Translating Omega-Regular Specifications to Average Objectives for Model-Free Reinforcement Learning

Milad Kazemi
Newcastle University
UK
m.kazemi2@newcastle.ac.uk

Mateo Perez
University of Colorado Boulder
USA
Mateo.Perez@colorado.edu

Fabio Somenzi
University of Colorado Boulder
USA
fabio@colorado.edu

Sadegh Soudjani
Newcastle University
UK
sadegh.soudjani@ncl.ac.uk

Ashutosh Trivedi
University of Colorado Boulder
USA
Ashutosh.Trivedi@colorado.edu

Alvaro Velasquez
Air Force Research Laboratory
Rome, New York, USA
alvaro.velasquez.1@us.af.mil

ABSTRACT

Recent success in reinforcement learning (RL) has brought renewed attention to the design of reward functions by which agent behavior is reinforced or deterred. Manually designing reward functions is tedious and error-prone. An alternative approach is to specify a formal, unambiguous logic requirement, which is automatically translated into a reward function to be learned from. Omega-regular languages, of which Linear Temporal Logic (LTL) is a subset, are a natural choice for specifying such requirements due to their use in verification and synthesis. However, current techniques based on omega-regular languages learn in an episodic manner whereby the environment is periodically reset to an initial state during learning. In some settings, this assumption is challenging or impossible to satisfy. Instead, in the continuing setting the agent explores the environment without resets over a single lifetime. This is a more natural setting for reasoning about omega-regular specifications defined over infinite traces of agent behavior. Optimizing the average reward instead of the usual discounted reward is more natural in this case due to the infinite-horizon objective that poses challenges to the convergence of discounted RL solutions.

We restrict our attention to the omega-regular languages which correspond to *absolute liveness* specifications. These specifications cannot be invalidated by any finite prefix of agent behavior, in accordance with the spirit of a continuing problem. We propose a translation from absolute liveness omega-regular languages to an average reward objective for RL. Our reduction can be done on-the-fly, without full knowledge of the environment, thereby enabling the use of model-free RL algorithms. Additionally, we propose a reward structure that enables RL without episodic resetting in communicating MDPs, unlike previous approaches. We demonstrate empirically with various benchmarks that our proposed method of using average reward RL for continuing tasks defined by omega-regular specifications is more effective than competing approaches that leverage discounted RL.

KEYWORDS

Reinforcement Learning, Formal Synthesis, Average Reward, Omega-Regular, Linear Temporal Logic, Reward Machine, Continuing RL

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online.

ACM Reference Format:
Milad Kazemi, Mateo Perez, Fabio Somenzi, Sadegh Soudjani, Ashutosh Trivedi, and Alvaro Velasquez. 2022. Translating Omega-Regular Specifications to Average Objectives for Model-Free Reinforcement Learning. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 10 pages.

1 INTRODUCTION

The area of reinforcement learning (RL) for sequential decision-making has witnessed tremendous success in recent years. This is evidenced by RL architectures with superhuman performance in games of perception and precision such as Go [1, 2], general board games [3], and Atari [4–6], among others. In these settings, the reward signal by which agent experience is labeled for positive or negative reinforcement need only account for the current state observed by the agent and the action chosen by the same. However, it is often necessary or useful to account for the history of the agent when arbitrating the credit assignment computed by the reward function of the underlying decision process. Examples of this include learning in decision processes where rewards are sparse [7], where states are partially observable [8], or where the objective is temporally extended [9]. Moreover, it is often more natural to express the goal of the agent as the language of desirable and undesirable outcomes, with the reward signal reflecting the pursuit and avoidance, respectively, of such behaviors. The use of formal language structures to define such behavioral specifications has been well-studied in the area of formal verification and is gaining traction in specifying reward signals for RL. These specifications take the form of automata with various accepting conditions that define the language they capture. It is worth noting that there exist techniques to translate natural language objectives to their corresponding automata representations in some settings [10].

The recent development of reward machines provides a similar structured representation of the underlying reward signal and can capture non-Markovian, or history-dependent, behavior [11, 12]. These reward machines are automata whose transitions denote the reward observed by an agent for traversing from the initial node in the reward machine to some other node via a sequence of transitions that capture semantically meaningful events in the decision process. This naturally enables the definition of temporally extended objectives in RL as well as the augmentation of the underlying decision process to include observed transitions in the reward

machine, thereby transforming some non-Markovian objectives into Markovian tasks over the augmented decision process [13, 14]. Traditional off-the-shelf RL solutions can be employed for these Markovian tasks.

The field of RL [15] studies sampling-based approaches to derive decision-making policies that rely on scalar reward signals to optimize for the underlying learning objective. Samples of behavior and their associated rewards are used in a data-driven fashion to refine state or action value functions and compute policies that maximize expected cumulative reward. In *episodic* RL, the environment is periodically reset to an initial state over the course of learning. In *continuing* RL, the environment is not reset and the agent seeks to maximize its performance over its lifetime. Additionally, the environment in this setting should permit the agent to visit any state from all other states, in order to allow the agent to correct early mistakes. Such an environment is called *communicating*.

The foregoing notions of formal languages and RL have been used to great effect in the formal synthesis of control policies, which has garnered much interest in recent years [16–18]. This paradigm enables developers to focus on defining the behavioral specification of interest in some formal language as opposed to translating and implementing said specification as a reward signal or learning objective manually, which is known to be error-prone and lacking guarantees of behavior [19]. Formal synthesis algorithms leverage the underlying specification and compute a correct-by-construction policy yielding the desired behavior. In this paper, we explore such formal synthesis of policies through the use of average-reward model-free reinforcement learning (RL) [20, 21] for a class of formal specifications expressed in omega-regular languages [22]. These languages provide a rich formalism to unambiguously express desired properties of the system. These languages are accepted by automata on infinite words, where a word denotes a sequence of semantically meaningful observations observed by the agent. We introduce the notion of nondeterministic reward machines to capture reward inherent in ω-regular automata. Then, by computing a product Markov decision process (MDP) between the reward machine of an ω-regular specification and the MDP that models the agent-environment dynamics, existing continuing RL algorithms can be readily adopted to search for an optimal policy.

We focus our attention to the problem of translating omega-regular objectives to average reward for model-free RL. This is justified by challenges facing the adoption of discounted RL for continuing tasks, as discussed in the sequel. Consider the cumulative reward that is often expressed as a discounted sum of the individual rewards received by the agent at each step. The use of a discount factor ensures that the cumulative reward is bounded even for an infinite sequence of actions and rewards, thereby facilitating convergence. While mathematically convenient, discounting results in short-term rewards being valued higher than the long-run performance of the system. Thus, obtaining a suitable policy for long-run behavior depends on choosing the right discount factor, which may have to approach 1 as the size of the environment increases. However, choosing a discount factor close to 1 results in a weak contraction in RL algorithms, causing slow convergence and instability. This is exacerbated in continuing task settings, where one has to choose a very high discount factor to approximate the maximization of long-run performance. Moreover, despite the success of discounted RL for episodic tasks [5], the solution of discounted RL depends on initial state distributions, which makes it an optimization that is not compatible with function approximations in continuing settings [23]. Such function approximation is critical for learning policies on large-scale models as evidenced by the adoption of large learning models in state-of-the-art RL solutions. Thus, a natural alternative to discounting is to optimize the average reward of the agent in these settings.

