2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Convergence Guarantee of Dynamic Programming for LTL Surrogate Reward

Zetong Xuan and Yu Wang

Abstract— **Linear Temporal Logic (LTL) is a formal way of specifying complex objectives for planning problems modeled as Markov Decision Processes (MDPs). The planning problem aims to find the optimal policy that maximizes the satisfaction probability of the LTL objective. One way to solve the planning problem is to use the surrogate reward with two discount factors and dynamic programming, which bypasses the graph analysis used in traditional model-checking. The surrogate reward is designed such that its value function represents the satisfaction probability. However, in some cases where one of the discount factors is set to 1 for higher accuracy, the computation of the value function using dynamic programming is not guaranteed. This work shows that a multi-step contraction always exists during dynamic programming updates, guaranteeing that the approximate value function will converge exponentially to the true value function. Thus, the computation of satisfaction probability is guaranteed.**

I. Introduction

Modern autonomous systems need to solve planning problems for complex rule-based tasks that are usually expressible by linear temporal logic (LTL) [1]. LTL is a symbolic language to describe high-level tasks like reaching a sequence of goals or ordering a set of events. The planning problem with the LTL objective is to find the optimal policy that maximizes the probability of satisfying the given LTL objective. This problem can be formally treated as a quantitative model-checking problem [2] when the environment is modeled as MDPs with known transition probability. That is, given an MDP and an LTL objective, find the maximum satisfaction probability within all possible policies.

Although an LTL objective can be complex, its maximal satisfaction probability and optimal policy can be computed by quantitative (or probabilistic) model checking [2]–[4] via reachability. First, a specific set of states is selected so the reachability probability to this set equals the maximum satisfaction probability of the LTL objective. The set of states shall be identified by graph analysis, such as a depth-first search, on the product MDP. The product MDP is a product of the original MDP and an ω-regular automaton, encoding all the information necessary for quantitative model-checking. Then, dynamic programming or linear programming shall be applied to compute the reachability probability of the set.

An alternative approach finds the optimal policy via a surrogate reward function without the graph analysis, thus is more generalizable to model-free reinforcement learning (RL) [5]–[8]. The surrogate reward is a reward function automatically derived from the LTL objective, which yields a value function representing the satisfaction probability of the LTL objective. The satisfaction probability can be computed using dynamic programming. Dynamic programming iteratively updates an approximate value function by the Bellman equation. Ideally, the approximate value function will converge to the value function during updates. Furthermore, the finding of the optimal policy and the corresponding maximum satisfaction probability can be done by policy or value iteration. Another advantage of using surrogate reward can be a smooth transformation into model-free RL to deal with situations when the MDP transitions are unknown. In this case, the two-phase model-checking approach has to be transformed into model-based RL and requires additional computation [9].

This work focuses on a widely used surrogate reward [8] based on the limit-deterministic Büchi automaton. It assigns a constant reward for "good" states with two discount factors. Given a policy, the probability of infinitely often visiting the "good" states shall equal the satisfaction probability of the LTL objective. The surrogate reward yields a value function equal to the satisfaction probability when taking the limit of both discount factors to one.

Nevertheless, whether dynamic programming based on this surrogate reward can find the satisfaction probability has still not been fully studied. We noticed that more recent works [10]–[13] allow one discount factor to equal 1 while using the surrogate reward with two discount factors from [8]. Their Bellman equations yield multiple solutions, as the discount factor of 1 holds in many states. The approximate value function updated by these Bellman equations may not converge to the value function. [14] proposes a sufficient condition to identify the value function from multiple solutions satisfying the Bellman equation. However, when discounting is missing in many states, whether using dynamic programming or RL based on the Bellman equation shall give us the value function is still not fully answered.

This work gives a convergence guarantee for the approximate value function during dynamic programming updates. We find an upper bound that decays exponentially for the infinite norm of the approximation error. Even though the one-step dynamic programming update does not provide a contraction on the approximation error, we show that a multi-step contraction always happens as we keep doing the dynamic programming update for enough steps. We verify our result in the case study where our upper bound holds and approximation error goes to zero. The intuition behind our

Zetong Xuan is with Department of Mechanical & Aerospace Engineering, University of Florida, Gainesville, FL 32611, USA

Yu Wang is with Department of Mechanical & Aerospace Engineering, University of Florida, Gainesville, FL 32611, USA

result is that although only a few states have discounting, the discount works on them shall be propagated to other states via the reachability between states.

II. Preliminaries

This section introduces how an LTL objective can be translated to a surrogate reward function. First, we formulate the LTL planning problem into an MDP with a Büchi objective, which is a standard approach used by probabilistic model-checking [2], [15] and other LTL planning works (e.g., [8]). Then, we show that the satisfaction probability of the Büchi objective can be expressed in another form, i.e. the value function of the surrogate reward.

A. Modeling planning problems with LTL objectives as MDPs with Büchi objectives

We propose our model called *Markov decision processes with Büchi objective*. It augments general MDPs [2] with a set of accepting states.

