Reinforcement Learning of Control Policy for Linear Temporal Logic Specifications Using Limit-Deterministic Büchi Automata

Ryohei Oura, Ami Sakakibara, and Toshimitsu Ushio

***Abstract*— This letter proposes a novel reinforcement leaning method for the synthesis of a control policy satisfying a control specification described by a linear temporal logic formula. We assume that the controlled system is modeled by a Markov decision process (MDP). We transform the specification to a limit-deterministic Büchi automaton (LDBA) with several accepting sets that accepts all infinite sequences satisfying the formula. The LDBA is augmented so that it explicitly records the previous visits to accepting sets. We take a product of the augmented LDBA and the MDP, based on which we define a reward function. The agent gets rewards whenever state transitions are in an accepting set that has not been visited for a certain number of steps. Consequently, sparsity of rewards is relaxed and optimal circulations among the accepting sets are learned. We show that the proposed method can learn an optimal policy when the discount factor is sufficiently close to one.**

***Index Terms*— Reinforcement Learning, Linear Temporal Logic, Limit-Deterministic Büchi Automata.**

I. Introduction

Temporal logic has been developed in computer engineering as a useful formalism of formal specifications [1], [2]. A merit of temporal logics is its resemblance to natural languages and it has been widely used in several other areas of engineering. Especially, a complicated mission or task in computer-controlled systems such as robots can be described by a temporal logic specification precisely and many synthesis algorithms of a controller or a planner that satisfies the specification have been proposed [3]–[6]. Linear temporal logic (LTL) is often used as a specification language because of its rich expressiveness. It can explain many important ω-regular properties such as liveness, safety, and persistence [1]. It is known that the LTL specification is converted into an ω-automaton such as a nondeterministic Büchi automaton and a deterministic Rabin automaton [1], [7]. In the synthesis of a control policy for the LTL specification, we model a controlled system by a transition system that abstracts its dynamics, construct a product automaton of the transition system and the ω-automaton corresponding to the LTL specification, and compute a winning strategy of a game over the product automaton [7].

In general, there are uncertainties in a controlled system and we often use a Markov decision process (MDP) as a finite-state abstraction of the controlled system [8]. In the case where the probabilities are unknown a priori, we have two approaches to the synthesis of the control policy. One is robust control where we assume that state transition probabilities are in uncertainty sets [9] while the other is learning using samples [10].

Reinforcement learning (RL) is a useful approach to learning an optimal policy from sample behaviors of the controlled system [11]. In RL, we use a reward function that assigns a reward to each transition in the behaviors and evaluate a control policy by the return that is an expected (discounted) sum of the rewards along the behaviors. Thus, to apply RL to the synthesis of a control policy for the LTL specification, it is an important issue how to introduce the reward function, which depends on the acceptance condition of an ω-automaton converted from the LTL specification. A reward function based on the acceptance condition of a Rabin automaton was proposed in [10]. It was applied to a control problem where the controller optimizes a given control cost under the LTL constraint [12].

Recently, a limit-deterministic Büchi automaton (LDBA) is paid much attention to as an ω-automaton corresponding to the LTL specification [13]. The RL-based approaches to the syntehsis of a control policy using LDBAs have been proposed in [14]–[17]. To deal with the acceptance condition of an LDBA that accepts behaviors visiting all accepting sets infinitely often, the accepting frontier function was introduced in [14], [16]. The reward function is defined based on the function. However, the function is memoryless, that is, it does not provide the information of accepting sets that have been visited, which is important to improve learning performance. In this letter, we propose a novel method to augment an LDBA converted from a given LTL formula. Then, we define a reward function based on the acceptance condition of the product MDP of the augmented LDBA and the controlled system. As a result, we can learn a dynamic control policy that satisfies the LTL specification.

The rest of the letter is organized as follows. Section II reviews an MDP, LTL, and automata. Section III proposed a novel RL-based method for the synthesis of a control policy. Section IV presents a numerical example for which the previous method cannot learn a control policy but the proposed one can.

II. Preliminaries

A. Markov Decision Process

Definition 1: A (labeled) Markov decision process (MDP) is a tuple $M = (S, A, \mathcal{A}, P, s_{init}, AP, L)$, where S is a finite set of states, A is a finite set of actions, $\mathcal{A} : S \rightarrow 2^A$ is a

This work was partially supported by JST-ERATO HASUO Project Grant Number JPMJER1603, Japan, JST-Mirai Program Grant Number JPMJMI18B4, Japan, and JSPS KAKENHI Grant Number JP19J13487, Japan.

The authors are with the Graduate School of Engineering Science, Osaka University, Toyonaka 560-8531, Japan (e-mail: r-oura, sakakibara@hopf.sys.es.osaka-u.ac.jp; ushio@sys.es.osaka-u.ac.jp).

mapping that maps each state to the set of possible actions at the state, $P : S \times S \times A \to [0,1]$ is a transition probability such that $\sum_{s' \in S} P(s'|s,a) = 1$ for any state $s \in S$ and any action $a \in \mathcal{A}(s)$, $s_{init} \in S$ is the initial state, AP is a finite set of atomic propositions, and $L : S \times A \times S \to 2^{AP}$ is a labeling function that assigns a set of atomic propositions to each transition $(s,a,s') \in S \times A \times S$.

In the MDP M, an infinite path starting from a state $s_0 \in S$ is defined as a sequence $\rho = s_0 a_0 s_1 \ldots \in S(AS)^{\omega}$ such that $P(s_{i+1}|s_i,a_i) > 0$ for any $i \in \mathbb{N}_0$, where $\mathbb{N}_0$ is the set of natural numbers including zero. A finite path is a finite sequence in $S(AS)^*$. In addition, we sometimes represent ρ as ρ_{init} to emphasize that ρ starts from $s_0 = s_{init}$. For a path $\rho = s_0 a_0 s_1 \ldots$, we define the corresponding labeled path $L(\rho) = L(s_0,a_0,s_1)L(s_1,a_1,s_2)\ldots \in (2^{AP})^{\omega}$. $InfPath^M$ (resp., $FinPath^M$) is defined as the set of infinite (resp., finite) paths starting from $s_0 = s_{init}$ in the MDP M. For each finite path ρ, $last(\rho)$ denotes its last state.