However, the adoption of average reward RL faces its own set of challenges. While establishing the existence of an optimal policy for discounted RL is relatively straightforward, analyzing MDPs with the average reward objective is more difficult and requires some assumptions over the structure of the underlying MDP. Unlike discounted RL approaches, where the discount factor plays the role of the contraction parameter and enables convergence, in average reward RL algorithms the contraction factor depends on communicating assumptions of the MDP. When the communicating assumption is satisfied, there are model-free convergent average reward RL algorithms. Satisfying the communicating assumption presents a challenge to the adoption of average reward RL for the formal synthesis of policies satisfying omega-regular specifications. Indeed, the product MDP resulting from the property and the underlying MDP may not be communicating. When episodic resetting is unavailable, communication is a natural assumption. The challenge is then to ensure that this property is preserved in the product MDP. We demonstrate that this communicating property is preserved in the product MDP for an important class of omega-regular specifications by leveraging the proposed reward machines.

The main contribution of this paper is to provide an average-reward model-free RL algorithm for the design of policies that satisfy a given *absolute liveness* omega-regular specification. Our approach ensures that the communicating property is preserved in the product, enabling the learning of optimal policies, while not requiring episodic resetting. Despite the assumption of communicating MDPs, the naive synchronization of the MDP with the automaton is not generally communicating. We propose a reward machine and an augmented specification such that the communicating property of the synchronized MDP is preserved. Our work is the first to provide a translation from omega-regular objectives to average-reward RL with formal guarantees. We validate our approach with an implementation of the proposed construction and demonstrate its effectiveness on several benchmarks.

The paper is organized as follows. Section 2 includes the preliminaries and states the problem definition. Section 3 presents the main results of the paper, which establish a novel algorithm for producing optimal policies for an absolute liveness property with average reward RL. In Section 4, we test the performance of our approach on different case studies against prior techniques. Section 5 discusses related work in formal synthesis, average reward RL, and related areas. We conclude with a summary in Section 6.

2 PROBLEM DEFINITION

Markov Decision Processes. Let $\mathcal{D}(S)$ be the set of distributions over a given set S. A Markov decision process (MDP) $\mathcal{M}$ is a tuple (S, s_0, A, T, AP, L) where S is a finite set of states, $s_0 \in S$ is the initial state, A is a finite set of *actions*, $T\colon S \times A \rightharpoonup \mathcal{D}(S)$ is the

probabilistic transition function, AP is the set of *atomic propositions*, and $L\colon S \to 2^{AP}$ is the *labeling function*.

For any state $s \in S$, we let $A(s)$ denote the set of actions that can be selected in state s. An MDP is a Markov chain if $A(s)$ is singleton for all $s \in S$. For states $s, s' \in S$ and $a \in A(s)$, $T(s,a)(s')$ equals $p(s'|s,a)$. A *run* of $\mathcal{M}$ is an ω-word $\langle s_0, a_1, s_1, \ldots\rangle \in S \times (A \times S)^\omega$ such that $p(s_{i+1}|s_i, a_{i+1})>0$ for all $i \geq 0$. A finite run is a finite such sequence. For a *run* $r = \langle s_0, a_1, s_1, \ldots\rangle$ we define the corresponding labeled run as $L(r) = \langle L(s_0), L(s_1), \ldots\rangle \in (2^{AP})^\omega$. We write $Runs^{\mathcal{M}}(FRuns^{\mathcal{M}})$ for the set of runs (finite runs) of the MDP $\mathcal{M}$ and $Runs^{\mathcal{M}}(s)(FRuns^{\mathcal{M}}(s))$ for the set of runs (finite runs) of the MDP $\mathcal{M}$ starting from the state s. We write $last(r)$ for the last state of a finite run r.

A *strategy* in $\mathcal{M}$ is a function $\sigma\colon FRuns \to \mathcal{D}(A)$ such that $supp(\sigma(r)) \subseteq A(last(r))$, where $supp(d)$ denotes the support of the distribution d. A memory skeleton is a tuple $M = (M, m_0, \alpha_u)$ where M is a finite set of memory states, m_0 is the initial state, and $\alpha_u : M\times\Sigma \to M$ is the memory update function. We define the extended memory update function $\hat{\alpha}_u : M\times\Sigma^* \to M$ in a straightforward way. A finite memory strategy for $\mathcal{M}$ over a memory skeleton M is a Mealy machine (M, α_x) where $\alpha_x : S\times M \to \mathcal{D}(A)$ is the *next action function* that suggests the next action based on the MDP and memory state. The semantics of a finite memory strategy (M, α_x) is given as a strategy $\sigma : FRuns \to \mathcal{D}(A)$ such that for every $r \in FRuns$ we have that $\sigma(r) = \alpha_x(last(r), \hat{\alpha}_u(m_0, L(r)))$.

A strategy σ is *pure* if $\sigma(r)$ is a point distribution for all runs $r \in FRuns^{\mathcal{M}}$ and is *mixed* (short for strictly mixed) if $supp(\sigma(r)) = A(last(r))$ for all runs $r \in FRuns^{\mathcal{M}}$. Let $Runs^{\mathcal{M}}_\sigma(s)$ denote the subset of runs $Runs^{\mathcal{M}}(s)$ that correspond to strategy σ with initial state s. Let $\Pi_{\mathcal{M}}$ be the set of all strategies. We say that σ is *stationary* if $last(r) = last(r')$ implies $\sigma(r) = \sigma(r')$ for all runs $r, r' \in FRuns^{\mathcal{M}}$. A stationary strategy can be given as a function $\sigma : S \to \mathcal{D}(A)$. A strategy is *positional* if it is both pure and stationary.

An MDP $\mathcal{M}$ under a strategy σ results in a Markov chain $\mathcal{M}_\sigma$. If σ is a finite memory strategy, then $\mathcal{M}_\sigma$ is finite-state Markov chain. The behavior of an MDP $\mathcal{M}$ under a strategy σ and starting state $s \in S$ is defined on a probability space $(Runs^{\mathcal{M}}_\sigma(s), \mathcal{F}_{Runs^{\mathcal{M}}_\sigma(s)}, \Pr^{\mathcal{M}}_\sigma(s))$ over the set of infinite runs of σ with starting state s. Given a random variable $f\colon Runs^{\mathcal{M}} \to \mathbb{R}$, we denote by $\mathbb{E}^{\mathcal{M}}_\sigma(s)\{f\}$ the expectation of f over the runs of $\mathcal{M}$ originating at s that follow σ.

