \cap B_i \neq \varnothing$, where $\mathrm{inf}(\sigma)$ denotes the set of states visited by the path σ infinitely many times. The ***Rabin index*** of an LTL formula is the minimal number of accepting pairs a DRA recognizing the formula can have. Without loss of generality, we assume that the number of accepting pairs k, equals the Rabin index.

Example 1: Fig. 1 shows a DRA of the formula $\varphi = \Box\Diamond b \vee \Diamond\Box d$, with Rabin acceptance sets $B_1 = \{q_1\}$, $C_2 = \{q_0\}$ and $B_2 = \{q_2\}$ (i.e., $\mathrm{Acc} = \{(\varnothing, B_1), (C_2, B_2)\}$ is the acceptance condition). Any path σ containing infinitely many states labeled with b induces an execution that visits q_1 infinitely many times; thereby satisfying the Rabin condition. Other paths satisfying the Rabin condition visit q_2 infinitely many times but q_0 only finitely many times, i.e., the paths that after some point, do not contain a state without the label d.

C. Reinforcement Learning for Stochastic Games

Let $R : S \to \mathbb{R}$ be a *reward function* and $\gamma \in (0,1)$ the discount factor for a given two-player zero-sum stochastic game $\mathcal{G}$. The *value* of a state s under a strategy pair (μ, ν)

$$v_{\mu,\nu}(s) = \mathbb{E}_{\sigma \sim \mathcal{G}_{\mu,\nu}} \left[\sum_{i=0}^{\infty} \gamma^i R(\sigma[t+i]) \,\middle|\, \sigma[t] = s \right], \quad (1)$$

for any fixed $t \in \mathbb{N}$, such that $Pr_{\sigma \sim \mathcal{G}_{\mu,\nu}}[\sigma[t] = s] > 0$. In the rest of the paper, we simplify our notation, omit the subscript $_{\sigma \sim \mathcal{G}_{\mu,\nu}}$ from the expectation and use $\mathbb{E}$ rather than $\mathbb{E}_{\sigma \sim \mathcal{G}_{\mu,\nu}}$.

The RL objective is to find an optimal *control strategy* μ_* that maximizes the values of every state under the worst *environment strategy*. A pure and memoryless optimal strategy always exists in two-player turn-based zero-sum stochastic games [27], [28]. The optimal values in these games satisfy $v_*(s) = \max_\mu \min_\nu v_{\mu,\nu}(s)$, where μ and ν are pure and memoryless control and environment strategies [29]. Also, the *optimal values* $v_*(s)$ satisfy the Bellman equations

$$v_*(s) = R(s) + \gamma \begin{cases} \max\limits_{a \in A(s)} \sum\limits_{s' \in S} P(s,a,s') v_*(s') & \text{if } s \in S_\mu, \\ \min\limits_{a \in A(s)} \sum\limits_{s' \in S} P(s,a,s') v_*(s') & \text{if } s \in S_\nu. \end{cases}$$

Model-free RL methods aim to learn the optimal values of the stochastic game, when neither the transition probabilities nor the game topology are known, without explicitly constructing a transition model of the game. A popular example is the minimax-Q method that generalizes the standard off-policy Q-learning algorithm to stochastic games. The minimax-Q method can learn the optimal values from any (likely non-optimal) strategies used during learning as long as all actions in each state are chosen infinitely often [28], [30].

D. Problem Formulation

We assume that the considered game $\mathcal{G}$ is fully observable for both players; i.e., both are aware of the current game state. The considered control synthesis problem is to find a strategy for the controller that *maximizes the probability that a produced path satisfies the specification in the worst case.*

To simplify our notation, we use $Pr^{\mathcal{G}}_{\mu,\nu}(s \models \varphi)$ to denote the probability of the paths that start from the state s and satisfy the formula φ under the strategy pair (μ, ν) – i.e.,

$$Pr^{\mathcal{G}}_{\mu,\nu}(s \models \varphi) \coloneqq Pr_{\sigma\sim\mathcal{G}^s_{\mu,\nu}}(\sigma \models \varphi); \tag{2}$$

we write $Pr_{\mu,\nu}(\mathcal{G} \models \varphi)$ for $Pr^{\mathcal{G}}_{\mu,\nu}(s_0 \models \varphi)$ and use Pr_* to denote the maximin probability $\max_\mu \min_\nu Pr_{\mu,\nu}$. We can now formally define the considered problem as follows.

Problem 1: Given a labeled turn-based stochastic game $\mathcal{G}$, where the transition probabilities are fully unknown, and an LTL specification φ, design a model-free RL algorithm that finds a *pure finite-memory* controller strategy μ_* such that

$$Pr_{\mu_*,\nu}(\mathcal{G} \models \varphi) \geq Pr_*(\mathcal{G} \models \varphi) \tag{3}$$

for any environment strategy ν.

III. Learning for Stochastic Rabin Games

In this section, we introduce our model-free RL approach to solve Problem 1. First, we describe the product game construction, a key step in reducing the problem of satisfying an LTL specification into the problem of satisfying a Rabin condition. We then consider the case where the DRA derived from the LTL objective φ has a single Rabin pair, and introduce our rewarding and discounting mechanisms based on it. We show that maximization of the discounted reward maximizes the minimal probability of satisfying the single pair Rabin condition, and thus the initial LTL objective. Finally, we provide a generalization to Rabin conditions with an arbitrary ($k{>}1$) number of accepting pairs; thereby allowing the use of our method for all possible LTL specifications.