Definition 1: A Markov decision process with Büchi objective is a tuple $\mathcal{M} = (S, A, P, s_{\text{init}}, B)$ where

- S is a finite set of states and $s_{\text{init}} \in S$ is the initial state,
- A is a finite set of actions where $A(s)$ denotes the set of allowed actions in the state $s \in S$,
- $P : S \times A \times S \to [0, 1]$ is the transition probability function such that for all $s \in S$, we have

$$\sum_{s' \in S} P(s, a, s') = \begin{cases} 1, & a \in A(s) \\ 0, & a \notin A(s) \end{cases},$$

- $B \subseteq S$ is the set of accepting states. The set of rejecting states $\neg B := S \backslash B$.

A path of the MDP $\mathcal{M}$ is an infinite state sequence $\sigma = s_0 s_1 s_2 \cdots$ such that for all $i \geq 0$, there exists $a_i \in A(s)$ and $s_i, s_{i+1} \in S$ with $P(s_i, a_i, s_{i+1}) > 0$. Given a path σ, the ith state is denoted by $\sigma[i] = s_i$. We denote the prefix by $\sigma[:i] = s_0 s_1 \cdots s_i$ and suffix by $\sigma[i{+}1:] = s_{i+1} s_{i+2} \cdots$. We say a path σ satisfies the Büchi objective φ_B if $\inf(\sigma) \cap B \neq \emptyset$. Here, $\inf(\sigma)$ denotes the set of states visited infinitely many times on σ.

Our model is valid as the LTL objective can be translated into a Büchi objective. The translation is done by constructing a product MDP from the original MDP and a limit-deterministic Büchi automaton [15] generated from the LTL objective. Instead of looking for the optimal memory-dependent policy for the LTL objective, one can find the optimal memoryless policy on the product MDP [15]. Since the maximum satisfaction probability of Büchi objective can be achieved by a memoryless deterministic policy.

B. Policy evaluation for Büchi objective via reachability

We change an LTL planning problem into seeking the policy that maximizes the satisfaction probability of the Büchi objective. Evaluation of the policy requires the calculation of the satisfaction probability, which can be done by calculating the reachability probability.

Definition 2: A memoryless policy π is a function $\pi : S \to A$ such that $\pi(\sigma[n]) \in A(\sigma[n])$. Given an MDP $\mathcal{M} = (S, A, P, s_{init}, B)$ and a memoryless policy π, a Markov chain (MC) induced by policy π is a tuple $\mathcal{M}_\pi = (S, P_\pi, s_{init}, B)$ where $P_\pi(s, s') = P(s, \pi(s), s')$ for all $s, s' \in S$.

Under the policy π, for a path σ starting at state s, its satisfaction probability of the Büchi objective φ_B is defined as

$$\mathbb{P}_\pi(s, B) := Pr_{\sigma \sim \mathcal{M}_\pi}\big(\inf(\sigma) \cap B \neq \emptyset \mid \exists t : \sigma[t] = s\big). \tag{1}$$

The calculation of the satisfaction probability can be done by calculating the reachability probability. The probability of a path under the policy π satisfying the Büchi objective equals the probability of a path entering the accepting bottom strongly connected component (BSCC) of the induced MC $\mathcal{M}_\pi$.

Definition 3: A bottom strongly connected component (BSCC) of an MC is a strongly connected component without outgoing transitions. A strongly connected component of an MC is a communicating class, which is a maximal set of states that communicate with each other. A BSCC is rejecting[1] if all states $s \notin B$. Otherwise, we call it an accepting BSCC.

Any path on the MC will eventually enter a BSCC and stay there. Any path entering an accepting BSCC will visit accepting states infinitely many times, therefore satisfying the Büchi objective.

The calculation of the reachability probability can be done via dynamic programming after one detects all the BSCCs. This approach is adopted by probabilistic model-checking and requires complete model knowledge.

C. Surrogate reward that approximates the satisfaction probability

A surrogate reward allows one to calculate the satisfaction probability even without knowledge of BSCCs. This equivalent representation of the Büchi objective allows one to transfer an LTL objective as a discounted reward.

We study the surrogate reward for Büchi objective proposed in [8], which is also widely used in [10]–[13], [16]. This surrogate reward is designed in a way that its value function approximates the satisfaction probability. It consists of a reward function $R : S \to \mathbb{R}$ and a state-dependent discount factor function $\Gamma : S \to (0, 1]$ with two discount factors $0 < \gamma_B < \gamma \leq 1$,

$$R(s) := \begin{cases} 1 - \gamma_B & s \in B \\ 0 & s \notin B \end{cases}, \quad \Gamma(s) := \begin{cases} \gamma_B & s \in B \\ \gamma & s \notin B \end{cases}. \tag{2}$$

A positive reward is collected only when an accepting state is visited along the path. For this surrogate reward, the K-step return ($K \in \mathbb{N}$ or $K = \infty$) of a path from time $t \in \mathbb{N}$