A *sub-MDP* of $\mathcal{M}$ is an MDP $\mathcal{M}' = (S', A', T', AP, L')$, where $S' \subset S$, $A' \subseteq A$ is such that $A'(s) \subseteq A(s)$ for every $s \in S'$, and T' and L' are analogous to T and L when restricted to S' and A'. Moreover $\mathcal{M}'$ is closed under probabilistic transitions. An *end-component* [24] of an MDP $\mathcal{M}$ is a sub-MDP $\mathcal{M}'$ such that for every state pair $s, s' \in S'$ there is a strategy that can reach s' from s with positive probability. A maximal end-component is an end-component that is maximal under set-inclusion. Every state s of an MDP $\mathcal{M}$ belongs to at most one maximal end-component. An MDP $\mathcal{M}$ is *communicating* if it is equal to its maximal end-component. A *bottom strongly connected component* (BSCC) of a Markov chain is any of its end-components.

Reward Machines. In the classical RL literature, the learning objective is specified using Markovian reward functions, i.e. a function $\rho : S \times A \to \mathbb{R}$ assigning utility to state-action pairs. A *rewardful* MDP is a tuple $\mathcal{M} = (S, s_0, A, T, \rho)$ where S, s_0, A, and T are defined in a similar way as for MDPa, and ρ is a Markovian reward function. A rewardful MDP $\mathcal{M}$ under a strategy σ determines a sequence of random rewards $\rho(X_{i-1}, Y_i)_{i\geq 1}$, where X_i and Y_i are the random variables denoting the i-th state and action, respectively. For $\lambda \in [0, 1[$, the *discounted reward* $Disct(\lambda)^{\mathcal{M}}_\sigma(s)$ is defined as

$$\lim_{N\to\infty} \mathbb{E}^{\mathcal{M}}_\sigma(s)\left\{\sum_{1\leq i\leq N} \lambda^{i-1}\rho(X_{i-1}, Y_i)\right\},$$

while the *average reward* $Avg^{\mathcal{M}}_\sigma(s)$ is defined as

$$\limsup_{N\to\infty} \frac{1}{N}\mathbb{E}^{\mathcal{M}}_\sigma(s)\left\{\sum_{1\leq i\leq N} \rho(X_{i-1}, Y_i)\right\}.$$

For an objective $Reward^{\mathcal{M}} \in \{Disct(\lambda)^{\mathcal{M}}, Avg^{\mathcal{M}}\}$ and state s, we define the optimal reward $Reward^{\mathcal{M}}_*(s)$ as $\sup_{\sigma\in\Pi_{\mathcal{M}}} Reward^{\mathcal{M}}_\sigma(s)$. A strategy σ is optimal for $Reward^{\mathcal{M}}$ if $Reward^{\mathcal{M}}_\sigma(s) = Reward^{\mathcal{M}}_*(s)$ for all $s \in S$. The optimal cost and strategies for these objectives can be computed in polynomial time [25].

Often, complex learning objectives cannot be expressed using Markovian reward signals. A recent trend is to express learning objectives using finite-state reward machines [11]. We require a more expressive variant of reward machine capable of ϵ transitions and nondeterminisim. We call them nondeterministic reward machines. A (nondeterministic) reward machine is a tuple $\mathcal{R} = (\Sigma_\epsilon, U, u_0, \delta_r, \rho)$ where U is a finite set of states, $u_0 \in U$ is the starting state, $\delta_r : U \times \Sigma_\epsilon \to 2^U$ is the transition relation, and $\rho : U \times \Sigma_\epsilon \times U \to \mathbb{R}$ is the reward function, where $\Sigma_\epsilon = (\Sigma \cup \{\epsilon\})$ and ϵ is a special silent transition.

Given an MDP $\mathcal{M} = (S, s_0, A, T, AP, L)$ and a reward machine $\mathcal{R} = (\Sigma_\epsilon, U, u_0, \delta_r, \rho)$ over the alphabet $\Sigma = 2^{AP}$, their product

$$\mathcal{M} \times \mathcal{R} = (S\times U, s_0\times u_0, (A\times U) \cup \{\epsilon\}, T^\times, \rho^\times)$$

is a rewardful MDP where $T^\times : (S\times U)\times((A\times U)\cup\{\epsilon\}) \to \mathcal{D}(S\times U)$ is such that $T^\times((s,u),\alpha)((s',u'))$ equals

$$\begin{cases} T(s,a)(s') & \text{if } \alpha = (a,u') \text{ and } (u, L(s), u') \in \delta_r \\ 1 & \text{if } \alpha = \epsilon \text{ and } s = s' \text{ and } \delta(u,\epsilon,u') \in \delta_r \\ 0 & \text{otherwise.} \end{cases}$$

and $\rho^\times : (S\times U) \times ((A\times U) \cup \{\epsilon\}) \times (S\times U) \to \mathbb{R}$ is defined such that $\rho^\times((s,u),\alpha,(s',u'))$ equals

$$\begin{cases} \rho(u, L(s), u') & \text{if } \alpha = (a,u') \text{ and } (u, L(s), u') \in \delta_r \\ \rho(u,\epsilon,u') & \text{if } \alpha = \epsilon. \end{cases}$$

For technical convenience, we assume that $\mathcal{M}\times\mathcal{R}$ contains only reachable states from (s_0, u_0). For both discounted and average objectives, the optimal strategies of $\mathcal{M}\times\mathcal{R}$ are positional on $\mathcal{M}\times\mathcal{R}$. Moreover, these positional strategies characterize a finite memory strategy (with memory skeleton based on the states of $\mathcal{R}$ and the next-action function based on the positional strategy) over $\mathcal{M}$ maximizing the learning objective given by $\mathcal{R}$.

Omega-Regular Specifications. Formal specification languages, such as ω-automata and logical based objectives, provide a rigorous and unambiguous mechanism to express learning objective over continuing tasks. There is a growing trend [26–30] in expressing

learning objectives in RL using linear temporal logic (LTL) and ω-regular languages (that strictly generalize LTL). We will express ω-regular languages as *good-for-MDP* Büchi automata [31].

LTL [22] is a temporal logic whose formulae describe a subset of the ω-regular languages, which is often used to specify objectives in human-readable form. Given a set of atomic propositions AP, the LTL formulae over AP can be defined via the following grammar:

$$\varphi := a \in AP \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi \,\mathsf{U}\, \varphi. \tag{1}$$

Additional operators are defined as abbreviations: $\top \stackrel{\text{def}}{=} a \vee \neg a$; $\bot \stackrel{\text{def}}{=} \neg\top$; $\varphi \wedge \psi \stackrel{\text{def}}{=} \neg(\neg\varphi \vee \neg\psi)$; $\varphi \rightarrow \psi \stackrel{\text{def}}{=} \neg\varphi \vee \psi$; $\mathsf{F}\,\varphi \stackrel{\text{def}}{=} \top \,\mathsf{U}\, \varphi$; and $\mathsf{G}\,\varphi \stackrel{\text{def}}{=} \neg\,\mathsf{F}\,\neg\varphi$. We write $w \models \varphi$ if ω-word w over 2^{AP} satisfies LTL formula φ. The satisfaction relation is defined inductively [22]. Every LTL formula can be converted [32, 33] into a Good-for-MDP Büchi automaton, defined later.