A. Product Game Construction

By forming an augmented state space, Problem 1 can be reduced into finding a memoryless control strategy. Specifically, we compose the states of the game $\mathcal{G}$ with the states of the DRA $\mathcal{A}$ derived from the LTL specification φ. Then, the goal in this space is to satisfy the Rabin acceptance condition, for which memoryless control strategies suffice [31].

Definition 4 (Product Game): A product game $\mathcal{G}^\times{=}(S^\times, (S^\times_\mu, S^\times_\nu), A^\times, P^\times, s^\times_0, \mathrm{Acc}^\times)$ of a labeled turn-based stochastic game $\mathcal{G}{=}(S, (S_\mu, S_\nu), A, P, s_0, \mathrm{AP}, L)$ and a DRA $\mathcal{A} = (Q, 2^{\mathrm{AP}}, \delta, q_0, \mathrm{Acc})$ is defined as follows: $S^\times{=}S{\times}Q$ is the set of augmented states, the initial state $s^\times_0$ is $\langle s_0, q_0\rangle$, $S^\times_\mu{=}S_\mu{\times}Q$ and $S^\times_\nu{=}S_\nu{\times}Q$ are the sets of augmented controller and environment states, respectively; $A^\times{=}A$ is the set of actions; $P^\times{:}S^\times{\times}A^\times{\times}S^\times{\to}[0,1]$ is the transition function such that

$$P^\times(\langle s,q\rangle, a, \langle s',q'\rangle){=}\begin{cases} P(s,a,s') & \text{if } q'{=}\delta(q, L(s)) \\ 0 & \text{otherwise;}\end{cases}$$

and $\mathrm{Acc}^\times$ is a set of k accepting pairs $\{(C^\times_i, B^\times_i)\}^k_{i=1}$ where $C^\times_i{=}C_i{\times}Q$ and $B^\times_i{=}B_i{\times}Q$.

Similarly to DRAs, a path $\sigma^\times$ of the product game $\mathcal{G}^\times$ satisfies the Rabin condition if there exists i, such that $\inf(\sigma^\times){\cap}C^\times_i{=}\varnothing \ \wedge \ \inf(\sigma^\times){\cap}B^\times_i{\neq}\varnothing$. Finally, we refer to a product game with k accepting pairs as a Rabin(k) game.

There is a one-to-one correspondence between the paths in the product and original games. Similarly, a strategy for the product game induces a strategy in the original game and vice versa. However, the corresponding strategies in the original game require additional memory described by the DRA; i.e., the strategy in the original game may not be memoryless. Yet, the probability of satisfying the Rabin condition under any strategy pair in the product game is equal to the probability of satisfying the LTL formula in the original game under the corresponding strategy pair. Hence, in the rest of the section we focus on the product games, i.e., stochastic Rabin games; to simplify our notation, we omit the superscript $^\times$ and use $\mathcal{G}{=}(S, (S_\mu, S_\nu), A, P, s_0, \mathrm{Acc})$ and $s \in S$ instead of $\mathcal{G}^\times{=}(S^\times, (S^\times_\mu, S^\times_\nu), A^\times, P^\times s^\times_0, \mathrm{Acc}^\times)$ and $\langle s,q\rangle{\in}S^\times$.

B. Rabin(1) Condition to Discounted Rewards

We start with the case where the LTL formula φ has one accepting pair in the Rabin acceptance condition. In stochastic Rabin(1) games, where $\mathrm{Acc}{=}\{(C,B)\}$, the controller objective is to repeatedly visit some states in B and visit the states in C only finitely many times. The environment's goal is to prevent this from happening, which can be also expressed as a Rabin condition $\mathrm{Acc}'{=}\{(\varnothing, C), (B, S)\}$ with two accepting pairs. Thus, pure and memoryless strategies suffice for both players on the considered product game [31].

To solve Problem 1 for stochastic Rabin(1) games, our key idea is to assign small rewards to the states in B to encourage visiting B states as often as possible; but discount more compared to the other states to eliminate the importance of the frequency of visits. In addition, we discount even more in the states in C without giving any rewards, which diminishes the worth of the rewards to be obtained by visiting the states in B. The following theorem summarizes our key results.

Theorem 1: Consider a given turn-based stochastic Rabin(1) product game $\mathcal{G}$ and the return of any path σ as

$$G_t(\sigma) \coloneqq \sum\nolimits_{i=0}^{\infty} R_B(\sigma[t{+}i]) \cdot \prod\nolimits_{j=0}^{i-1} \Gamma_{B,C}(\sigma[t{+}j]), \tag{4}$$

where $\prod_{j=0}^{-1} \coloneqq 1$, $R_B : S \to [0,1)$ and $\Gamma_{B,C} : S \to (0,1)$ are the reward and the terminal functions defined as

$$R_B(s){\coloneqq}\begin{cases} 1{-}\gamma_B, & \text{if } s{\in}B \\ 0, & \text{if } s{\notin}B \end{cases}, \quad \Gamma_{B,C}(s){\coloneqq}\begin{cases} \gamma_B, & \text{if } s{\in}B \\ \gamma_C, & \text{if } s{\in}C \\ \gamma, & \text{otherwise}\end{cases}.$$

Here, γ_B and γ_C are functions of γ such that $0{<}\gamma_C(\gamma){<}\gamma_B(\gamma){<}\gamma{<}1$, and $\lim_{\gamma\to1^-} \gamma_B{=}\lim_{\gamma\to1^-} \gamma_C{=}1$, as well as

$$\lim_{\gamma\to1^-} \frac{1-\gamma}{1-\gamma_B(\gamma)} = \lim_{\gamma\to1^-} \frac{1-\gamma_B(\gamma)}{1-\gamma_C(\gamma)} = 0. \tag{5}$$

Then, the value of the game $v^\gamma_{\mu,\nu}$ (i.e., the expected return $\mathbb{E}\left[G_t(\sigma)\right]$) for the strategy pair (μ,ν) and the discount factor γ satisfies that for all states $s \in S$ it holds that

$$\lim_{\gamma\to1^-} v^\gamma_{\mu,\nu}(s) = Pr^{\mathcal{G}}_{\mu,\nu}(s \models \varphi_{B,C}); \tag{6}$$

here, $\varphi_{B,C} \coloneqq \Box\Diamond B \wedge \neg\Box\Diamond C$ is the Rabin condition of the DRA derived from the LTL objective φ.