Definition 2: A policy on an MDP M is defined as a mapping $\pi : FinPath^M \times \mathcal{A}(last(\rho)) \to [0,1]$. A policy π is a *positional* policy if for any $\rho \in FinPath^M$ and any $a \in \mathcal{A}(last(\rho))$, it holds that $\pi(\rho,a) = \pi(last(\rho),a)$ and there exists $a' \in \mathcal{A}(last(\rho))$ such that

$$\pi(\rho,a) = \begin{cases} 1 & \text{if } a = a', \\ 0 & \text{otherwise.} \end{cases}$$

Let $InfPath_{\pi}^{M}$ (resp., $FinPath_{\pi}^{M}$) be the set of infinite (resp., finite) paths starting from $s_0 = s_{init}$ in the MDP M under a policy π. The behavior of an MDP M under a policy π is defined on a probability space $(InfPath_{\pi}^{M}, \mathcal{F}_{InfPath_{\pi}^{M}}, Pr_{\pi}^{M})$.

A Markov chain induced by an MDP M with a positional policy π is a tuple $MC_{\pi} = (S_{\pi}, P_{\pi}, s_0, AP, L)$, where $S_{\pi} = S$, $P_{\pi}(s'|s) = P(s'|s,a)$ for $s, s' \in S$ and $a \in \mathcal{A}(s)$ such that $\pi(s,a) = 1$. The state set S_{π} of MC_{π} can be represented as a disjoint union of a set of transient states T_{π} and closed irreducible sets of recurrent states R_{π}^{j} with $j \in \{1,\ldots,h\}$, as $S_{\pi} = T_{\pi} \sqcup R_{\pi}^{1} \sqcup \ldots \sqcup R_{\pi}^{h}$ [18]. In the following, we say a "recurrent class" instead of a "closed irreducible set of recurrent states" for simplicity.

In an MDP M, we define a reward function $R : S \times A \times S \to \mathbb{R}_{\geq 0}$, where $\mathbb{R}_{\geq 0}$ is the set of nonnegative real numbers. The function denotes the immediate scalar bounded reward received after the agent performs an action a at a state s and reaches a next state s' as a result.

Definition 3: For a policy π on an MDP M, any state $s \in S$, and a reward function R, we define the expected discounted reward as

$$V^{\pi}(s) = \mathbb{E}^{\pi}[\sum_{n=0}^{\infty} \gamma^n R(S_n, A_n, S_{n+1}) | S_0 = s],$$

where $\mathbb{E}^{\pi}$ denotes the expected value given that the agent follows the policy π from the state s and $\gamma \in [0,1)$ is a discount factor. The function $V^{\pi}(s)$ is often referred to as a state-value function under the policy π. For any state-action pair $(s,a) \in S \times A$, we define an action-value function $Q^{\pi}(s,a)$ under the policy π as follows.

$$Q^{\pi}(s,a) = \mathbb{E}^{\pi}[\sum_{n=0}^{\infty} \gamma^n R(S_n, A_n, S_{n+1}) | S_0 = s, A_0 = a].$$

Definition 4: For any state s in S, a policy π^* is optimal if

$$\pi^* \in \arg\max_{\pi \in \Pi^{pos}} V^{\pi}(s),$$

where Π^{pos} is the set of positional policies over the state set S.

B. Linear Temporal Logic and Automata

In our proposed method, we use linear temporal logic (LTL) formulas to describe various constraints or properties and to systematically assign corresponding rewards. LTL formulas are constructed from a set of atomic propositions, Boolean operators, and temporal operators. We use the standard notations for the Boolean operators: $\top$ (true), $\neg$ (negation), and $\wedge$ (conjunction). LTL formulas over a set of atomic propositions AP are defined as

$$\varphi ::= \top \mid \alpha \in AP \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \mathbf{X}\varphi \mid \varphi_1 \mathbf{U} \varphi_2,$$

where φ, φ_1, and φ_2 are LTL formulas. Additional Boolean operators are defined as $\bot := \neg\top$, $\varphi_1 \vee \varphi_2 := \neg(\neg\varphi_1 \wedge \neg\varphi)$, and $\varphi_1 \Rightarrow \varphi_2 := \neg\varphi_1 \vee \varphi_2$. The operators $\mathbf{X}$ and $\mathbf{U}$ are called "next" and "until", respectively. Using the operator $\mathbf{U}$, we define two temporal operators: (1) *eventually*, $\mathbf{F}\varphi := \top\mathbf{U}\varphi$ and (2) *always*, $\mathbf{G}\varphi := \neg\mathbf{F}\neg\varphi$.

Let M be an MDP. For an infinite path $\rho = s_0 a_0 s_1 \ldots$ of M with $s_0 \in S$, let $\rho[i]$ be the i-th state of ρ i.e., $\rho[i] = s_i$ and let $\rho[i:]$ be the i-th suffix $\rho[i:] = s_i a_i s_{i+1} \ldots$.

Definition 5: For an LTL formula φ, an MDP M, and an infinite path $\rho = s_0 a_0 s_1 \ldots$ of M with $s_0 \in S$, the satisfaction relation $M, \rho \models \varphi$ is recursively defined as follows.

$$\begin{aligned}
& M,\rho \models \top, \\
& M,\rho \models \alpha \in AP && \Leftrightarrow \alpha \in L(s_0,a_0,s_1), \\
& M,\rho \models \varphi_1 \wedge \varphi_2 && \Leftrightarrow M,\rho \models \varphi_1 \wedge M,\rho \models \varphi_2, \\
& M,\rho \models \neg\varphi && \Leftrightarrow M,\rho \not\models \varphi, \\
& M,\rho \models \mathbf{X}\varphi && \Leftrightarrow M,\rho[1:] \models \varphi, \\
& M,\rho \models \varphi_1 \mathbf{U} \varphi_2 && \Leftrightarrow \\
& \quad \exists j \geq 0,\ M,\rho[j:] \models \varphi_2 \wedge \forall i, 0 \leq i < j,\ M,\rho[i:] \models \varphi_1.
\end{aligned}$$

The next operator $\mathbf{X}$ requires that φ is satisfied by the next state suffix of ρ. The until operator $\mathbf{U}$ requires that φ_1 holds true until φ_2 becomes true over the path ρ. In the following, we write $\rho \models \varphi$ for simplicity without referring to MDP M.