[1]Here we call a state $s \in B$ as an accepting state, a state $s \notin B$ as a rejecting state. Notice that an accepting state must not exist in a rejecting BSCC, and a rejecting state may exist in an accepting BSCC.

is

$$G_{t:K}(\sigma) := \sum_{i=0}^{K} R(\sigma[t+i]) \cdot \prod_{j=0}^{i-1} \Gamma(\sigma[t+j])$$
$$G_t(\sigma) := \lim_{K\to\infty} G_{t:K}(\sigma). \tag{3}$$

Here, the definition follows a standard discounted reward setting [17] but allows state-dependent discounting. Suppose the discount factor $\gamma = 1$. If a path satisfies the Büchi objective, its return shall be a summation of a geometric series as $\sum_{i=0}^{\infty}(1-\gamma_B)\gamma_B^i = \frac{1-\gamma_B}{1-\gamma_B} = 1$.

The value function $V_\pi(s)$ is the expected return conditional on the path starting at s under the policy π. And it is related to Büchi objective as follows,

$$\begin{aligned} &V_\pi(s) = \mathbb{E}_\pi[G_t(\sigma) \,|\, \sigma[t] = s] \\ &= \mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t] = s, \inf(\sigma) \cap B \neq \emptyset] \cdot \mathbb{P}_\pi(s, B) \\ &+ \mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t] = s, \inf(\sigma) \cap B = \emptyset] \cdot (1 - \mathbb{P}_\pi(s, B)), \end{aligned} \tag{4}$$

where $1 - \mathbb{P}_\pi(s, B)$ stands for the probability of a path not satisfying the Büchi objective conditional on the path starting at s. As γ_B, γ close to 1, the value function becomes close to $\pi(s, B)$ as

$$\begin{aligned} &\lim_{\gamma\to 1^-} \mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t] = s, \inf(\sigma) \cap B \neq \emptyset] = 1 \\ &\lim_{\gamma_B\to 1^-} \mathbb{E}_\pi[G_t(\sigma) \mid \sigma[t] = s, \inf(\sigma) \cap B = \emptyset] = 0. \end{aligned} \tag{5}$$

The setting γ_B and γ is critical for solving an discounted reward problem using the value iteration, or Q-learning. These methods are guaranteed to find the optimal policy for the surrogate reward when $\gamma < 1$. To make sure the optimal policy for the surrogate reward is the optimal policy for the LTL objective, one has to take γ, γ_B as close to 1 as possible to reduce the error between the value function and the satisfaction probability. For γ_B, one can never take $\gamma_B = 1$ as $1 - \gamma_B = 0$ can not serve as a positive reward. Setting $\gamma = 1$ seems to work in several works. However, [14] exposes setting $\gamma = 1$ would break the uniqueness of the solution of the Bellman equation, thus hindering a correct evaluation of the satisfaction probability.

Remark 1: Other surrogate rewards based on Büchi and Rabin automata have been studied but have flaws. The surrogate rewards [6] based on limit-deterministic Büchi automata assign a constant reward for "good" states with a constant discount factor. This approach is technically flawed, as demonstrated by [7]. Surrogate reward [5] based on Rabin automata assigns constant positive rewards to certain "good" states and negative rewards to "bad" states. However, this surrogate reward function is also not technically correct, as demonstrated in [18].

III. Problem Formulation and Main Result

This section introduces a key challenge when one calculates the satisfaction probability and our answer to it. Specifically, one needs to utilize the recursive formulation of the Bellman equation to solve it. Thus, two conditions (i) the Bellman equation to have a unique solution, and (ii) a discount works on every dynamic programming update are required. However, our surrogate reward breaks both conditions as it allows $\gamma = 1$. Here, we investigate how dynamic programming calculates the satisfaction probability under the state-dependent discounting.

A. Bellman equation and sufficient condition for the uniqueness of the solution

Given a policy, the value function satisfies the Bellman equation.[2] The Bellman equation is derived from the fact that the value of the current state is equal to the expectation of the current reward plus the discounted value of the next state. For the surrogate reward in the equation (2), the Bellman equation is given as follows:

$$V_\pi(s) = \begin{cases} 1 - \gamma_B + \gamma_B \sum_{s'\in S} P_\pi(s, s') V_\pi(s') & s \in B \\ \gamma \sum_{s'\in S} P_\pi(s, s') V_\pi(s) & s \notin B \end{cases}. \tag{6}$$

Previous work [10], [11], [13] allows $\gamma = 1$. However, setting $\gamma = 1$ yields multiple solutions of the Bellman equations, raising concerns about applying dynamic programming or RL. A sufficient condition to restrict the uniqueness of the solution is proposed in [14].

Lemma 1: The Bellman equation (6) has the value function as the unique solution, if and only if the solution for any state in a rejecting BSCC is zero [14].