Before proving Theorem 1, we use Lemma 1 to establish bounds on the state values. We show that if we replace a state on a path with a state in B, we obtain a larger or equal return;

if we replace it with a state in $S \setminus B$, we obtain a smaller or equal return; and the return is always between 0 and 1.

Lemma 1: For any path σ and a fixed $t \geq 0$, in a stochastic game with the path return defined as in (4), it holds that

$$\gamma_C G_{t+1}(\sigma) \leq \gamma G_{t+1}(\sigma) \leq G_t(\sigma) \leq 1-\gamma_B+\gamma_B G_{t+1}(\sigma), \quad (7)$$
$$0 \leq G_t(\sigma) \leq 1. \quad (8)$$

Proof: Due to space constraint, the proof is in [32]. ■

Under a strategy pair (μ, ν), it is straightforward to check the probability that a Rabin condition is satisfied in a game $\mathcal{G}$ (i.e., MC $\mathcal{G}_{\mu,\nu}$). All paths in the induced MC $\mathcal{G}_{\mu,\nu}$ eventually reach a BSCC $T \in \mathcal{B}(\mathcal{G}_{\mu,\nu})$ and visit its states infinitely many times. A path reaching a state in a BSCC that does not contain any state in B or C, does not satisfy the Rabin condition. We denote the set of all such states by $U_{\overline{BC}}$. Similarly, if a path reaches a state in a BSCC without any state in C but with a state in B, it satisfies the Rabin condition; finally, if it reaches a state in a BSCC that does contain a state from C, it does not satisfy the Rabin condition. We write U_B and U_C to denote the set of these states, respectively (formally defined in Lemma 2). This reasoning reduces finding the probability of satisfying the Rabin condition to finding the probability of reaching a state in U_B, which allows us to focus on the reachability objective $\varphi_{U_B} \coloneqq \Diamond U_B$ instead of $\varphi_{B,C}$ defined in Theorem 1.

We now show that the expected values of the returns (4) (i.e., the state values) reflect the Rabin acceptance condition.

Lemma 2: For any stochastic Rabin game $\mathcal{G}$ with Acc $= \{(C, B)\}$ under a strategy pair (μ, ν), it holds that:

$$\lim_{\gamma \to 1^-} v^{\gamma}_{\mu,\nu}(s) = 0 \text{ if } s \in U_{\overline{BC}}, \quad (9)$$
$$\lim_{\gamma \to 1^-} v^{\gamma}_{\mu,\nu}(s) = 1 \text{ if } s \in U_B, \quad (10)$$
$$\lim_{\gamma \to 1^-} v^{\gamma}_{\mu,\nu}(s) = 0 \text{ if } s \in U_C, \quad (11)$$

where the sets $U_{\overline{BC}}$, U_B and U_C are defined as:

$$U_{\overline{BC}} \coloneqq \{s_{\overline{BC}} \mid s_{\overline{BC}} \in T, T \in \mathcal{B}(\mathcal{G}_{\mu,\nu}), T \cap B = \varnothing, T \cap C = \varnothing\},$$
$$U_B \coloneqq \{s_B \mid \exists T \in \mathcal{B}(\mathcal{G}_{\mu,\nu}), s_B \in T \cap B, T \cap C = \varnothing\}, \quad (12)$$
$$U_C \coloneqq \{s_C \mid \exists T \in \mathcal{B}(\mathcal{G}_{\mu,\nu}), s_C \in T \cap C\} \quad (13)$$

Proof: Due to space constraint, the proof is in [32]. ■

We now provide the proof of Theorem 1.

Proof: (Theorem 1) We divide the expected return of a random path σ visiting a state $s \in S$ depending on whether it satisfies $\varphi_{U_B} \coloneqq \Diamond U_B$ or not – i.e.,

$$\begin{aligned} v^{\gamma}(s) =& \mathbb{E}[G_t(\sigma) \mid \sigma[t]{=}s, \sigma{\models}\Diamond U_B] Pr(\sigma{\models}\Diamond U_B) \\ &+ \mathbb{E}[G_t(\sigma) \mid \sigma[t]{=}s, \sigma{\not\models}\Diamond U_B] Pr(\sigma{\not\models}\Diamond U_B) \quad (14) \end{aligned}$$

for some fixed $t \in \mathbb{N}$. Notice that $\sigma{\not\models}\Diamond U_B$ implies $\sigma[t{:}]{\not\models}\Diamond U_B$, and $\sigma{\models}\Diamond U_B$ implies $\sigma[t{:}] \models \Diamond U_B$ almost surely. Hence, $Pr(s{\models}\Diamond U_B)$ and $Pr(s{\not\models}\Diamond U_B)$ can be replaced with $Pr(\sigma{\models}\Diamond U_B)$ and $Pr(\sigma{\not\models}\Diamond U_B)$, respectively.