For any policy π, we denote the probability of all paths starting from s_{init} on the MDP M that satisfy an LTL formula φ under the policy π as

$$Pr_{\pi}^{M}(s_{init} \models \varphi) := Pr_{\pi}^{M}(\{\rho_{init} \in InfPath_{\pi}^{M} ; \rho_{init} \models \varphi\}).$$

We say that an LTL formula φ is satisfied by a positional policy π if

$$Pr_{\pi}^{M}(s_{init} \models \varphi) > 0.$$

Any LTL formula φ can be converted into various automata, namely finite state machines that recognize all words satisfying φ. We define a generalized Büchi automaton at the beginning, and then introduce a limit-deterministic Büchi automaton.

Definition 6: A transition-based generalized Büchi automaton (tGBA) is a tuple $B = (X,\ x_{init},\ \Sigma,\ \delta,\ \mathcal{F})$, where X is a finite set of states, $x_{init} \in X$ is the initial state, Σ is an input alphabet, $\delta \subset X \times \Sigma \times X$ is a set of transitions, and $\mathcal{F} = \{F_1, \ldots, F_n\}$ is an acceptance condition, where for each $j \in \{1, \ldots, n\}$, $F_j \subset \delta$ is a set of accepting transitions and called an accepting set.

Let Σ^ω be the set of all infinite words over Σ and let an infinite run be an infinite sequence $r = x_0\sigma_0x_1 \ldots \in X(\Sigma X)^\omega$ where $(x_i, \sigma_i, x_{i+1}) \in \delta$ for any $i \in \mathbb{N}_0$. An infinite word $w = \sigma_0\sigma_1 \ldots \in \Sigma^\omega$ is accepted by B_φ if and only if there exists an infinite run $r = x_0\sigma_0x_1 \ldots$ starting from $x_0 = x_{init}$ such that $inf(r) \cap F_j \neq \emptyset$ for each $F_j \in \mathcal{F}$, where $inf(r)$ is the set of transitions that occur infinitely often in the run r.

Definition 7: A tGBA $B = (X, x_{init}, \Sigma, \delta, \mathcal{F})$ is limit-deterministic (tLDBA) if the following conditions hold.

- $\exists X_{initial},\ X_{final} \subset X$ s.t. $X = X_{initial} \cup X_{final} \wedge X_{initial} \cap X_{final} = \emptyset$,
- $F_j \subset X_{final} \times \Sigma \times X_{final}$, $\forall j \in \{1, ..., n\}$,
- $|\{(x, \sigma, x') \in \delta; x' \in X_{initial}\}| \leq 1$, $\forall x \in X_{initial}, \forall \sigma \in \Sigma$,
- $|\{(x, \sigma, x') \in \delta; x' \in X_{final}\}| \leq 1$, $\forall x \in X_{final}, \forall \sigma \in \Sigma$,
- $|\{(x, \sigma, x') \in \delta; x' \in X_{initial}\}|$=0, $\forall x \in X_{final}, \forall \sigma \in \Sigma$.

A tLDBA is a tGBA whose state set can be partitioned into the initial part $X_{initial}$ and the final part X_{final}, and they are connected by a single "guess". The final part has all accepting sets. The transitions in each part are deterministic. It is known that, for any LTL formula φ, there exists a tLDBA that accepts all words satisfying φ [13]. In particular, we represent a tLGBA recognizing an LTL formula φ as B_φ, whose input alphabet is given by $\Sigma = 2^{AP}$.

III. Reinforcement-Learning-Based Synthesis of Control Policy

We introduce an automaton augmented with binary vectors. The automaton can explicitly represent whether transitions in each accepting set occur at least once, and ensure transitions in each accepting set occur infinitely often.

Let $V = \{(v_1, \ldots, v_n)^T\ ;\ v_i \in \{0, 1\},\ i \in \{1, \ldots, n\}\}$ be a set of binary-valued vectors, and let $\mathbf{1}$ and $\mathbf{0}$ be the n-dimentional vectors with all elements 1 and 0, respectively. In order to augment a tLDBA B_φ, we introduce three functions $visitf : \delta \to V$, $reset : V \to V$, and $Max : V \times V \to V$ as follows. For any $e \in \delta$, $visitf(e) = (v_1, \ldots, v_n)^T$, where

$$v_i = \begin{cases} 1 & \text{if } e \in F_i, \\ 0 & \text{otherwise.} \end{cases}$$

For any $v \in V$,

$$reset(v) = \begin{cases} \mathbf{0} & \text{if } v = \mathbf{1}, \\ v & \text{otherwise.} \end{cases}$$

For any $v, u \in V$, $Max(v, u) = (l_1, \ldots, l_n)^T$, where $l_i = max\{v_i, u_i\}$ for any $i \in \{1, \ldots, n\}$.

Intuitively, each vector v represents which accepting sets have been visited. The function $visitf$ returns a binary vector whose i-th element is 1 if and only if a transition in the accepting set F_i occurs. The function $reset$ returns the zero vector $\mathbf{0}$ if at least one transition in each accepting set has occurred after the latest reset. Otherwise, it returns the input vector without change.

Definition 8: For a tLDBA $B_\varphi = (X, x_{init}, \Sigma, \delta, \mathcal{F})$, its augmented automaton is a tLDBA $\bar{B}_\varphi$ = $(\bar{X}, \bar{x}_{init}, \bar{\Sigma}, \bar{\delta}, \bar{\mathcal{F}})$, where $\bar{X} = X \times V$, $\bar{x}_{init} = (x_{init}, \mathbf{0})$, $\bar{\Sigma} = \Sigma$, $\bar{\delta}$ is defined as $\bar{\delta}$ = $\{((x, v), \bar{\sigma}, (x', v')) \in \bar{X} \times \bar{\Sigma} \times \bar{X}\ ;\ (x, \bar{\sigma}, x') \in \delta,\ v' = reset(Max(v, visitf((x, \bar{\sigma}, x'))))\}$, and $\bar{\mathcal{F}} = \{\bar{F}_1, \ldots, \bar{F}_n\}$ is defined as $\bar{F}_i = \{((x, v), \bar{\sigma}, (x', v')) \in \bar{\delta}\ ;\ (x, \sigma, x') \in F_i,\ v_i = 0,\ visitf((x, \bar{\sigma}, x'))_i = 1\}$ for each $i \in \{1, ..., n\}$, where $visitf((x, \bar{\sigma}, x'))_i$ is the i-th element of $visitf((x, \bar{\sigma}, x'))$.