Nondeterministic Büchi automata are finite state machines capable expressing all ω-regular languages. Formally, a (nondeterministic) *Büchi automaton* is a tuple $\mathcal{A} = (\Sigma, Q, q_0, \delta, F)$, where Σ is a finite *alphabet*, Q is a finite set of *states*, $q_0 \in Q$ is the *initial state*, $\delta\colon Q \times \Sigma \to 2^Q$ is the *transition function*, and $F \subset Q \times \Sigma \times Q$ is the set of *accepting transitions*.

A *run* r of $\mathcal{A}$ on $w \in \Sigma^\omega$ is an ω-word $r_0, w_0, r_1, w_1, \ldots$ in $(Q \times \Sigma)^\omega$ such that $r_0 = q_0$ and, for $i > 0$, $r_i \in \delta(r_{i-1}, w_{i-1})$. Each triple (r_{i-1}, w_{i-1}, r_i) is a *transition* of $\mathcal{A}$. We write $\mathsf{inf}(r)$ for the set of transitions that appear infinitely often in the run r. A run r of $\mathcal{A}$ is *accepting* if $\mathsf{inf}(r) \cap F \neq \emptyset$. The *language* $\mathcal{L}(\mathcal{A})$ of $\mathcal{A}$ is the subset of words in Σ^ω that have accepting runs in $\mathcal{A}$. A language is *ω-regular* if it is accepted by a Büchi automaton.

Given an MDP $\mathcal{M}$ and an ω-regular objective φ given as an ω-automaton $\mathcal{A}_\varphi = (\Sigma, Q, q_0, \delta, F)$, we want to compute an optimal strategy satisfying the objective. We define the satisfaction probability of σ from starting state s as:

$$\mathsf{PSem}^{\mathcal{M}}_{\mathcal{A}}(s, \sigma) = \Pr^{\mathcal{M}}_{\sigma}(s)\left\{r \in \mathit{Runs}^{\mathcal{M}}_{\sigma}(s)\colon L(r) \in \mathcal{L}(\mathcal{A})\right\}.$$

The optimal satisfaction probability $\mathsf{PSem}^{\mathcal{M}}_{\mathcal{A}}(s)$ for specification $\mathcal{A}$ is defined as $\sup_{\sigma \in \Pi_{\mathcal{M}}} \Pr^{\mathcal{M}}_{\sigma}(s, \sigma)$ and we say that $\sigma \in \Pi_{\mathcal{M}}$ is an optimal strategy for $\mathcal{A}$ if $\mathsf{PSem}^{\mathcal{M}}_{\mathcal{A}}(s, \sigma)(s) = \mathsf{PSem}^{\mathcal{M}}_{\mathcal{A}}(s)$.

Given an MDP $\mathcal{M} = (S, s_0, A, T, AP, L)$ and automaton $\mathcal{A} = (2^{AP}, Q, q_0, \delta, F)$, the *product* $\mathcal{M}\times\mathcal{A} = (S\times Q, (s_0, q_0), A\times Q, T^\times, F^\times)$ is an MDP with initial state (s_0, q_0) and accepting transitions $F^\times$ where $T^\times : (S \times Q) \times (A \times Q) \rightharpoonup \mathcal{D}(S \times Q)$ is defined by

$$T^\times((s,q),(a,q'))((s',q')) = \begin{cases} T(s,a)(s') & \text{if } (q, L(s,a,s'), q') \in \delta \\ 0 & \text{otherwise.} \end{cases}$$

The final state $F^\times \subseteq (S \times Q) \times (A \times Q) \times (S \times Q)$ is defined by $((s,q),(a,q'),(s',q')) \in F^\times$ if, and only if, $(q, L(s,a,s'), q') \in F$ and $T(s,a)(s') > 0$. A strategy $\sigma^\times$ on the product defines a strategy σ on the MDP with the same value, and vice versa. Note that for a stationary $\sigma^\times$, the strategy σ may need memory. End-components and runs of the product MDP are defined just like for MDPs.

A run of $\mathcal{M}\times\mathcal{A}$ is accepting if $\mathsf{inf}(r) \cap F^\times \neq \emptyset$. We define the *syntactic satisfaction* probabilities $\mathit{PSat}^{\mathcal{M}}_{\mathcal{A}}((s,q),\sigma^\times)$ as the probability of accepting runs, i.e.

$$\Pr^{\mathcal{M}\times\mathcal{A}}_{\sigma^\times}(s,q)\left\{r \in \mathit{Runs}^{\mathcal{M}\times\mathcal{A}}_{\sigma^\times}(s,q) : \mathsf{inf}(r) \cap F^\times \neq \emptyset\right\}$$

Similarly, we define $\mathit{PSat}^{\mathcal{M}}_{\mathcal{A}}(s)$ as the optimal probability over the product, i.e. $\sup_{\sigma^\times}\left(\mathit{PSat}^{\mathcal{M}}_{\mathcal{A}}((s,q_0),\sigma^\times)\right)$. For a deterministic $\mathcal{A}$ the equality $\mathit{PSat}^{\mathcal{M}}_{\mathcal{A}}(s) = \mathsf{PSem}^{\mathcal{M}}_{\mathcal{A}}(s)$ holds; however it is not guaranteed for nondeterministic Büchi automata as the optimal resolution of nondeterministic choices may require access to future events. This motivates for the definition of a good-for-MDP nondeterminisitc Büchi automata. A Büchi automaton $\mathcal{A}$ is *good for MDPs* (GFM), if $\mathit{PSat}^{\mathcal{M}}_{\mathcal{A}}(s_0) = \mathsf{PSem}^{\mathcal{M}}_{\mathcal{A}}(s_0)$ holds for all MDPs $\mathcal{M}$ and starting states s_0 [31]. Note that every ω-regular objective can be expressed as a GFM automaton [31]. A popular class of GFM automata is suitable limit-deterministic Büchi automata [32, 34]. This paper considers only GFM Büchi automata.

The satisfaction of an ω-regular objective given as a GFM automaton $\mathcal{A}$ by an MDP $\mathcal{M}$ can be formulated in terms of the accepting maximal end-components of the product $\mathcal{M}\times\mathcal{A}$, i.e. the maximal end-component that contains an accepting transition from $\mathbf{F}^\times$. The optimal satisfaction probabilities and strategies can be computed by computing the accepting maximal end-component of $\mathcal{M} \times \mathcal{A}$ and then maximizing the probability to reach states in such components. The optimal strategies are positional on $\mathcal{M} \times \mathcal{A}$ and characterize a finite memory strategy over $\mathcal{M}$ maximizing satisfaction probability of the learning objective given by $\mathcal{A}$.