B. Open question on the convergence of dynamic programming

Based on Lemma 1, dynamic programming is a way to compute the value function using the Bellman equation. Given an initialization of the approximate value function $U_{(0)}$ and the Bellman equation (6), we shall iteratively do the dynamic programming update as

$$U_{(k+1)} = (1 - \gamma_B) \begin{bmatrix} \mathbb{I}_m \\ \mathbb{O}_n \end{bmatrix} + \begin{bmatrix} \gamma_B I_{m\times m} & \\ & \gamma I_{n\times n} \end{bmatrix} P_\pi U_{(k)}, \tag{7}$$

where $m = |B|$ is the number of accepting states, $n = |\neg B|$ is the number of rejecting states. $\mathbb{I}$ and $\mathbb{O}$ are column vectors with all 1 and 0 elements, respectively. We expect the approximate value function $U_{(k)}$ to converge to the value function V during updates.

Suppose we initialize $U_{(0)}$ as a zero vector. The sufficient condition for the uniqueness of the solution holds for all $U_{(k)}$. The value function is the only fix-point for the above update. However, a convergence guarantee is still needed to show we can compute the satisfaction probability using the above dynamic programming update.

When $\gamma < 1$, the convergence is guaranteed by the one-step contraction shown as

$$\|U_{(k+1)} - V\|_\infty \leq \gamma \|U_{(k)} - V\|_\infty. \tag{8}$$

[2]We call $V_\pi(s) = R(s) + \Gamma(s) \sum_{s'\in S} P_\pi(s, s') V_\pi(s')$ as the Bellman equation and $V_\pi^*(s) = \max_{a\in A(s)}\{R(s) + \Gamma(s) \sum_{s'\in S} P(s, a, s') V_\pi^*(s')\}$ as the Bellman optimality equation.

As $\gamma = 1$, this contraction no longer holds, and the convergence to the value function is still an open question. This motivates us to study the following problem.

> **Problem Formulation:** For an MDP with Büchi objective $\mathcal{M}$ by Definition 1 and the surrogate reward (2), given a policy π. Starting with $U_{(0)} = \mathbb{O}$, show approximate value function $U_{(k)}$ updated by dynamic programming (7) will converge to the value function V as $k \to \infty$. And give an upper bound for the error $\|U_{(k)} - V_\pi\|_\infty$.

In the following, we assume a fixed policy π, leading us to omit the π subscript from most notation when its implication is clear from the context.

C. Overview on main result

When $\gamma = 1$, we find a *multi-step contraction* shown as

$$\|U_{(k+N)} - V\|_\infty \leq c\|U_{(k)} - V\|_\infty \tag{9}$$

exists for the dynamic programming update, where $N \in \mathbb{N}^+$ and $c \in (0, 1)$ is a constant. Even though rejecting states lacks discounting, their reachability to accepting states still provides contraction. With this finding, we claim the convergence guarantee as follows.

Theorem 1: Given an MDP $\mathcal{M}$ and the surrogate reward (2), given a policy π. Starting with $U_{(0)} = \mathbb{O}$, approximate value function $U_{(k)}$ updated by dynamic programming (7) will converge to the value function V as

- if $\gamma < 1$

$$\left\|U_{(k)} - V\right\|_\infty \leq \gamma^k \|V\|_\infty, \tag{10}$$

- if $\gamma = 1$

$$\left\|U_{(k)} - V\right\|_\infty \leq (1 - (1 - \gamma_B)\varepsilon^{n'})^{\left\lfloor \frac{k}{n'+1} \right\rfloor} \|V\|_\infty, \tag{11}$$

  where ε is the lower bound for all possible transitions, that is, for all $(s, s') \in \{(s, s') | P_\pi(s, s') > 0\}, P_\pi(s, s') \geq \varepsilon > 0$.

As the theorem claims, one can use the surrogate reward and dynamic programming to compute the satisfaction probability of the Büchi objective on the MDP. Thus, the following corollary holds.

Corollary 1: For a product MDP constructed by the MDP and the limit-deterministic Büchi automaton [15] generated from the LTL objective. Given a policy π on the product MDPs. The surrogate reward and dynamic programming can be used to compute the satisfaction probability of the LTL objective on the MDPs.

Here, we give the convergence guarantee based on the discount factor γ, γ_B and the reachability to the discounted state expressed as $\varepsilon^{n'}$. The first bound is commonly seen for dynamic programming as each update provides one-step contraction when $\gamma < 1$. The second bound relies on the multi-step contraction shown in the following section.

IV. Multi-step contraction and proof of the main result

In this section, we will formally prove the main result by exploiting the reachability from undiscounted rejecting states to discounted accepting states. First, we simplify the dynamic programming update (7) using Lemma 1. Then, we show the infinite norm of the error vector will surely be contracted when the update is applied enough times.