After visiting the state s at time t, let L_t be the number of time steps until the first visit to a state in U_B in (12) – i.e.,

$$L_t = \min\{\tau \mid \sigma[t{+}\tau] \in U_B, \tau > 0\}. \quad (15)$$

Then, by Lemma 1, it holds that

$$\begin{aligned} v^{\gamma}(s) &\geq \mathbb{E}[G_t(\sigma) \mid \sigma[t]{=}s, \sigma{\models}\Diamond U_B] Pr(s{\models}\Diamond U_B) \\ &\geq \mathbb{E}\left[\gamma^{L_t} G_{t+L_t}(\sigma) \mid \sigma[t]{=}s, \sigma{\models}\Diamond U_B\right] Pr(s{\models}\Diamond U_B) \\ &\overset{①}{\geq} \mathbb{E}\left[\gamma^{L_t} \mid \sigma[t]{=}s, \sigma{\models}\Diamond U_B\right] \underline{v}^{\gamma}(U_B) Pr(s{\models}\Diamond U_B) \\ &\overset{②}{\geq} \gamma^{\mathbb{E}[L_t \mid \sigma[t]=s, \sigma\models\Diamond U_B]} \underline{v}^{\gamma}(U_B) Pr(s{\models}\Diamond U_B) = \\ &= \gamma^{l} \underline{v}^{\gamma}(U_B) Pr(s{\models}\Diamond U_B); \quad (16) \end{aligned}$$

here, $\underline{v}^{\gamma}(U_B) = \min_{s_B \in U_B} v^{\gamma}(s_B)$, l is constant, and ① and ② hold from the Markov property and Jensen's inequality.

Similarly, after leaving s at t, let L'_t be the number of time steps until the first time a state in $U_{\overline{BC}} \cup U_C$ is reached – i.e.,

$$L'_t = \min\left\{\tau \mid \sigma[t{+}\tau] \in U_{\overline{BC}} \cup U_C, \tau > 0\right\}. \quad (17)$$

Then, using Lemma 1 and the Markov property, it holds that

$$\begin{aligned} v^{\gamma}(s) &\leq \mathbb{E}[G_t(\sigma) \mid \sigma[t]{=}s, \sigma{\not\models}\Diamond U_B] Pr(s{\not\models}\Diamond U_B) + Pr(s{\models}\Diamond U_B) \\ &\leq \mathbb{E}[1{-}\gamma_B^{L'_t} \mid \sigma[t]{=}s, \sigma{\not\models}\Diamond U_B] Pr(s{\not\models}\Diamond U_B) + Pr(s{\models}\Diamond U_B) \\ &\leq 1{-}\gamma_B^{\mathbb{E}[L'_t \mid \sigma[t]=s, \sigma\not\models\Diamond U_B]} Pr(s{\not\models}\Diamond U_B) + Pr(s{\models}\Diamond U_B) \\ &= (1{-}\gamma_B^{l'}) Pr(s{\not\models}\Diamond U_B) + Pr(s{\models}\Diamond U_B), \quad (18) \end{aligned}$$

where l' is some constant. The upper bound (18) and the lower bound (16) (due to (10)) approach the probability $Pr(s{\models}\Diamond U_B)$ as $\gamma \to 1^-$, thereby concluding the proof. ■

C. Reduction to Stochastic Rabin(1) Games

To generalize our approach to Rabin conditions with k pairs, we construct k different stochastic Rabin(1) games and connect them with ε actions so that the controller is able to switch between the Rabin pairs it aims to satisfy.

Definition 5 (k-copy Game): Let $[n]$ denote the set $\{1, 2, \ldots, n\}$ for a positive integer n. For a given stochastic Rabin(k) game $\mathcal{G} = (S, (S_\mu, S_\nu), A, P, s_0, \text{Acc})$, with Acc$=\{(C_i, B_i)\}_{i=0}^{k}\}$, a k-copy game $\mathcal{G}^\star = (S^\star, (S^\star_\mu, S^\star_\nu), A^\star, P^\star, s^\star_0, \text{Acc}^\star)$ is a stochastic Rabin(1) game defined by:

- $S^\star = (S_\mu \times [2k]) \cup (S_\nu \times [k])$ is the augmented state set with $S^\star_\mu = S_\mu \times [k]$ the controller and $S^\star_\nu = S \setminus S^\star_\mu$ the environment states, and $s^\star_0 = \langle s_0, 1\rangle$ is the initial state;
- $A^\star = A \cup \{\varepsilon_i \mid i \in [k]\} \cup \{\varepsilon'\}$ is the set of actions;
- $P^\star : S^\star {\times} A^\star {\times} S^\star \to [0, 1]$ is the transition function defined as $P^\star(\langle s, i\rangle, a, \langle s', i'\rangle)$
$$= \begin{cases} P(s, a, s') & \text{if } a \in A, i = i', \\ 1 & \text{if } s \in S_\mu, s = s', a = \varepsilon_i, i' = k + i, \\ 1 & \text{if } s \in S_\nu, s = s', a = \varepsilon', i' = i - k, \\ 0, & \text{otherwise;} \end{cases}$$
- Acc$^\star = \{(C^\star, B^\star)\}$ is the Rabin accepting set where
$$C^\star \coloneqq \{\langle s, i\rangle \mid s \in C_i, i \in [k] \text{ or } s \in S_\mu, i \in [2k] \setminus [k]\},$$
$$B^\star \coloneqq \{\langle s, i\rangle \mid s \in B_i,\ i \in [k]\}.$$