Definition 9: Given an augmented tLDBA $\bar{B}_\varphi$ and an MDP M, a tuple $M \otimes \bar{B}_\varphi = M^\otimes = (S^\otimes, A^\otimes, \mathcal{A}^\otimes, s^\otimes_{init}, P^\otimes, \delta^\otimes, \mathcal{F}^\otimes)$ is a product MDP, where $S^\otimes = S \times \bar{X}$ is the finite set of states, $A^\otimes = A$ is the finite set of actions, $\mathcal{A}^\otimes : S^\otimes \to 2^{A^\otimes}$ is the mapping defined as $\mathcal{A}^\otimes((s, \bar{x})) = \mathcal{A}(s)$, $s^\otimes_{init} = (s_{init}, \bar{x}_{init})$ is the initial states, $P^\otimes : S^\otimes \times S^\otimes \times A^\otimes \to [0, 1]$ is the transition probability defined as

$$P^\otimes(s^{\otimes\prime}|s^\otimes, a) = \begin{cases} P(s'|s, a) & \text{if } (\bar{x}, L((s, a, s')), \bar{x}') \in \bar{\delta}, \\ 0 & \text{otherwise,} \end{cases}$$

$\delta^\otimes = \{(s^\otimes, a, s^{\otimes\prime}) \in S^\otimes \times A^\otimes \times S^\otimes; P^\otimes(s^{\otimes\prime}|s^\otimes, a) > 0\}$ is the set of transitions, and $\mathcal{F}^\otimes = \{\bar{F}^\otimes_1, \ldots, \bar{F}^\otimes_n\}$ is the acceptance condition, where $\bar{F}^\otimes_i = \{((s, \bar{x}), a, (s', \bar{x}')) \in \delta^\otimes\ ;\ (\bar{x}, L(s, a, s'), \bar{x}') \in \bar{F}_i\}$ for each $i \in \{1, \ldots, n\}$.

Definition 10: The reward function $\mathcal{R} : S^\otimes \times A^\otimes \times S^\otimes \to \mathbb{R}_{\geq 0}$ is defined as

$$\mathcal{R}(s^\otimes, a, s^{\otimes\prime}) = \begin{cases} r_p \text{ if } \exists i \in \{1, \ldots, n\},\ (s^\otimes, a, s^{\otimes\prime}) \in \bar{F}^\otimes_i, \\ 0 \text{ otherwise,} \end{cases}$$

where r_p is a positive value.

Under the product MDP $M^\otimes$ and the reward function $\mathcal{R}$, which is based on the acceptance condition of $M^\otimes$, we show that if there exists a positional policy π satisfying the LTL specification φ, maximizing the expected discounted reward produces a policy satisfying φ.

For a Markov chain $MC^\otimes_\pi$ induced by a product MDP $M^\otimes$ with a positional policy π, let $S^\otimes_\pi = T^\otimes_\pi \sqcup R^{\otimes 1}_\pi \sqcup \ldots \sqcup R^{\otimes h}_\pi$ be the set of states in $MC^\otimes_\pi$, where $T^\otimes_\pi$ is the set of transient states and $R^{\otimes i}_\pi$ is the recurrent class for each $i \in \{1, \ldots, h\}$, and let $R(MC^\otimes_\pi)$ be the set of all recurrent classes in $MC^\otimes_\pi$. Let $\delta^\otimes_{\pi,i}$ be the set of transtions in a recurrent class $R^{\otimes i}_\pi$, namely $\delta^\otimes_{\pi,i} = \{(s^\otimes, a, s^{\otimes\prime}) \in \delta^\otimes; s^\otimes \in R^{\otimes i}_\pi,\ P^\otimes(s^{\otimes\prime}|s^\otimes, a) > 0\}$, and let $P^\otimes_\pi : S^\otimes_\pi \times S^\otimes_\pi \to [0, 1]$ be the transition probability under π.

Lemma 1: For any policy π and any recurrent class $R^{\otimes i}_\pi$ in the Markov chain $MC^\otimes_\pi$, $MC^\otimes_\pi$ satisfies one of the following conditions.

1) $\delta^{\otimes}_{\pi,i} \cap \bar{F}^{\otimes}_{j} \neq \emptyset$, $\forall j \in \{1, \ldots, n\}$,
2) $\delta^{\otimes}_{\pi,i} \cap \bar{F}^{\otimes}_{j} = \emptyset$, $\forall j \in \{1, \ldots, n\}$.

Proof: Suppose that $MC^{\otimes}_{\pi}$ satisfies neither conditions 1 nor 2. Then, there exists a policy π, $i \in \{1, \ldots, h\}$, and j_1, $j_2 \in \{1, \ldots, n\}$ such that $\delta^{\otimes}_{\pi,i} \cap \bar{F}^{\otimes}_{j_1} = \emptyset$ and $\delta^{\otimes}_{\pi,i} \cap \bar{F}^{\otimes}_{j_2} \neq \emptyset$. In other words, there exists a nonempty and proper subset $J \in 2^{\{1,\ldots,n\}} \setminus \{\{1, \ldots, n\}, \emptyset\}$ such that $\delta^{\otimes}_{\pi,i} \cap \bar{F}^{\otimes}_{j} \neq \emptyset$ for any $j \in J$. For any transition $(s^{\otimes}, a, s^{\otimes\prime}) \in \delta^{\otimes}_{\pi,i} \cap \bar{F}^{\otimes}_{j}$, the following equation holds by the properties of the recurrent states in $MC^{\otimes}_{\pi}$ [18].

$$\sum_{k=0}^{\infty} p^k((s^{\otimes}, a, s^{\otimes\prime}), (s^{\otimes}, a, s^{\otimes\prime})) = \infty, \qquad (1)$$

where $p^k((s^{\otimes}, a, s^{\otimes\prime}), (s^{\otimes}, a, s^{\otimes\prime}))$ is the probability that the transition $(s^{\otimes}, a, s^{\otimes\prime})$ occurs again after the occurrence of itself in k time steps. Eq. (1) means that the agent obtains a reward infinitely often. This contradicts the definition of the acceptance condition of the product MDP $M^{\otimes}$. ■

Lemma 1 implies that for an LTL formula φ if a path ρ under a policy π does not satisfy φ, then the agent obtains no reward in recurrent classes; otherwise there exists at least one recurrent class where the agent obtains rewards infinitely often.