Reinforcement Learning. Given an MDP $\mathcal{M}$, reward machine $\mathcal{R}$, and an optimization objective (discounted or average reward), an optimal strategy can be computed in polynomial time using linear programming [25]. Similarly, graph-theoretic techniques to find maximal end-components can be combined with linear programming to compute optimal strategies for ω-regular objectives [34]. However, when the transition/reward structure of the MDP is unknown, such techniques are not applicable.

Reinforcement learning [15] (RL) is a sampling-based optimization approach where an agent learns to optimize its strategy by repeatedly interacting with the environment relying on the reinforcements (numerical reward signals) it receives for its actions. We focus on model-free approach to RL where the learner computes optimal strategies without explicitly estimating the transition probabilities and rewards. These approaches are asymptotically space-efficient [35] than model-based RL and have been shown to scale well [5, 36]. Some prominent model-free RL algorithms for discounted and average reward objectives include Q-learning and TD(λ) [15] and Differential Q-learning [21, 37].

In some applications, such as running a maze or playing tic-tac-toe—the interaction between the agent and the environment naturally breaks into finite length learning sequences, called episodes. Thus the agent optimizes its strategy by combining its experience over different episodes. We call such tasks *episodic*. On the other hand, for some applications—such as process control and reactive systems—this interaction continues ad-infinitum and the agent lives and learns over a single lifetime. We call such tasks *continuing*.

Problem Statement and Assumptions. This paper develops a model-free RL algorithm for continuing tasks where the learning objective is given as an ω-regular objective given as a GFM automaton. Prior solutions [26–29] focused on episodic setting and have proposed a model-free reduction (does not require access to the

MDP) from ω-regular objectives to discounted-reward objectives. Recently several researchers [15, 23] made the case for adopting average reward formulation for continuing tasks due to several limitations of discounted-reward RL in continuing tasks. This paper investigates a model-free reduction from ω-regular objectives to average-reward objectives in model-free RL.

Problem 1 (ω-Regular to Average Reward Translation). *Given an unknown communicating MDP $\mathcal{M} = (S, s_0, A, T, AP, L)$ and a GFM automaton $\mathcal{A} = (\Sigma, Q, q_0, \delta, F)$, the* reward translation *problem is to design a reward machine $\mathcal{R}$ such that an optimal positional strategy maximizing the average reward for $\mathcal{M}\times\mathcal{R}$ provides a finite memory strategy maximizing the satisfaction probability of $\mathcal{A}$ in $\mathcal{M}$.*

The existing average RL algorithms such as Differential Q-learning provide convergence guarantees under the assumption that the MDP $\mathcal{M}$ is communicating [21]. Thus, for the reward translation to be effective, we need to make sure that the product $\mathcal{M} \times \mathcal{A}$ is communicating. Unfortunately, even when $\mathcal{M}$ is communicating, the product $\mathcal{M} \times \mathcal{A}$ may violate the communicating requirement.

We give a solution for the translation problem for an important class of properties called *absolute liveness* [38]. Recall that a property is absolute liveness if appending an arbitrary finite prefix to an accepting word produces an accepting word. Formally, a language $L \subseteq \Sigma^\omega$ is an *absolute liveness* property if for every $w \in L$ and $a \in \Sigma$ we have that $aw \in L$. Note that for an absolute liveness language L and for every $x \in \Sigma^*$ we have that $xw \in L$. This implies that an LTL property φ is absolute liveness property if φ is satisfiable and φ and $\mathsf{F}\,\varphi$ are expressively equivalent. For average reward objectives adding a prefix to a trace should not change the average value associated with the trace. This is aligned with the satisfaction of absolute liveness properties. Moreover, since absolute liveness properties cannot be rejected for any finite word, they preserve the continual nature of the learning procedure. To solve Problem 1, we make the following assumption.

Assumption 1. *Given an MDP $\mathcal{M}$ and ω-automaton $\mathcal{A}$, we assume that: 1) $\mathcal{M}$ is communicating; 2) $\mathcal{A}$ is a GFM automaton; and 3) $\mathcal{A}$ is an absolute liveness property.*

3 CONSTRUCTION AND CORRECTNESS

Let us fix a communicating MDP $\mathcal{M} = (S, s_0, A, T, AP, L)$ and an absolute liveness GFM property $\mathcal{A} = (\Sigma, Q, q_0, \delta, F)$ for the rest of this section. Our goal is to learn a reward machine $\mathcal{R}$ such that we can use an off-the-shelf average reward RL on $\mathcal{M} \times \mathcal{R}$ to compute an optimal strategy of $\mathcal{M}$ against $\mathcal{A}$.

Since the optimal strategies are not positional on $\mathcal{M}$ but rather positional on $\mathcal{M}\times\mathcal{A}$, it is natural to assume that the reward machine $\mathcal{R}$ takes the structure of $\mathcal{A}$ with a reward function providing positive reinforcement with every accepting transition. Unfortunately, even for absolute liveness GFM automata $\mathcal{A}$, the product $\mathcal{M}\times\mathcal{A}$ with a communicating MDP $\mathcal{M}$ may not be communicating.

Example 1. *Assume a communicating MDP M with at least one state labeled a or b, and the absolute liveness property $\varphi = \mathsf{F}(\mathsf{G}\,a \vee \mathsf{G}\,\mathsf{F}\,b)$ and its automaton shown in Fig. 1. Observe that any run that visits one of the two accepting states cannot not visit the other one. Hence, the product does not satisfy the communicating property.*

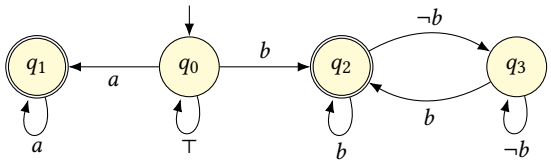


Figure 1: A Büchi automaton for $\varphi = \mathsf{F}(\mathsf{G}\,a \vee \mathsf{G}\,\mathsf{F}\,b)$

Reward Machine Construction Let $\mathcal{A} = (\Sigma, Q, q_0, \delta, F)$ be an absolute liveness GFM automaton. Consider $\mathcal{R}_\mathcal{A} = (\Sigma_\epsilon, Q, q_0, \delta', \rho)$ where $\delta'(q, a) = \delta(q, a)$ for all $a \in \Sigma$ and ϵ transitions reset to the starting state, i.e. $\delta'(q, \epsilon) = q_0$. Note that by adding the reset (ϵ) action from every state of $\mathcal{R}$ to its initial state, the graph structure of $\mathcal{M}$ is strongly connected. The reward function $\rho : Q\times\Sigma\cup\{\epsilon\}\times Q\to\mathbb{R}$ is such that

$$\rho(q, a, q') = \begin{cases} c & \text{if } a = \epsilon \\ 1 & \text{if } (q, a, q') \in F \\ 0 & \text{otherwise.} \end{cases}$$

Lemma 1 (Preservation of Communication). *For a communicating MDP $\mathcal{M}$ and reward machine $\mathcal{R}_\mathcal{A}$ for an absolute liveness GFM automaton $\mathcal{A}$, we have that the product $\mathcal{M}\times\mathcal{R}_\mathcal{A}$ is communicating.*