Intuitively, the k-copy game $\mathcal{G}^\star$ consists of k exact copies of the original game $\mathcal{G}$ for each accepting pair, and a dummy state $\langle s, i{+}k\rangle$ for every controller state $s{\in}S_\mu$ for each copy $i{\in}[k]$. The controller can choose an ε_j in a state $\langle s, i\rangle$ and makes a transition to the dummy environment state $\langle s, j{+}k\rangle$ where the environment can only take the action ε', which makes a transition to the controller state $\langle s, j\rangle$. The idea here is to connect the k copies of the original game using these ε-actions so that in any state, the controller can jump to the j-th copy via an ε_j→a-dummy-state→ε'

sequence. All the dummy states belong to $C^\star$, prohibiting the ε-actions from being visited infinitely many times. Also, the only states belonging to $C^\star$ and $B^\star$ in the i-th copy are the ones belonging to C_i and B_i, respectively. This allows each accepting pair to be independently satisfied in its corresponding copy as stated in the following theorem.

Theorem 2: Let $\mathcal{G}^{(j)}$ be the stochastic Rabin(1) game obtained from a Rabin(k) game $\mathcal{G}$ by replacing Acc with $\{(C_j, B_j)\}$, and $W^{(j)}$ be the set of winning states such that for any $s \in W^{(j)}$, $Pr_*^{\mathcal{G}^{(j)}}(s \models \varphi_{B_j,C_j}) = 1$. Then, for any $\langle s,i\rangle \in S^\star$, it holds that

$$Pr_*^{\mathcal{G}^\star}(\langle s,i\rangle \models \varphi_{B^\star,C^\star}) = Pr_*^{\mathcal{G}^\star}(\langle s,i\rangle \models \Diamond V), \qquad (19)$$

where $V = \left\{\langle s',i'\rangle \in S^\star \mid s' \in \bigcup_{j=0}^k W^{(j)}\right\}$.

Proof: We prove (19) in two directions.
$\geq$: If a state $\langle s,i\rangle \in V$, then, by definition, there exists j such that $s \in W^{(j)}$. The controller can make a transition from $\langle s,i\rangle$ to $\langle s,j\rangle$ via the ε-actions and satisfy $\varphi_{B^\star,C^\star}$ by satisfying φ_{B_j,C_j}. Thus, the control strategy maximizing the reachability probabilities in the worst case also guarantees the satisfaction probabilities of at least the maxmin reachability probabilities i.e. $Pr_*^{\mathcal{G}^\star}(\langle s,i\rangle \models \varphi_{B^\star,C^\star}) \geq Pr_*^{\mathcal{G}^\star}(\langle s,i\rangle \models \Diamond V)$.
$\leq$: All the transitions via the ε-actions pass through a state in $C^\star$. Under any strategy pair, the BSCCs having ε-transitions of the induced MC are rejecting. Since without some ε-transitions, it is not possible for a BSCC to contain states from two different accepting pairs, an accepting BSCC must satisfy only a single pair. In addition, in the worst case, the satisfaction probability can be maximized by maximizing the probability of reaching a state that belongs to an accepting BSCC for any environment strategy. Thus, such states must be a winning state for some accepting pair, which implies that $Pr_*^{\mathcal{G}^\star}(\langle s,i\rangle \models \varphi_{B^\star,C^\star}) \leq Pr_*^{\mathcal{G}^\star}(\langle s,i\rangle \models \Diamond V)$. ■

Any control strategy $\mu^\star$ in $\mathcal{G}^\star$ has a corresponding finite-memory strategy μ in the Rabin(k) game $\mathcal{G}$, which can be captured by a deterministic finite automaton (DFA) with k states. In state s, the state of the DFA changes from state $i \in [k]$ to $j \in [k]$, if $\mu^\star(\langle s,i\rangle) = \varepsilon_j$; the DFA state stays the same and the control strategy μ chooses action $a \in A$ if $\mu^\star(\langle s,i\rangle) = a$. If $\mu^\star$ is a maximin strategy for $\mathcal{G}^\star$ then under the induced strategy μ, the controller satisfies the acceptance condition with probability that is not lower than the probability of reaching a winning state of an accepting pair.

Corollary 1: A maximin control strategy for $\mathcal{G}^\star$ of a stochastic $Rabin(k)$ game $\mathcal{G}$ induces a control strategy μ for $\mathcal{G}$ such that, for any environment strategy ν,

$$Pr_{\mu,\nu}\left(\mathcal{G} \models \varphi_{\text{Acc}}\right) \geq Pr_*\left(\mathcal{G} \models \Diamond W\right), \qquad (20)$$

where $\varphi_{\text{Acc}} := \bigvee_{(B_i,C_i)\in\text{Acc}} (\Box\Diamond B_i \wedge \neg\Box\Diamond C_i)$, and $W := \bigcup_{i=1}^k W^{(i)}$, with $W^{(i)}$ defined as in Theorem 2.