A. Dynamic programming for states outside rejecting BSCCs

The sufficient condition in Lemma 1 always holds as we initialize $U_{(0)} = \mathbb{O}$. Since the approximate value function on the rejecting BSCCs stays at zero. We can simplify the dynamic programming update (7) by dropping all states inside rejecting BSCCs,

$$\begin{bmatrix} U^B \\ U^{\neg B_{T,A}} \end{bmatrix}_{(k+1)} = (1 - \gamma_B) \begin{bmatrix} \mathbb{I}_m \\ \mathbb{O}_{n'} \end{bmatrix} + H \begin{bmatrix} U^B \\ U^{\neg B_{T,A}} \end{bmatrix}_{(k)}, \tag{12}$$

where

$$\begin{aligned} H &= \begin{bmatrix} \gamma_B I_{m\times m} & \\ & \gamma I_{n'\times n'} \end{bmatrix} \underbrace{\begin{bmatrix} P_{B\to B} & P_{B\to \neg B_{T,A}} \\ P_{\neg B_{T,A}\to B} & P_{\neg B_{T,A}\to \neg B_{T,A}} \end{bmatrix}}_{T} \\ &= \begin{bmatrix} \gamma_B P_{B\to B} & \gamma_B P_{B\to \neg B_{T,A}} \\ \gamma P_{\neg B_{T,A}\to B} & \gamma P_{\neg B_{T,A}\to \neg B_{T,A}} \end{bmatrix}. \end{aligned} \tag{13}$$

Here, the state space is partitioned into three sets of states B, $\neg B_R$, $\neg B_{T,A}$ representing the set of accepting states, the set of states in the rejecting BSCCs and the set of remaining rejecting states. $U_\pi^B \in \mathbb{R}^m$, $U_\pi^{\neg B_{T,A}} \in \mathbb{R}^{n'}$ are the vectors listing the approximate value function for all $s \in B$, $s \in \neg B_{T,A}$, respectively. Matrix T represents the transition between states in $X := \{B, \neg B_{T,A}\}$. Sub-matrix $P_{B\to B}$, $P_{B\to \neg B_{T,A}}$ etc. contains the transition probability from a set of states to a set of states.

At each dynamic programming update, the approximation error is updated by a linear mapping as

$$D_{(k+1)} = HD_{(k)}. \tag{14}$$

where

$$D_{(k)} = \begin{bmatrix} U^B \\ U^{\neg B_{T,A}} \end{bmatrix}_{(k)} - \begin{bmatrix} V^B \\ V^{\neg B_{T,A}} \end{bmatrix}. \tag{15}$$

$V_\pi^B \in \mathbb{R}^m$, $V_\pi^{\neg B_{T,A}} \in \mathbb{R}^{n'}$ are the vectors listing the value function for states in B and $\neg B_{T,A}$, respectively. When setting $\gamma = 1$, the requirement for the one-step contraction (8), which is $\|H\|_\infty < 1$ does not hold.

B. Multi-step contraction

Even with $\gamma = 1$, we can still show there exists a $N \in \mathbb{N}^+$ such that $\|H^N\|_\infty < 1$. The multi-step update is given as

$$D_{(k+N)} = H^N D_{(k)}. \tag{16}$$

By showing each row sum of H^N is strictly less than that of T^N, we guarantee $\|H^N\|_\infty < 1$. In T^N, each element $\{T^N\}_{ij}$ is the probability of a path starting at i visiting j in

the next N steps, thus $\|T^N\|_\infty \leq 1$. In H^N, each element $\{H^N\}_{ij}$ is expressed as a "discounted" version of $\{T^N\}_{ij}$,

$$\{H^N\}_{ij} = \sum_{s_1 \in X} P_{\pi,\gamma_B}(i, s_1) \sum_{s_2 \in X} P_{\pi,\gamma_B}(s_1, s_2) \cdots \sum_{s_{N-1} \in X} P_{\pi,\gamma_B}(s_{N-2}, s_{N-1}) P_{\pi,\gamma_B}(s_{N-1}, j), \quad (17)$$

where

$$P_{\pi,\gamma_B}(s, s') = \begin{cases} \gamma_B P_\pi(s, s') & s \in B \\ P_\pi(s, s') & s \in X \backslash B \end{cases}. \quad (18)$$

Whenever a one-step transition starting from the accepting state happens, a discount γ_B shall be applied in (18), making $\{H^N\}_{ij} < \{T^N\}_{ij}$.

The row sum of H^N shall be strictly less than the corresponding row sum of T^N if the probability of visiting an accepting state within the future $N-1$ steps is greater than zero. Thus, the contraction $\|H^N\|_\infty < 1$ is brought in by the reachability from rejecting states to accepting, which is formalized as the following Lemma.

Lemma 2: Starting with a state in $\neg B_{T,A}$, the probability of a path not visiting the set B in $n' := |\neg B_{T,A}|$ steps is upper bounded by $1 - \varepsilon^{n'}$.

Proof: A transition leaving $\neg B_{T,A}$ must exist since all states in $\neg B_{T,A}$ are either

1) a recurrent state inside an accepting BSCC. Any path starting from such a recurrent state will eventually meet an accepting state. Thus, at least one path leaves the set $\neg B_{T,A}$.
2) A transient state that will enter a BSCC. Any path starting from such a transient state will eventually enter an accepting BSCC or a rejecting BSCC. Either way, the transient leaving the set $\neg B_{T,A}$ will happen.