Proof. To show that $\mathcal{M}\times\mathcal{R}_\mathcal{A}$ is communicating, we need to show that for arbitrary states $(s, q), (s', q') \in S \times Q$ reachable from the initial state (s_0, q_0), we have that there is a strategy that can reach (s', q') from (s, q) with positive probability. Note that since $\mathcal{M}$ is communicating, it is possible to reach (s_0, q') from (s, q) for some q' of $\mathcal{R}_\mathcal{A}$ using a strategy to reach s_0 from s in $\mathcal{M}$. We can then use a reset (ε) action in $\mathcal{R}_\mathcal{A}$ to reach the state (s_0, q_0). Since (s', q') is reachable from the initial state (s_0, q_0), we have a strategy to reach (s', q') from (s, q) with positive probability. □

Lemma 2 (Average and Probability). *There exists a $c^* < 0$ such that for all $c < c^*$, positional strategies that maximize the average reward on $\mathcal{M} \times \mathcal{R}_\mathcal{A}$ will maximize the satisfaction probability of $\mathcal{A}$.*

Proof. The proof is in three parts.

(1) First observe that if $c < 0$, then for any average-reward optimal strategy in $\mathcal{M} \times \mathcal{R}_\mathcal{A}$, the expected average reward is non-negative. This is so because all other actions except ε actions provide non-negative rewards. Hence, any strategy that takes ε actions only finitely often, results in a non-negative average reward.
(2) Let Π^* be the set of positional strategies in $\mathcal{M}\times\mathcal{R}_\mathcal{A}$ such that the ε actions are taken only finitely often, i.e. no BSCC of the corresponding Markov chain contains an ε transition. Let Π^ϵ be the set of remaining positional strategies, i.e., the set of positional strategies that visit an ϵ transition infinitely often. Let $0<p_{\min}<1$ be a lower bound on the expected long-run frequency of the ϵ transitions among all strategies in Π^ϵ. Let $c_* = -1/p_{\min}$. Observe that for every policy $\sigma' \in \Pi^\epsilon$, the expected average reward is negative and cannot be an optimal strategy in $\mathcal{M} \times \mathcal{R}_\mathcal{A}$. To see that, let $0 < p \leq 1$ be the the long-run frequency of the ϵ transitions for σ and let

$0 \leq q < 1$ be the long-run frequency of visiting accepting transitions for σ. The average reward for σ is

$$\begin{aligned} Avg_{\sigma}^{\mathcal{M}\times\mathcal{R}_{\mathcal{A}}}(s_0, q_0) &= p \cdot c + q \cdot 1 + (1-p-q) \cdot 0 \\ &\leq p \cdot c + q \cdot 1 + (1-p-q) \cdot 1 \\ &= p \cdot c + (1-p) \\ &\leq p \cdot c_* + (1-p) \\ &= -p/p_{\mathtt{min}} + (1-p) \\ &\leq -1 + (1-p) \leq -p. \end{aligned}$$

Since every optimal policy must have a non-negative average reward, no policy in Π^{ϵ} is optimal for $c < c_*$.

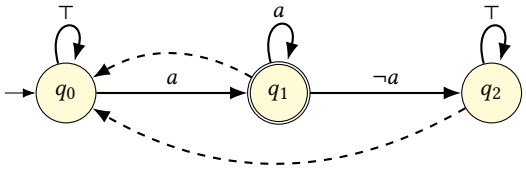
(3) Now consider an optimal policy σ^* in Π^*. We show that this policy also optimizes the probability of satisfaction of $\mathcal{A}$. There are two cases to consider.
   (a) If the expected average reward of σ^* is 0, then under no strategy it is possible to reach an accepting transitions (positive reward transition) in $\mathcal{M} \times \mathcal{R}_{\mathcal{A}}$. Hence, every policy is optimal in $\mathcal{M}$ against $\mathcal{A}$, and so is σ^*.
   (b) If the expected average reward of σ^* is positive, then notice that for every BSCC of the Markov chain of $\mathcal{M} \times \mathcal{R}_{\mathcal{A}}$ under σ_*, the average reward is the same. This is so because otherwise, there is a positional policy that reaches the BSCC with the optimal average from all the other BSCCs with lower averages, contradicting the optimality of σ_*. Since for an optimal policy σ_*, every BSCC provides the same positive average, every BSCC must contain an accepting transition. Hence, every run of the MDP $\mathcal{M}$ under σ_* will eventually dwell in an accepting component and in the process will see a finitely many ϵ (reset) transitions. For any such given run r, consider the the suffix r' of the run after the last ϵ transition is taken and let $r = wr'$ for some finite run w. Since $L(r')$ is an accepting word in $\mathcal{A}$, and since $\mathcal{A}$ is an absolute liveness property any arbitrary prefix w' to this run r' is also accepting. This implies that the original run r is also accepting for $\mathcal{A}$. It follows that for such a strategy σ_*, the probability of satisfaction of $\mathcal{A}$ is 1, making σ_* an optimal policy for $\mathcal{M}$ against $\mathcal{A}$. □

Since our translation from ω-regular objective to reward machines is model-free, the following theorem is immediate.

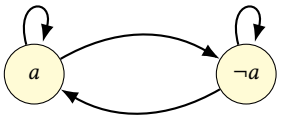
Theorem 1 (Convergence of Model-free RL). *Differential Q-learning algorithm for maximizing average reward objective on $\mathcal{M} \times \mathcal{R}_{\mathcal{A}}$ will converge to a strategy maximizing the probability of satisfaction of $\mathcal{A}$ for a suitable value of c. Moreover, the product construction $\mathcal{M} \times \mathcal{R}_{\mathcal{A}}$ can be done on-the-fly and it is model-free.*

As an example, consider the property F G a and an MDP with two states and all transitions between states are available as deterministic actions (Fig. 2). Only one of the states is labeled a. An infinite memory strategy could see a for one step, reset, then see two as, reset, then see three as and so forth. This strategy will produce the same average value as the positional strategy which sees a forever without resetting. However, the infinite memory strategy will fail the property while the positional one will not.

Shaping Rewards via Hard Resets. For a Büchi automaton $\mathcal{A}$, we say that its state $q \in Q$ is *coaccessible* if there exists a path starting from that state to a accepting transition. If a state is not coaccessible then any run of the product $\mathcal{M} \times \mathcal{A}$ that ends in such a state will never be accepting, and hence one can safely redirect all of its outgoing transitions to the initial state with reward c (a hard reset). Such hard resets will promote speedy learning by reducing the time spent in such states during unsuccessful explorations, and at the same time adding these resets does not make a non-accepting run accepting or vice versa. Lemma 1, Lemma 2, and Theorem 1 continue to hold with such hard resets. Introducing hard resets is a reward shaping procedure in that it is a reward transformation [39] under which optimal strategies remain invariant.