Theorem 1: Let $M^{\otimes}$ be the product MDP corresponding to an MDP M and an LTL formula φ. If there exists a positional policy satisfying φ, then there exists a discount factor γ^* such that any algorithm that maximizes the expected reward with $\gamma > \gamma^*$ will find a positional policy satisfying φ.

Proof: Suppose that π^* is an optimal policy but does not satisfy the LTL formula φ. Then, for any recurrent class $R^{\otimes i}_{\pi^*}$ in the Markov chain $MC^{\otimes}_{\pi^*}$ and any accepting set $\bar{F}^{\otimes}_{j}$ of the product MDP $M^{\otimes}$, $\delta^{\otimes}_{\pi^*,i} \cap \bar{F}^{\otimes}_{j} = \emptyset$ holds by Lemma 1. Thus, the agent under the policy π^* can obtain rewards only in the set of transient states. We consider the best scenario in the assumption. Let $p^k(s, s')$ be the probability of going to a state s' in k time steps after leaving the state s, and let $Post(T^{\otimes}_{\pi^*})$ be the set of states in recurrent classes that can be transitioned from states in $T^{\otimes}_{\pi^*}$ by one action. For the initial state $s^{\otimes}_{init}$ in the set of transient states, it holds that

$$\begin{aligned} V^{\pi^*}(s^{\otimes}_{init}) = & \sum_{k=0}^{\infty} \sum_{s^{\otimes} \in T^{\otimes}_{\pi^*}} \gamma^k p^k(s^{\otimes}_{init}, s^{\otimes}) \\ & \sum_{s^{\otimes\prime} \in T^{\otimes}_{\pi^*} \cup Post(T^{\otimes}_{\pi^*})} P^{\otimes}_{\pi^*}(s^{\otimes\prime} | s^{\otimes}) \mathcal{R}(s^{\otimes}, a, s^{\otimes\prime}) \\ \leq & \; r_p \sum_{k=0}^{\infty} \sum_{s^{\otimes} \in T^{\otimes}_{\pi^*}} \gamma^k p^k(s^{\otimes}_{init}, s^{\otimes}), \end{aligned}$$

where the action a is selected by π^*. By the property of the transient states, for any state $s^{\otimes}$ in $T^{\otimes}_{\pi^*}$, there exists a bounded positive value m such that $\sum_{k=0}^{\infty} \gamma^k p^k(s^{\otimes}_{init}, s^{\otimes}) \leq \sum_{k=0}^{\infty} p^k(s^{\otimes}_{init}, s^{\otimes}) < m$ [18]. Therefore, there exists a bounded positive value $\bar{m}$ such that $V^{\pi^*}(s^{\otimes}_{init}) < \bar{m}$. Let $\bar{\pi}$ be a positional policy satisfying φ. We consider the following two cases.

1) Assume that the initial state $s^{\otimes}_{init}$ is in a recurrent class $R^{\otimes i}_{\bar{\pi}}$ for some $i \in \{1, \ldots, h\}$. For any accepting set $\bar{F}^{\otimes}_{j}$, $\delta^{\otimes}_{\bar{\pi},i} \cap \bar{F}^{\otimes}_{j} \neq \emptyset$ holds by the definition of $\bar{\pi}$. The expected discounted reward for $s^{\otimes}_{init}$ is given by

$$\begin{aligned} V^{\bar{\pi}}(s^{\otimes}_{init}) = & \sum_{k=0}^{\infty} \sum_{s^{\otimes} \in R^{\otimes i}_{\bar{\pi}}} \gamma^k p^k(s^{\otimes}_{init}, s^{\otimes}) \\ & \sum_{s^{\otimes\prime} \in R^{\otimes i}_{\bar{\pi}}} P^{\otimes}_{\bar{\pi}}(s^{\otimes\prime} \mid s^{\otimes}) \mathcal{R}(s^{\otimes}, a, s^{\otimes\prime}), \end{aligned}$$

where the action a is selected by $\bar{\pi}$. Since $s^{\otimes}_{init}$ is in $R^{\otimes i}_{\bar{\pi}}$, there exists a positive number $\bar{k} = \min\{k \; ; \; k \geq n, p^k(s^{\otimes}_{init}, s^{\otimes}_{init}) > 0\}$ [18]. We consider the worst scenario in this case. It holds that

$$\begin{aligned} & V^{\bar{\pi}}(s^{\otimes}_{init}) \\ \geq & \sum_{k=n}^{\infty} p^k(s^{\otimes}_{init}, s^{\otimes}_{init})(\gamma^k + \gamma^{k-1} + \ldots + \gamma^{k-n+1}) r_p \\ \geq & \sum_{k=1}^{\infty} p^{k\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init})(\gamma^{k\bar{k}} + \ldots + \gamma^{k\bar{k}-n+1}) r_p \\ > & r_p \sum_{k=1}^{\infty} \gamma^{k\bar{k}} p^{k\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init}), \end{aligned}$$

whereas all states in $R(MC^{\otimes}_{\bar{\pi}})$ are positive recurrent because $|S^{\otimes}| < \infty$ [19]. Obviously, $p^{k\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init}) \geq (p^{\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init}))^k > 0$ holds for any $k \in (0, \infty)$ by the Chapman-Kolmogorov equation [18]. Furthermore, we have $\lim_{k \to \infty} p^{k\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init}) > 0$ by the property of irreducibility and positive recurrence [20]. Hence, there exists $\bar{p}$ such that $0 < \bar{p} < p^{k\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init})$ for any $k \in (0, \infty]$ and we have

$$V^{\bar{\pi}}(s^{\otimes}_{init}) > r_p \bar{p} \gamma^{\bar{k}} p^{\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init}) \frac{1}{1 - \gamma^{\bar{k}}}.$$

Therefore, for any $\bar{m} \in (V^{\pi^*}(s^{\otimes}_{init}), \infty)$ and any $r_p < \infty$, there exists $\gamma^* < 1$ such that $\gamma > \gamma^*$ implies $V^{\bar{\pi}}(s^{\otimes}_{init}) > r_p \bar{p} \gamma^{\bar{k}} p^{\bar{k}}(s^{\otimes}_{init}, s^{\otimes}_{init}) \frac{1}{1-\gamma^{\bar{k}}} > \bar{m}$.