Proof: For any environment strategy ν in $\mathcal{G}$ we can construct a corresponding environment strategy $\nu^\star$ in $\mathcal{G}^\star$ such that $\nu^\star(\langle s,i\rangle) = \nu(s)$ for all $i \in [k]$ and $\nu^\star(\langle s,i\rangle) = \varepsilon'$ for all $[2k] \setminus [k]$. Note that the strategy pairs (μ,ν) and $(\mu^\star,\nu^\star)$ induce the same MCs. Since satisfying (C_j, B_j) satisfies φ_{Acc}, we have $Pr_{\mu,\nu}\left(\mathcal{G} \models \varphi_{\text{Acc}}\right) \geq Pr_{\mu^\star,\nu^\star}\left(\mathcal{G}^\star \models \varphi_{B^\star,C^\star}\right)$, which combined with Theorem 2 concludes the proof. ■

The induced control strategy μ guarantees a satisfaction probability that is larger than or equal to $Pr_*\left(\mathcal{G} \models \Diamond W\right)$. Note that computing the winning states in stochastic Rabin games is NP-Complete in the number accepting pairs [31]. Thus, it is unlikely to construct a stochastic Rabin(1) game from any given stochastic Rabin(k) game without an exponential blowup in the number of states.

D. Controller Synthesis via Reinforcement Learning

We now state the main result of our approach.

Theorem 3: For a given stochastic Rabin(k) game, there exists a γ' such that for any $\gamma \in (\gamma', 1)$, the minimax-Q using the reward and the discount functions in Theorem 1 is guaranteed to converge to a strategy μ satisfying (20).

Proof: The claim directly follows from Theorem 1, Corollary 1, and the fact that pure and memoryless strategies are finite and sufficient for both the controller and the environment in stochastic Rabin(1) games [31]. ■

For a given stochastic game and an LTL specification, we reduce the control synthesis problem to finding a maximin controller strategy in a stochastic Rabin(k) game $\mathcal{G}$ using the automata-based approach from Sec. III-A. This is further reduced to finding a strategy that maximizes the probability of satisfying a single Rabin pair in the worst case, in a stochastic Rabin(1) game $\mathcal{G}^\star$ using the method from Sec. III-C. Finally, we transform the objective of satisfying of a Rabin pair to a discounted reward maximization objective (Sec. III-B), allowing the use of RL to synthesize controllers.

Algorithm 1 summarizes the steps of our approach. Here, α is the learning rate and the bold $\mathbf{s}$ character denotes a state vector consisting of the state of the original game, the DRA state, and the index of Rabin pair. After the construction of $\mathcal{G}^\star$, the algorithm performs minimax-Q learning. In each iteration of learning, the algorithm derives an ϵ-greedy strategy pair, which means that under these strategies, the controller and the environment randomly choose their actions with probability of ϵ, as well choose their best action with probability of $1-\epsilon$. After the convergence, the algorithm returns a maximin control strategy $\mu_*^\star$ for $\mathcal{G}^\star$, which induces a finite-memory strategy for the original game, which guarantees the lower bound provided in Corollary 3.

IV. Experimental Results

Our RL-based control synthesis framework was implemented as a software tool [33] in *Python*; we used *Rabinizer 4* [34] to translate LTL formulas into DRAs, and minimax-Q learning using the presented reward and discount functions.

During learning, ϵ-greedy strategies are followed by both players after starting in a random state, and the episodes are terminated after 1K steps. We set the parameter ϵ and the learning rate α to 0.5 and gradually decreased them to 0.05 during learning; we used the discount factors of $\gamma_C = 1-(0.01)$, $\gamma_B = 1-(0.01)^2$ and $\gamma = 1-(0.01)^3$.

We considered robot planning tasks in 2-D grid worlds. A mobile robot can move to adjacent four cells in a single step using actions: *North, South, East* and *West.* When the robot tries to move to a cell with an obstacle, it hits the obstacle and stays in its previous position; once it moves to an absorbing cell it cannot leave it. In the figures, obstacles and absorbing cells are represented by filled and empty circles. Finally, each cell is labeled with a set of atomic propositions.

Algorithm 1: Model-free RL for control synthesis in stochastic games from LTL specifications.

```
Input: LTL formula $\varphi$, stochastic game $\mathcal{G}$
Translate $\varphi$ to a DRA $\mathcal{A}_\varphi$
Construct the product $\mathcal{G}^\times$ of $\mathcal{G}$ and $\mathcal{A}_\varphi$
Reduce $\mathcal{G}^\times$ to $\mathcal{G}^\star$; Initialize $Q(\mathbf{s}, a)$ on $\mathcal{G}^\star$
for $t = 0, 1, \ldots, T$ do
    Derive an $\epsilon$-greedy strategy pair $(\mu^\star, \nu^\star)$ from $Q$
    Take the action $a_t \leftarrow \begin{cases} \mu^\star(\mathbf{s}_t), & \mathbf{s}_t \in S^\star_\mu \\ \nu^\star(\mathbf{s}_t), & \mathbf{s}_t \in S^\star_\nu \end{cases}$
    Observe the next state $\mathbf{s}_{t+1}$
    $Q(\mathbf{s}_t, a_t) \leftarrow (1-\alpha)Q(\mathbf{s}_t, a_t) + \alpha R(\mathbf{s}_t)$
    $+\alpha\Gamma(\mathbf{s}_t) \cdot \begin{cases} \max_{a'} Q(\mathbf{s}_{t+1}, a'), & \mathbf{s}_{t+1} \in S^\star_\mu \\ \min_{a'} Q(\mathbf{s}_{t+1}, a'), & \mathbf{s}_{t+1} \in S^\star_\nu \end{cases}$
end for
return a greedy control strategy $\mu^\star_*$ from $Q$
```

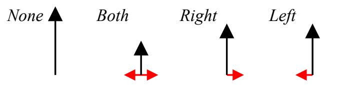


Fig. 2: The environment actions for control action *North*. The arrow lengths are proportional to the movement direction probabilities.