(a) Automaton of F G a**, dashed lines represent reset transitions**



(b) MDP, each transition represents an action

Figure 2: The two state MDP and a persistence property

4 EXPERIMENTAL RESULTS

We implemented the reduction[1] with hard resets presented in Section 3. As described, we do not build the product MDP explicitly, and instead compose it on-the-fly by keeping track of the MDP and automaton states independently. We use Differential Q-learning [21] to learn optimal, positional average reward strategies. For our experiments, we have collected a set of communicating MDPs with absolute liveness properties[2].

We compare with two previous approaches for translating omega-regular languages to rewards: the method of [26] with Q-learning and the method of [40] with Q-learning. The method of [26] translates a GFM Büchi automaton into a reachability problem through a suitable parameter ζ. This reachability problem can be solved with discounted RL by rewarding reaching the target state and using a large enough discount factor. The method of [40] uses a state dependent discount factor γ_B and a GFM Büchi automaton. By using a suitable γ_B and large enough discount factor, one can learn optimal strategies for the omega-regular objective.

RQ1. How do previous approaches perform in the continuing setting? The methods of [26, 40] may produce product MDPs that are not communicating (see Example 1). This means that a single continuing run of the MDP may not explore all relevant states and actions. Thus, previous methods are not guaranteed to converge in this setting. We studied if this behavior affects these

[1]The implementation is available at https://plv.colorado.edu/mungojerrie/.
[2]Case studies are available at https://plv.colorado.edu/mungojerrie/aamas22.

prior methods in practice. As a baseline, we include our proposed approach. Instead of tuning hyperparameters for each method, where hyperparameters that lead to convergence may not exist, we take a sampling approach. We select a wide distribution over hyperparameters for each method and sample 200 hyperparameter combinations for each method and example. We then train for 10 million steps on each combination. The selected hyperparameter distribution is $\alpha \sim \mathcal{D}(0.01, 0.5)$, $\varepsilon \sim \mathcal{D}(0.01, 1.0)$, $c \sim \mathcal{D}(1, 200)$, $\eta \sim \mathcal{D}(0.01, 0.5)$, $\zeta \sim \mathcal{D}(0.5, 0.995)$, $\gamma_B \sim \mathcal{D}(0.5, 0.995)$, and $\gamma \sim \mathcal{D}(0.99, 0.99999)$ where $\mathcal{D}(a, b)$ is a log-uniform distribution from a to b. The end points of these distributions and the training amount was selected by finding hyperparameters which led to convergence in the episodic setting for these methods.

Figure 3 shows the resulting distribution over runs. A distribution entirely at 0 (1) indicates that all sampled runs produced strategies that satisfy the property with probability 0 (1). *For many examples, prior approaches had no successful hyperparameter combinations, with distributions centered entirely at* 0. However, our proposed approach always had some hyperparameters that led to optimal, probability 1, strategies, as indicated by the tails of the distributions touching the probability 1 region of the plot.

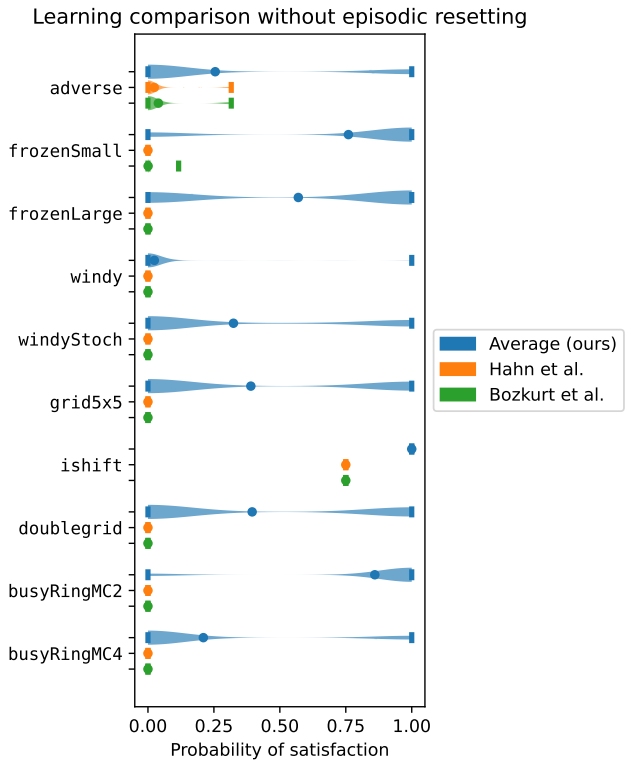


Figure 3: Comparison of the distributions of probability of satisfaction of learned policies across sampled hyperparameters in the continuing setting. For each distribution, the mean is shown as a circle, and the maximum and minimum are shown as vertical bars. We compare our proposed reduction, the reduction of [26] with Q-learning, and the reduction of [40] with Q-learning. Episodic resetting was not used.

RQ2. How does our method compare to previous approaches when we allow episodic setting? By allowing episodic resetting, we can now find hyperparameters for previous methods that lead to convergence. We tuned all hyperparameters by hand to minimize training time, while verifying with a model checker that the produced strategies are optimal. Table 1 shows learning times, as well as hyperparameters for our reduction. We report the number of states reachable in the MDP and the product, learning times averaged over 5 runs, the reset penalty c, the ε-greedy exploration rate ε, the Differential Q-learning learning rates α and η, as well as the number of training steps. Note that we do not do any episodic resetting when training with our reduction. This means that the RL agent must learn to recover from mistakes during training, while previous approaches are periodically reset to a good initial state. *Our reduction using Differential Q-learning is competitive with previous approaches while not being reliant on episodic resetting.*

5 RELATED WORK

The development and use of formal reward structures for RL have witnessed increased interest in recent years. For episodic RL, logics have been developed over finite traces of agent behavior, including LTL_f and Linear Dynamic Logic (LDL_f) [41, 42]. These logics have equivalent automaton and reward machine representations that have catalyzed a series of efforts on defining novel reward shaping functions to accelerate the convergence of RL algorithms subject to formal specifications [9, 43, 44]. These methods leverage the graph structure of the automaton to provide an artificial reward signal to the agent. More recently, dynamic reward shaping using LTL_f has been introduced as a means to both learn the transition values of a given reward machine and leverage these values for reward shaping and transfer learning [45]. There has also been work on learning or synthesizing the entire structure of such reward machines from agent interactions with the environment by leveraging techniques from satisfiability and active grammatical inference [7, 8, 13, 14, 46].

For the infinite-trace settings, LTL has been extensively used to verify properties and synthesize policies formally using the mathematical model of a system [22, 30, 47–51]. Considering the generality of the results in terms of structure of the underlying MDP, most of the research focuses on discounted reward structures. Despite the simplicity of discounted Markov decision problems, the discounted reward structure (unlike average reward) prioritizes the transient response of the system. However, application of the average reward objective because of the restriction over the structure of the MDP is limited. The work [52] proposes a policy iteration algorithm for satisfying properties of the form $\mathsf{G}\,\mathsf{F}\,\phi \wedge \psi$ for a communicating MDP almost surely. The work [53] proposes a value iteration algorithm for solving the average reward problem for multichain MDPs, where the algorithm first computes the optimal value for each of strongly connected components and then weighted reachability to find the optimal policy. The work [54] provides a linear program for policy synthesis of multichain MDPs with steady-state constraints.