2) Assume that the initial state $s^{\otimes}_{init}$ is in the set of transient states $T^{\otimes}_{\bar{\pi}}$. $P^{M^{\otimes}}_{\bar{\pi}}(s^{\otimes}_{init} \models \varphi) > 0$ holds by the definition of $\bar{\pi}$. For a recurrent class $R^{\otimes i}_{\bar{\pi}}$ such that $\delta^{\otimes}_{\bar{\pi},i} \cap \bar{F}^{\otimes}_{j} \neq \emptyset$ for each accepting set $\bar{F}^{\otimes}_{j}$, there exist a number $\bar{l} > 0$, a state $\hat{s}^{\otimes}$ in $Post(T^{\otimes}_{\bar{\pi}}) \cap R^{\otimes i}_{\bar{\pi}}$, and a subset of transient states $\{s^{\otimes}_{1}, \ldots, s^{\otimes}_{\bar{l}-1}\} \subset T^{\otimes}_{\bar{\pi}}$ such that $p(s^{\otimes}_{init}, s^{\otimes}_{1}) > 0$, $p(s^{\otimes}_{i}, s^{\otimes}_{i+1}) > 0$ for $i \in \{1, ..., \bar{l} - 2\}$, and $p(s^{\otimes}_{\bar{l}-1}, \hat{s}^{\otimes}) > 0$ by the property of transient states. Hence, it holds that $p^{\bar{l}}(s^{\otimes}_{init}, \hat{s}^{\otimes}) > 0$ for the state $\hat{s}^{\otimes}$. Thus, by ignoring rewards in $T^{\otimes}_{\bar{\pi}}$, we

have

$$\begin{aligned} V^{\bar{\pi}}(s^{\otimes}_{init}) \geq\ & P^{M^{\otimes}}_{\bar{\pi}}(s^{\otimes}_{init} \models \varphi)\gamma^{\bar{l}} p^{\bar{l}}(s^{\otimes}_{init}, \hat{s}^{\otimes}) \\ & \sum_{k=0}^{\infty} \sum_{s^{\otimes\prime} \in R^{\otimes i}_{\bar{\pi}}} \gamma^{k} p^{k}(\hat{s}^{\otimes}, s^{\otimes\prime}) \\ & \sum_{s^{\otimes\prime\prime} \in R^{\otimes i}_{\bar{\pi}}} P^{\otimes}_{\bar{\pi}}(s^{\otimes\prime\prime} | s^{\otimes\prime}) \mathcal{R}(s^{\otimes\prime}, a, s^{\otimes\prime\prime}) \\ >\ & P^{M^{\otimes}}_{\bar{\pi}}(s^{\otimes}_{init} \models \varphi)\gamma^{\bar{l}} p^{\bar{l}}(s^{\otimes}_{init}, \hat{s}^{\otimes}) \\ & r_p \bar{p} \gamma^{\bar{k}'} p^{\bar{k}'}(\hat{s}^{\otimes}, \hat{s}^{\otimes}) \frac{1}{1-\gamma^{\bar{k}'}}, \end{aligned}$$

where $\bar{k}' \geq n$ is a constant and $0 < \bar{p} < p^{k\bar{k}'}(\hat{s}^{\otimes}, \hat{s}^{\otimes})$ for any $k \in (0, \infty]$. Therefore, for any $\bar{m} \in (V^{\pi^*}(s^{\otimes}_{init}), \infty)$ and any $r_p < \infty$, there exists $\gamma^* < 1$ such that $\gamma > \gamma^*$ implies $V^{\bar{\pi}}(s^{\otimes}_{init}) > P^{M^{\otimes}}_{\bar{\pi}}(s^{\otimes}_{init} \models \varphi)\gamma^{\bar{l}} p^{\bar{l}}(s^{\otimes}_{init}, \hat{s}^{\otimes}) r_p \bar{p} \gamma^{\bar{k}'} p^{\bar{k}'}(\hat{s}^{\otimes}, \hat{s}^{\otimes}) \frac{1}{1-\gamma^{\bar{k}'}} > \bar{m}$.

The results contradict the optimality assumption of π^*. ■

IV. EXAMPLE

In this section, we evaluate our proposed method and compare it with an existing work. We consider a path planning problem of a robot in an environment consisting of eight rooms and one corridor as shown in Fig. 1. The state s_7 is the initial state and the action space is specified with $\mathcal{A}(s) = \{Right,\ Left,\ Up,\ Down\}$ for any state $s \neq s_4$ and $\mathcal{A}(s_4) = \{to\_s_0,\ to\_s_1,\ to\_s_2,\ to\_s_3,\ to\_s_5,\ to\_s_6,\ to\_s_7,\ to\_s_8\}$, where to_s_i means attempting to go to the state s_i for $i \in \{0,\ 1,\ 2,\ 3,\ 5,\ 6,\ 7,\ 8\}$. The robot moves in the intended direction with probability 0.9 and it stays in the same state with probability 0.1 if it is in the state s_4. In the states other than s_4, it moves in the intended direction with probability 0.9 and it moves in the opposite direction to that it intended to go with probability 0.1. If the robot tries to go to outside the environment, it stays in the same state. The labeling function is as follows.

$$L((s,a,s')) = \begin{cases} \{c\} & \text{if } s' = s_i,\ i \in \{2,3,5,6\}, \\ \{a\} & \text{if } (s,a,s') = (s_4, to\_s_0, s_0), \\ \{b\} & \text{if } (s,a,s') = (s_4, to\_s_8, s_8), \\ \emptyset & \text{otherwise.} \end{cases}$$