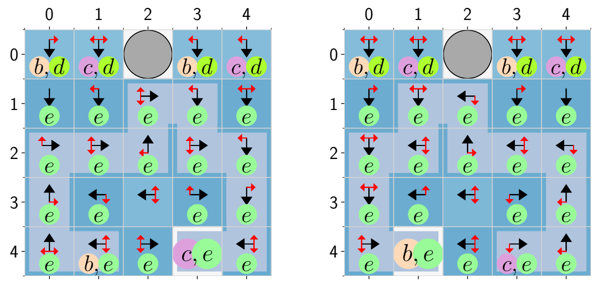


Fig. 3: The derived strategy: (a) from b to c, and (b) from c to b, under which φ_1 from (21) is almost surely satisfied, independently from the environment. The most likely paths are in a lighter blue.

A. Robust Control Design

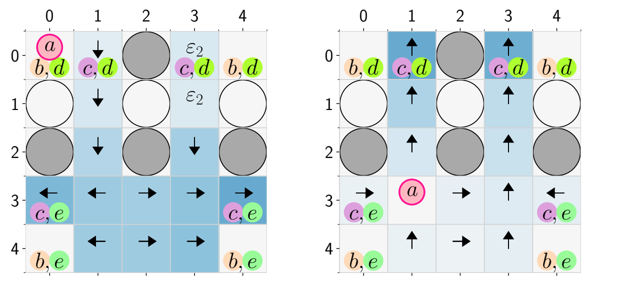
In our first case study, the robot could unpredictably move in a direction orthogonal to the intended direction. We model this source of nondeterminism as the environment, observing the actions of the controller and acting to minimize the probability that the controller achieves the given task. Specifically, the environment can reactively choose one of the actions: *None, Both, Right* and *Left* (Fig. 2). For *None*, the robot moves in the intended direction without any disturbance. With *Both*, it can go sideways with probability 0.2 (0.1 for each direction); with *Right*(*Left*), it moves as intended with probability of 0.9 and right(left) with probability of 0.1.

The control objective is to visit a state labeled with b and a state labeled with c repeatedly; and the safe states, labeled with either d or e, should not be left after a certain time, i.e.,

$$\varphi_1 = \square\lozenge b \wedge \square\lozenge c \wedge (\lozenge\square d \vee \lozenge\square e), \qquad (21)$$

which we translated into a DRA with two accepting pairs.

Fig. 3 shows the grid world and the derived strategy after 128K episodes. The objective cannot be satisfied by visiting the states (with b and c) at the top-left or top-right corner, since the environment could force the robot to leave the safe states. The only way to achieve the task is going from the state labeled with b to the state c and vice versa without leaving the safe states. With the strategy in Fig. 3, the robot does not move to the top row once leaving it, and does not visit the unsafe state in the middle regardless of the



(a) Adversary is at $(0, 0)$, i=1 (b) Adversary is at $(3, 1)$, i=2

Fig. 4: The control strategies for φ_2. The state values, represented by the shades of blue (the darker, the higher value), also capture how likely the controller satisfies the objective.

environment actions. With the strategies in Fig. 3(a),(b), the robot eventually reaches the states with b and c, respectively.

B. Avoiding Adversary

In this study, robot movement is not affected by the environment. Instead, the robot moves as intended with probability of 0.8 and goes to the right or the left side of the intended direction with probabilities of 0.1. Another agent, controlled by an adversary, can take the same four actions as the controller, with the same probability distribution.

The size of the state space here is $(5\times5)\times(5\times5) = 625$, as there are two independent agents. We define the labeling function as $L(\langle s_1, s_2\rangle) := \begin{cases} L(s_1), & s_1 \neq s_2, \\ L(s_1) \cup \{a\}, & s_1 = s_2, \end{cases}$ where the label a represents the state when both agents are in the same position (adversary 'catches' the robot). The robot objective is the same as in (21), except that it additionally needs to avoid the adversary at all costs, i.e., $\varphi_2 = \varphi_1 \wedge \square\neg a$.

Fig. 4 shows the control strategy obtained after 512K episodes. There are four safe zones: at the top-left, at the top-right, and two at the bottom part of the grid. The robot or the adversary can get trapped in a sink state with probability $p \geq 0.2$ while traveling between the top and the bottom parts of the grid. Thus, the optimal controller strategy is not to switch zones unless the adversary is in the same zone. For example, in Fig. 4a, if the robot is in the bottom part, the controller should not try to move the robot to the top-right part, a farther safe zone, because there is a chance ($p \geq 0.2$) that the adversary ends up with a sink state if (s)he tries to move to the bottom part. If the robot is in the top-right part, the controller should switch to the second Rabin pair via ε_2 and make the robot stay in the same zone. However, in Fig. 4b, the robot cannot stay in the bottom part because otherwise the adversary will eventually catch her or him.

V. Conclusions

In this paper, we introduced an RL-based approach for synthesis of controllers from LTL specifications in stochastic games. We reduced this problem to finding a control strategy in a stochastic Rabin game with a single accepting pair. We introduced a rewarding mechanism that transforms the objective of maximizing the (worst-case) probability of satisfying the Rabin condition into maximizing the discounted reward, and presented an RL algorithm to find such a strategy. We also generalized our approach to any LTL specification, with the Rabin condition having $k > 1$ accepting pairs, providing a lower bound on the satisfaction probabilities.

References

[1] Fahiem Bacchus and Froduald Kabanza. Using temporal logics to express search control knowledge for planning. *Artificial intelligence*, 116(1-2):123–191, 2000.

[2] Georgios E Fainekos, Hadas Kress-Gazit, and George J Pappas. Temporal logic motion planning for mobile robots. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2020–2025. IEEE, 2005.