As for all state $i \in \neg B_{T,A}$, there exists at least one path that will leave $\neg B_{T,A}$ in n' steps,

$$\begin{aligned} &1 - \mathbb{P}(s_1, \cdots s_{n'} \in \neg B_{T,A} | s_0 = i) \geq \varepsilon^{n'} \\ \Rightarrow &\mathbb{P}(s_1, \cdots s_{n'} \in \neg B_{T,A}, s_{n'+1} \in S | s_0 = i) \leq 1 - \varepsilon^{n'}. \end{aligned} \quad (19)$$

The existence of such a path can be shown by constructing a path starting at i and never visiting any states in $\neg B_{T,A}$ more than once. One can use the diameter of a graph to prove this existence formally and we omit it for space considerations. ∎

By this reachability property, we show the multi-step contraction, which is technically described by the following lemma.

Lemma 3: When $\gamma = 1$, given the approximation error $D_{(k)}$ updated by equation (14) , there always exists a constant $c \in (0, 1)$ and a positive integer $N \leq |\neg B_{T,A}| + 1$ such that

$$\|D_{(k+N)}\|_\infty \leq c \|D_{(k)}\|_\infty. \quad (20)$$

Proof: For the first m rows of H^N, every path starts at an accepting state and receives discounting in the beginning. For all $i \leq m$, each element $\{H^N\}_{ij} \leq \gamma_B \{T^N\}_{ij}$, we have $\sum_{j \in X} \{H^N\}_{ij} \leq \gamma_B \sum_{j \in X} \{T^N\}_{ij} \leq \gamma_B$.

The remaining n' rows need additional treatment, as discounting may not happen. However, we can rule out the case when no discount happens by setting $N = n' + 1$.

We split the sum of each row of $T^{n'+1}$ into two parts,

$$\begin{aligned} &\sum_{j \in X} \{T^{n'+1}\}_{ij} = \mathbb{P}(s_1, \cdots, s_{n'+1} \in X | s_0 = i) \\ &= \underbrace{\eta}_{\text{no accepting states are visited in } n' \text{ steps}} \\ &+ \underbrace{\mathbb{P}(s_1, \cdots, s_{n'+1} \in X | s_0 = i) - \eta}_{\text{at least one accepting state is visited in } n' \text{ steps}} \\ &\leq 1, \end{aligned} \quad (21)$$

where η represents all the paths that will not visit accepting states in n' steps and thus will not get discounted. Suppose $\eta = 1$, the sum of the entire row of $H^{n'+1}$ shall be 1. However Lemma 2 prohibits $\eta = 1$ from happening,

$$\begin{aligned} \eta &= \mathbb{P}(s_1, \cdots s_{n'} \in \neg B_{T,A}, s_{n'+1} \in X | s_0 = i) \\ &\leq \mathbb{P}(s_1, \cdots, s_{n'} \in \neg B_{T,A}, s_{n'+1} \in S | s_0 = i) \\ &\leq 1 - \varepsilon^{n'}. \end{aligned} \quad (22)$$

The second part in (21) represents all paths that will visit the accepting state in n' steps at least once. Thus, in $\sum_{j \in X} \{H^{n'+1}\}$ which represents sum of "discounted" probability (17), the first part stays the same. Meanwhile, the second part has to be discounted at least once,

$$\begin{aligned} &\sum_{j \in X} \{H^{n'+1}\}_{ij} \\ &\leq \underbrace{\eta}_{\text{no discounting}} \\ &+ \gamma_B \underbrace{(\mathbb{P}(s_1, \cdots, s_{n'+1} \in X | s_0 = i) - \eta)}_{\text{at least one accepting state is visited in n' steps}} \\ &\leq \eta + \gamma_B (1 - \eta) \\ &\leq 1 - (1 - \gamma_B) \varepsilon^{n'}. \end{aligned} \quad (23)$$

Thus, the sum of each row of $H^{n'+1} \leq c$ where $c = 1 - (1 - \gamma_B)\varepsilon^{n'}$. And we have

$$\|D_{(k+N)}\|_\infty = \|H^N D_{(k)}\|_\infty \leq c \|D_{(k)}\|_\infty. \quad (24)$$

∎

The multi-step update always provides a multi-step contraction, so we can upper bound the convergence of approximation error.

C. Proof of Theorem 1

In the case of $\gamma = 1$, we get the N-step update of the approximation error as,

$$D_{(k+N)} = H^N D_{(k)}. \quad (25)$$

By Lemma 3, after every N update, the infinite norm of error must shrink by a constant $c < 1$,

$$\left\|D_{(nN)}\right\|_\infty \leq c^n \left\|D_{(0)}\right\|_\infty. \quad (26)$$

As $n \to \infty$, the error $\|D_{(nN)}\|_\infty \to 0$.