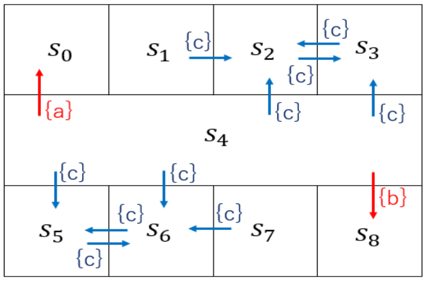
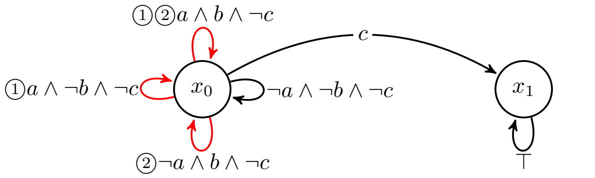
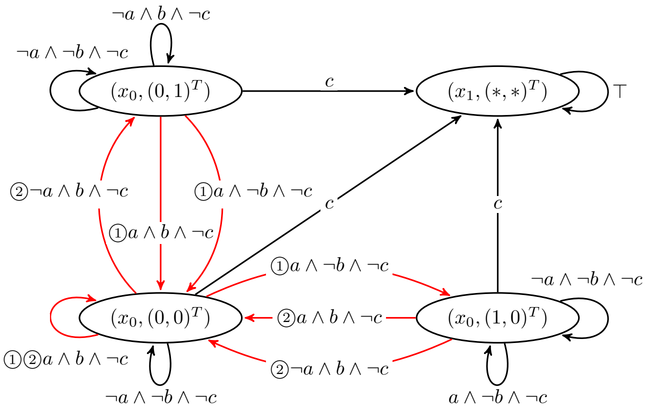


Fig. 1. The environment consisting of eight rooms and one corridor. Red arcs are the transitions that we want to occur infinitely often, while blue arcs are the transitions that we never want to occur. s_7 is the initial state.



x_0
x_1
①②$a \wedge b \wedge \neg c$
①$a \wedge \neg b \wedge \neg c$
$\neg a \wedge \neg b \wedge \neg c$
②$\neg a \wedge b \wedge \neg c$
c
$\top$

Fig. 2. The tLDBA recognizing the LTL formula $\mathbf{GF}a \wedge \mathbf{GF}b \wedge \mathbf{G}\neg c$, where the initial state is x_0. Red arcs are accepting transitions that are numbered in accordance with the accepting sets they belong to, e.g., ①$a \wedge \neg b \wedge \neg c$ means the transition labeled by it belongs to the accepting set F_1.



$\neg a \wedge b \wedge \neg c$
$\neg a \wedge \neg b \wedge \neg c$
$(x_0, (0,1)^T)$
c
$(x_1, (*,*)^T)$
$\top$
②$\neg a \wedge b \wedge \neg c$
①$a \wedge \neg b \wedge \neg c$
c
c
①$a \wedge b \wedge \neg c$
①$a \wedge \neg b \wedge \neg c$
$\neg a \wedge \neg b \wedge \neg c$
$(x_0, (0,0)^T)$
②$a \wedge b \wedge \neg c$
$(x_0, (1,0)^T)$
①②$a \wedge b \wedge \neg c$
②$\neg a \wedge b \wedge \neg c$
$\neg a \wedge \neg b \wedge \neg c$
$a \wedge \neg b \wedge \neg c$

Fig. 3. The augmented automaton for the tLDBA in Fig. 2 recognizing the LTL formula $\mathbf{GF}a \wedge \mathbf{GF}b \wedge \mathbf{G}\neg c$, where the initial state is $(x_0, (0,0)^T)$. Red arcs are accepting transitions that are numbered in accordance with the accepting sets they belong to. All states corresponding to x_1 are merged into $(x_1, (*,*)^T)$.

In the example, the robot tries to take two transitions that we want to occur infinitely often, represented by arcs labeled by $\{a\}$ and $\{b\}$, while avoiding unsafe transitions represented by the arcs labeled by $\{c\}$. This is formally specified by the following LTL formula.

$$\varphi = \mathbf{GF}a \wedge \mathbf{GF}b \wedge \mathbf{G}\neg c.$$

The above LTL formula requires the robot to keep on entering the two rooms s_0 and s_8 from the corridor s_4 regardless of the order of entries, while avoiding entering the four rooms s_2, s_3, s_5, and s_6.

We use Owl [22] to obtain the tLDBA corresponding to the LTL formula. The tLDBA $B_{\varphi} = (X, x_{init}, \Sigma, \delta, \mathcal{F})$ and its augmented automaton $\bar{B}_{\varphi} = (\bar{X}, \bar{x}_{init}, \bar{\Sigma}, \bar{\delta}, \bar{\mathcal{F}})$ are shown in Figs. 2 and 3, respectively. Specifically, the acceptance condition $\mathcal{F}$ of the tLDBA is given by $\mathcal{F} = \{F_1, F_2\}$, where $F_1 = \{(x_0, \{a\}, x_0),\ (x_0, \{a,b\}, x_0)\}$ and $F_2 = \{(x_0, \{b\}, x_0),\ (x_0, \{a,b\}, x_0)\}$.

We use Q-learning[1] with ε-greedy policy and gradually reduce ε to 0 to learn an optimal policy asymptotically. We set the positive reward $r_p = 2$, the epsilon greedy parameter $\varepsilon = \frac{0.95}{n_t(s^{\otimes})}$, where $n_t(s^{\otimes})$ is the number of visits to state $s^{\otimes}$ within t time steps [21], and the discount factor $\gamma = 0.9$. The

[1] We employ Q-learning here but any algorithm that maximizes the discounted expected reward can be applied to our proposed method.

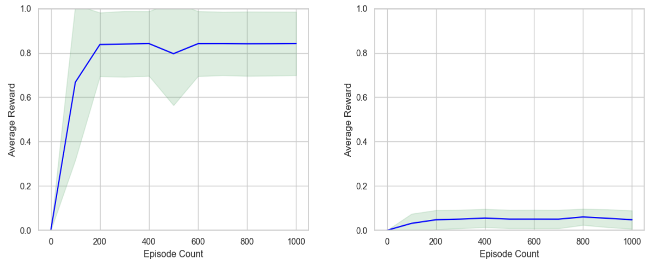


Fig. 4. The arithmetic mean of average reward in each episode for 20 learning sessions obtained from our proposed method (left) and the method by Hasanbeig $et\ al.$ [14] (right). They are plotted per 100 episodes and the green areas represent the range of standard deviations.