[3] Hadas Kress-Gazit, Georgios E Fainekos, and George J Pappas. Where's waldo? sensor-based temporal logic motion planning. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3116–3121. IEEE, 2007.

[4] Hadas Kress-Gazit, Georgios E Fainekos, and George J Pappas. Translating structured English to robot controllers. *Advanced Robotics*, 22(12):1343–1359, 2008.

[5] Georgios E Fainekos, Antoine Girard, Hadas Kress-Gazit, and George J Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

[6] Hadas Kress-Gazit, Georgios E Fainekos, and George J Pappas. Temporal-logic-based reactive mission and motion planning. *IEEE transactions on robotics*, 25(6):1370–1381, 2009.

[7] Amit Bhatia, Lydia E Kavraki, and Moshe Y Vardi. Sampling-based motion planning with temporal goals. In *2010 IEEE International Conference on Robotics and Automation*, pages 2689–2696. IEEE, 2010.

[8] Morteza Lahijanian, Joseph Wasniewski, Sean B Andersson, and Calin Belta. Motion planning and control from temporal logic specifications with probabilistic satisfaction guarantees. In *2010 IEEE International Conference on Robotics and Automation*, pages 3227–3232. IEEE, 2010.

[9] Stephen L Smith, Jana Tůmová, 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.

[10] Tichakorn Wongpiromsarn, Ufuk Topcu, and Richard M Murray. Receding horizon temporal logic planning. *IEEE Transactions on Automatic Control*, 57(11):2817–2830, 2012.

[11] Meng Guo, Karl H Johansson, and Dimos V Dimarogonas. Revising motion planning under linear temporal logic specifications in partially known workspaces. In *2013 IEEE International Conference on Robotics and Automation*, pages 5025–5032. IEEE, 2013.

[12] Alphan Ulusoy, Stephen L Smith, Xu Chu Ding, Calin Belta, and Daniela Rus. Optimality and robustness in multi-robot path planning with temporal logic constraints. *The International Journal of Robotics Research*, 32(8):889–911, 2013.

[13] Indranil Saha, Rattanachai Ramaithitima, Vijay Kumar, George J Pappas, and Sanjit A Seshia. Automated composition of motion primitives for multi-robot systems from safe ltl specifications. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1525–1532. IEEE, 2014.

[14] Keliang He, Morteza Lahijanian, Lydia E Kavraki, and Moshe Y Vardi. Towards manipulation planning with temporal logic specifications. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 346–352. IEEE, 2015.

[15] Jana Tumova and Dimos V Dimarogonas. Multi-agent planning under local ltl specifications and event-based synchronization. *Automatica*, 70:239–248, 2016.

[16] Yasser Shoukry, Pierluigi Nuzzo, Ayca Balkan, Indranil Saha, Alberto L Sangiovanni-Vincentelli, Sanjit A Seshia, George J Pappas, and Paulo Tabuada. Linear temporal logic motion planning for teams of underactuated robots using satisfiability modulo convex programming. In *2017 IEEE 56th annual conference on decision and control (CDC)*, pages 1132–1137. IEEE, 2017.

[17] Andrew G Barto. Reinforcement learning: Connections, surprises, challenges. *AI Magazine*, 40(1), 2019.

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

[19] 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.

[20] Alper Kamil Bozkurt, Yu Wang, Michael M Zavlanos, and Miroslav Pajic. Control synthesis from linear temporal logic specifications using model-free reinforcement learning, 2019. arXiv:1909.07299 [cs.RO].

[21] Mohammadhosein Hasanbeig, Yiannis Kantaros, Alessandro Abate, Daniel Kroening, George J Pappas, and Insup Lee. Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantees, 2019. arXiv:1909.05304 [cs.LO].

[22] 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.

[23] Thomas A Henzinger and Nir Piterman. Solving games without determinization. In *International Workshop on Computer Science Logic*, pages 395–410. Springer, 2006.

[24] Min Wen and Ufuk Topcu. Probably approximately correct learning in stochastic games with temporal logic specifications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 3630–3636. AAAI Press, 2016.

[25] Pranav Ashok, Jan Křetínskỳ, and Maximilian Weininger. PAC statistical model checking for Markov decision processes and stochastic games. In *International Conference on Computer Aided Verification*, pages 497–519. Springer, 2019.

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

[27] Lloyd S Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100, 1953.

[28] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

[29] Junling Hu and Michael P Wellman. Nash q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov):1039–1069, 2003.

[30] Michael Bowling and Manuela Veloso. An analysis of stochastic game theory for multiagent reinforcement learning. Technical report, Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science, 2000.

[31] Krishnendu Chatterjee and Thomas A. Henzinger. A survey of stochastic ω-regular games. *Journal of Computer and System Sciences*, 78(2):394 – 413, 2012. Games in Verification.

[32] Alper Kamil Bozkurt, Yu Wang, Michael Zavlanos, and Miroslav Pajic. Model-free reinforcement learning for stochastic games with linear temporal logic objectives. *arXiv preprint arXiv:2010.01050*, 2020.

[33] A. K. Bozkurt, Y. Wang, M. M. Zavlanos, and M. Pajic. CSRL, 2020. `https://gitlab.oit.duke.edu/cpsl/csrl`.

[34] Jan Křetínskỳ, Tobias Meggendorfer, Salomon Sickert, and Christopher Ziegler. Rabinizer 4: from LTL to your favourite deterministic automaton. In *International Conference on Computer Aided Verification*, pages 567–577. Springer, 2018.