Meanwhile, since the sum of each row of $H \leq 1$, the error won't grow at each one-step update. Thus, we have the convergence as,

$$\left\|D_{(k)}\right\|_\infty \leq c^{\lfloor \frac{k}{N} \rfloor} \left\|D_{(0)}\right\|_\infty. \tag{27}$$

Theorem 1 also considers the case when $\gamma < 1$. We get the upper bound for the approximation error as

$$\left\|D_{(k)}\right\|_\infty \leq \gamma^k \left\|D_{(0)}\right\|_\infty. \tag{28}$$

The upper bound on $\|D_{(k)}\|_\infty$ instantly holds for $\|U_{(k)} - V\|_\infty$ as starting the approximate value function at $\mathbb{O}$ guarantees $U_{(k)}^{\neg B_R} - V^{\neg B_R} = \mathbb{O}$, where $V^{\neg B_R}$ is the vector listing the value function for all states inside a rejecting BSCCs. ■

V. Case Study

In this section, we show our upper bound holds in a three-state Markov chain shown in Fig. 1. s_b, s_c are states in $\neg B_{T,A}$ and s_a is the only accepting state. All transitions are deterministic. The discount factor setting here is $\gamma_B = 0.99$, $\gamma = 1$. The dynamic programming update is given as

$$U_{(k+1)} = \begin{bmatrix} .01 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} .99 & & \\ & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} U_{(k)}.$$

The infinite norm of the H matrix in (16) is shown as

$$\|H\|_\infty = \|H^2\|_\infty = 1, \quad \|H^3\|_\infty = 0.99$$

The approximate value function is initialized as $U_{(0)} = \mathbb{O}$, thus the approximate error is $D_{(0)} = \mathbb{I}$ and during the first three updates we have,

$$D_{(1)} = \begin{bmatrix} .99 \\ 1 \\ 1 \end{bmatrix}, \quad D_{(2)} = \begin{bmatrix} .99 \\ .99 \\ 1 \end{bmatrix}, \quad D_{(3)} = \begin{bmatrix} .99^2 \\ .99 \\ .99 \end{bmatrix}.$$

Since the discount γ_B only works on s_a, $D(s_c)$ can decrease only after the $D(s_b)$ is decreased. It takes three updates for the $\|D\|_\infty$ to decrease from 1 to 0.99.

The upper bound for estimation error provided by Theorem 1 is $\left\|U_{(k)} - V\right\|_\infty \leq (1 - (1 - \gamma_B)\varepsilon^{n'})^{\left\lfloor \frac{k}{n'+1} \right\rfloor}$ where $\varepsilon = 1$ since all transitions are deterministic and $n' = 2$ as $|\neg B_{T,A}| = 2$. Thus, our theorem yields a three-step contraction shown as

$$\left\|U_{(k)} - V\right\|_\infty \leq 0.99^{\lfloor \frac{k}{3} \rfloor}$$

which captures the fact $\|H^3\|_\infty = 0.99$. The infinite norm of error shall be contracted by γ_B after every three dynamic programming updates.

In Fig. 2, the error $\|D_{(k)}\|_\infty$ is decreased by γ_B in the first three updates, aligning our upper bound. However, the error is decreased by γ_B after every two updates when $k > 3$. The reason is that our bound on the infinite norm only considers the worst case. Where the two-step update

$$H^2 = \begin{bmatrix} .99 & 0 & 0 \\ 0 & .99 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

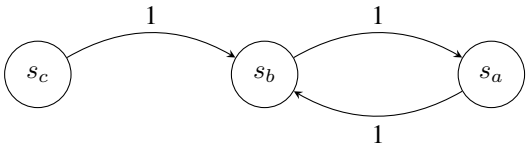

Fig. 1. Example of a three-state Markov Chain. A discount factor $\gamma_B < 1$ and a reward $1 - \gamma_B$ hold at s_a. Meanwhile, no rewards are gained at s_b and s_c. Discount γ is applied to s_b and s_c but γ can be set to 1.

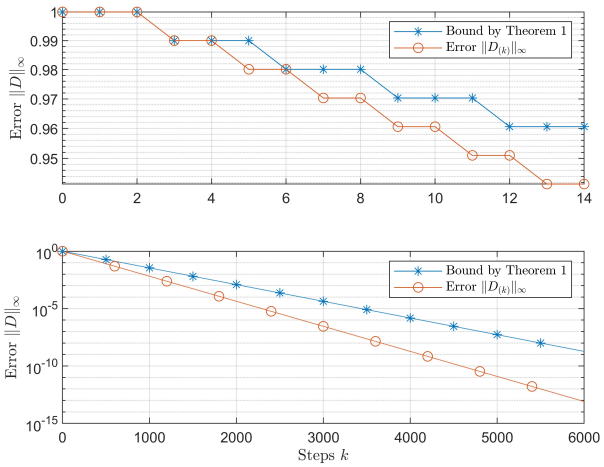

Fig. 2. Exponential convergence of estimation error $\|D_{(k)}\|$ during dynamic programming updates. Our theorem yields an upper bound that shrinks γ_B after every three dynamic programming updates. This upper bound captures the convergence in this example.

decreases $D(s_a)$, $D(s_b)$ by γ_B every two updates meanwhile making $D(s_c)_{(k+2)} = D(s_a)_{(k)}$. Since in vector $D_{(3)}$, $D(s_a)_{(3)}$ is the smallest element, thus $D_{(3)}$ serves as a good initialisation to make $\|D_{(k)}\|_\infty$ decreasing after every two updates.