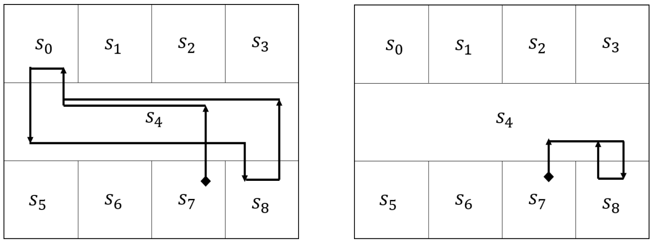


Fig. 5. The optimal policy obtained from our proposed method (left) and the method by Hasanbeig $et\ al.$ [14] (right).

learning rate α varies in accordance with *the Robbins-Monro condition*.

We also evaluate the method by Hasanbeig $et\ al.$ [14] with the same example. They use state-based LDBAs for LTL formulas and construct the product MDP of an MDP and a state-based LDBA to synthesize a policy satisfying the LTL formula. They proposed the accepting frontier function $Acc : X \times 2^X \to 2^X$ where X is the set of states of the state-based LDBA. Under initializing a set of states $\mathbb{F}$ with the union of the all accepting sets of the state-based LDBA, the function receives the state x after each transition and the set $\mathbb{F}$. If x is in $\mathbb{F}$, then Acc removes the accepting sets containing x from $\mathbb{F}$. The reward function is based on the varying set $\mathbb{F}$. We conduct the same example with their method using the tLDBA instead.

Figs. 4 and 5 show the average reward and the optimal policy, respectively, as a result of the learning when using our proposed method and the method in [14] after 10000 iterations and 1000 episodes. The arithmetic mean of average reward in each episode for 20 learning sessions is displayed per 100 episodes in Fig. 4.

The results suggest that our proposed method can synthesize a policy satisfying φ on the MDP, while the method in [14] cannot. This is because it is impossible that the transitions labeled by $\{a\}$ and $\{b\}$ occur from s_4 infinitely often by any positional policy with the tLDBA. In detail, the state of the tLDBA is always x_0 while the agent does not move to states s_2, s_3, s_5, and s_6. Thus, the state of the product MDP is always (s_4, x_0) while the agent stays in s_4. Therefore, the method in [14] may not synthesize policies satisfying LTL specifications depending on the setting of MDPs or LTL specifications.

V. Conclusions

The letter proposed a novel RL-based method for the synthesis of a control policy for an LTL specification using a limit-deterministic Büchi automaton. The proposed method improved the learning performance compared to an existing method. It is future work to extend the method to the synthesis of a hierarchical control policy.

References

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

[2] E. M. Clarke, Jr., O. Grumberg, D. Kroening, D. Peled, and H. Veith, *Model Checking*, 2nd Edition. MIT Press, 2018.

[3] M. Kloetzer, C. Belta, “A fully automated framework for control of linear systems from temporal logic specifications,” *IEEE Trans. Autom. Contr.*, vol. 53, no. 1, pp. 287–297, 2008.

[4] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, “Temporal-logic-based reactive mission and motion planning,” *IEEE Trans. Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[5] T. Wongpiromsarn, U. Topcu, and R. M. Murray, “Receding horizon temporal logic planning,” *IEEE Trans. Autom. Contr.*, vol. 57, no. 11, pp. 2817–2830, 2012.

[6] A. Sakakibara and T. Ushio, “Decentralized supervision and coordination of concurrent discrete event systems under LTL constraints,” in *Proc. 14th International Workshop on Discrete Event Systems*, 2018, pp. 18-23.

[7] C. Belta, B. Yordanov, and E. A. Gol, *Formal Methods for Discrete-Time Dynamical Systems*. Springer, 2017.

[8] M. L. Puterman, *Markov Decison Processes, Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[9] E. M. Wolff, U. Topcu, and R. M. Murray, “Robust control of uncertain Markov decision processes with temporal logic specifications,” in *Proc. 51st IEEE Conference on Decision and Control*, 2012, pp. 3372–3379.

[10] D. Sadigh, E. S. Kim, A. Coogan, S. S. Sastry, and S. Seshia, “A learning based approach to control synthesis of Markov decision processes for linear temporal logic specifications,” *in Proc. 53rd IEEE Conference on Decision and Control*, pp. 1091-1096, 2014.

[11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd Edition. MIT Press, 2018.

[12] M. Hiromoto and T. Ushio, “Learning an optimal control policy for a Markov decision process under linear temporal logic specifications,” in *Proc. 2015 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, 2015, pp. 548-555.

[13] S. Sickert, J. Esparaza, S. Jaax, and J. Křetı̀nský, “Limit-deterministic Büchi automata for linear temporal logic,” in *International Conference on Computer Aided Verification*, 2016, pp. 312-332.

[14] M. Hasanbeig, A. Abate, and D. Kroening, “Logically-constrained reinforcement learning,” *arXiv:1801.08099v8*, Feb. 2019.

[15] E. M. Hahn, M. Perez, S. Schewe, F. Somenzi, A. Triverdi, and D. Wojtczak, “Omega-regular objective in model-free reinforcement learning,” *Lecture Notes in Computer Science*, no. 11427, pp. 395–412, 2019.

[16] M. Hasanbeig, Y. Kantaros, A. Abate, D. Kroening, G. J. Pappas, and I. Lee, “Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantee,” *arXiv:1909.05304v1*, 2019.

[17] A. K. Bozkurt, Y. Wang, M. Zavlanos, and M. Pajic, “Control synthesis from linear temporal logic specifications using model-free reinforcement learning,” *arXiv:1909.07299*, 2019.

[18] R. Durrett, *Essentials of Stochastic Processes*, 2nd Edition. ser. Springer texts in statistics. New York; London; Springer, 2012.

[19] L. Breuer, “Introduction to Stochastic Processes,” [Online]. Available: https://www.kent.ac.uk/smsas/personal/lb209/files/sp07.pdf

[20] S.M. Ross, *Stochastic Processes*, 2nd Edition. University of California, Wiley, 1995.

[21] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesv́ari, “Convergence results for single-step on-policy reinforcement learning algorithms” *Machine Learning*, vol. 38, no. 3, pp, 287–308, 1998.

[22] J. Kretínský, T. Meggendorfer, S. Sickert, “Owl: A library for ω-words, automata, and LTL,” in *Proc. 16th International Symposium on Automated Technology for Verification and Analysis*, 2018, pp. 543–550.