VI. Conclusion and Future Work

This work answers a challenge when using surrogate reward with two discount factors for complex LTL objectives. Specifically, we can always find the value function using dynamic programming even if discounting does not hold in many states. We discuss the convergence when using dynamic programming and show that a multi-step contraction exists as we do dynamic programming updates enough times. Our findings have implications for the correct policy evaluation of LTL objectives.

Our future effect is to investigate if the convergence result still holds as we apply value iteration and Q-learning. The challenge is that once the policy is updated, it may induce a new MC, which may have different rejecting BSCCs. The sufficient condition for the uniqueness of the Bellman equation may not hold during policy updates.

References

[1] A. Pnueli, "The temporal logic of programs," in *Annual Symposium on Foundations of Computer Science*, 1977.

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

[3] G. Fainekos, H. Kress-Gazit, and G. Pappas, "Temporal Logic Motion Planning for Mobile Robots," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, 2005, pp. 2020–2025.
[4] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.
[5] D. Sadigh, E. S. Kim, S. Coogan, S. S. Sastry, and S. A. Seshia, "A learning based approach to control synthesis of Markov decision processes for linear temporal logic specifications," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 1091–1096.
[6] M. Hasanbeig, Y. Kantaros, A. Abate, D. Kroening, G. J. Pappas, and I. Lee, "Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantees," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 5338–5343.
[7] E. M. Hahn, M. Perez, S. Schewe, F. Somenzi, A. Trivedi, and D. Wojtczak, "Faithful and effective reward schemes for model-free reinforcement learning of omega-regular objectives," in *Automated Technology for Verification and Analysis: 18th International Symposium, ATVA 2020, Hanoi, Vietnam, October 19–23, 2020, Proceedings*. Springer-Verlag, 2020, pp. 108–124.
[8] A. K. Bozkurt, Y. Wang, M. M. Zavlanos, and M. Pajic, "Control Synthesis from Linear Temporal Logic Specifications using Model-Free Reinforcement Learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 10 349–10 355.
[9] P. Ashok, J. Křetínský, and M. Weininger, "PAC Statistical Model Checking for Markov Decision Processes and Stochastic Games," in *Computer Aided Verification*. Springer International Publishing, 2019, pp. 497–519.
[10] C. Voloshin, A. Verma, and Y. Yue, "Eventual Discounting Temporal Logic Counterfactual Experience Replay," in *Proceedings of the 40th International Conference on Machine Learning*. PMLR, 2023, pp. 35 137–35 150.
[11] D. Shao and M. Kwiatkowska, "Sample Efficient Model-free Reinforcement Learning from LTL Specifications with Optimality Guarantees," in *Thirty-Second International Joint Conference on Artificial Intelligence*, vol. 4, 2023, pp. 4180–4189.
[12] M. Cai, M. Hasanbeig, S. Xiao, A. Abate, and Z. Kan, "Modular deep reinforcement learning for continuous motion planning with temporal logic," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7973–7980, 2021.
[13] H. Hasanbeig, D. Kroening, and A. Abate, "Certified reinforcement learning with logic guidance," *Artificial Intelligence*, vol. 322, no. C, 2023.
[14] Z. Xuan, A. Bozkurt, M. Pajic, and Y. Wang, "On the uniqueness of solution for the Bellman equation of LTL objectives," in *Proceedings of the 6th Annual Learning for Dynamics & Control Conference*. PMLR, Jun. 2024, pp. 428–439.
[15] S. Sickert, J. Esparza, S. Jaax, and J. Křetínský, "Limit-Deterministic Büchi Automata for Linear Temporal Logic," in *Computer Aided Verification*, vol. 9780. Springer International Publishing, 2016, pp. 312–332.
[16] M. Cai, S. Xiao, J. Li, and Z. Kan, "Safe reinforcement learning under temporal logic with reward design and quantum action selection," *Scientific Reports*, vol. 13, no. 1, p. 1925, 2023.
[17] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, second edition ed. The MIT Press, 2018.
[18] E. M. Hahn, M. Perez, S. Schewe, F. Somenzi, A. Trivedi, and D. Wojtczak, "Omega-regular objectives in model-free reinforcement learning," in *Tools and Algorithms for the Construction and Analysis of Systems*. Springer International Publishing, 2019, pp. 395–412.