In the last few years, researchers have started developing data-driven policy synthesis techniques in order to satisfy temporal properties. There is a large body of literature in safe reinforcement learning (RL) (see e.g. [55–57]). The problem of learning a policy to maximize the satisfaction probability of a temporal property using discounted RL is studied recently [27, 28, 40, 58–61]. The work [26]

Name	states	prod.	time	time†	time‡	c	ε	α	η	train-steps
adverse	202	507	8.51	7.09	12.56	-150		0.2		10M
frozenSmall	16	64	0.99	20.23	9.88					500k
frozenLarge	64	256	4.07	3.88	8.79			0.02	0.02	3M
windy	123	366	1.40	1.81	2.61		0.95	0.5	0.05	1M
windyStoch	130	390	2.97	3.91	2.53			0.5		2M
grid5x5	25	100	0.62	1.12	1.02			0.5		200k
ishift	4	29	0.03	0.01	0.02					10k
doublegrid	1296	5183	16.43	3.45	3.09	-2	0.5	0.05	0.01	12M
busyRingMC2	72	288	0.03	0.03	0.03				0.01	10k
busyRingMC4	2592	15426	6.08	3.94	2.33				0.01	1.5M

Table 1: Learning results and comparison. Hyperparameters used for our reduction are shown. Blank entries indicate that default values were used. The default parameters are $c = -1$, $\varepsilon = 0.1$, $\alpha = 0.1$, and $\eta = 0.1$. Times are in seconds. Superscript † indicates results from Q-learning with reduction from [26], while superscript ‡ indicates Q-learning with reduction from [40]. Results for † and ‡ required episodic resetting. All hyperparameters were tuned by hand.

by using a parameterized augmented MDP provides an RL-based policy synthesis for finite MDPs with unknown transition probabilities. It shows that the optimal policy obtained by RL for the reachability probability on the augmented MDP gives a policy for the MDP with a suitable convergence guarantee. In [40] authors provide a path-dependent discounting mechanism for the RL algorithm based on a limit-deterministic Büchi automaton (LDBA) representation of the underlying omega-regular property, and prove convergence of their approach on finite MDPs when the discounting factor goes to one. An LDBA is also leveraged in [28, 60, 61] for discounted-reward model-free RL in both continuous- and discrete-state MDPs. The LDBA is used to define a reward function that incentivizes the agent to visit all accepting components of the automaton. These works use episodic discounted RL with discount factor close to one to solve the policy synthesis problem. There are two issues with the foregoing approaches. First, because of the episodic nature of the algorithms they are not applicable in continuing settings. Second, because of high discount factors in practice these algorithm are difficult to converge. On the other hand, recent work on reward shaping for average reward RL has been explored based on safety properties to be satisfied by the synthesized policy [62]. In contrast to the solution proposed in this paper, the preceding approach requires knowledge of the graph structure of the underlying MDP and does not account for absolute liveness properties.

There is a rich history of studies in average reward RL [20, 63]. Lack of stopping criteria for multichain MDPs affect the generality of model-free RL algorithms. In this way, all model-free RL algorithms put some restrictions on the structure of MDP (e.g. ergodicity [64, 65] or communicating property). The closest line of work to this work is to use average reward objective for safe RL. The work [66] proposes a model-based RL algorithm for maximizing average reward objective with safety constraint for communicating MDPs. It is worth noting that in multichain setting, the state-of-the-art learning algorithms use model-based RL algorithms. The work [67] studies satisfaction of ω-regular properties using data-driven approaches. The authors introduce an algorithm where the optimality of the policy is conditioned to not leaving the corresponding maximal end component which leads to a sub-optimal solution. The authors provide PAC analysis for the algorithm as well. Despite all the efforts of using data-driven approaches for satisfying the ω-regular properties, there is a gap in using average reward model-free RL algorithms for satisfying temporal properties.

This paper is an attempt to close this gap by proposing a model-free average reward RL algorithm for a subclass of LTL properties called absolute liveness properties. We claim this subclass captures a large class of interesting properties and are suitable for average reward RL. Furthermore, the eventual satisfaction semantics of an arbitrary omega-regular or LTL specification ϕ can be captured by an absolute liveness property $\mathsf{F}\,\phi$.

6 CONCLUSION

This work addressed the problem of synthesizing policies that satisfy a given absolute liveness omega-regular property in the continuing setting. Our key contribution is a model-free translation from the omega-regular specification to an average reward objective, enabling the use of off-the-shelf average reward RL. This is in contrast to existing methods in the literature that use discounted, episodic learning, which require the ability to reset the underlying environment and is restrictive in some settings. Our approach avoids this episodic learning and learns the optimal policy in one life-long episode without resetting. Furthermore, the proposed solution does not require access to a model of the environment nor to its graph structure, thereby avoiding a common assumption made in the literature on requiring the computation of end components for synthesis of policies subject to some omega-regular specification.

For our experiments, we applied Differential Q-learning to a range of case studies and showed that the proposed approach is successful in converging to optimal strategies under the raised assumptions. In particular, our experiments showed that the proposed approach is superior to previous methods in the continuing setting. This lends credence to the important and understudied idea that average reward RL is better-suited for continuing task settings than the more popular discounted RL. For future work, we will explore the use of function approximation in the hopes that average reward RL can experience the same success for continuing tasks that its discounted RL counterpart has witnessed in episodic settings.

ACKNOWLEDGMENTS

This material is based upon work supported under the EPSRC New Investigator Award CodeCPS (EP/V043676/1) and the National Science Foundation under Grant No. (2009022).

REFERENCES

[1] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[2] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354, 2017.

[3] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[4] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[5] V. Mnih et al. Human-level control through reinforcement learning. *Nature*, 518:529–533, February 2015.

[6] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[7] Daniel Neider, Jean-Raphael Gaglione, Ivan Gavran, Ufuk Topcu, Bo Wu, and Zhe Xu. Advice-guided reinforcement learning in a non-Markovian environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[8] Rodrigo Toro Icarte, Ethan Waldie, Toryn Klassen, Rick Valenzano, Margarita Castro, and Sheila McIlraith. Learning reward machines for partially observable reinforcement learning. *Advances in Neural Information Processing Systems*, 32:15523–15534, 2019.

[9] Alberto Camacho, Rodrigo Toro Icarte, Toryn Q Klassen, Richard Anthony Valenzano, and Sheila A McIlraith. LTL and beyond: Formal languages for reward function specification in reinforcement learning. In *IJCAI*, volume 19, pages 6065–6073, 2019.

[10] Andrea Brunello, Angelo Montanari, and Mark Reynolds. Synthesis of LTL formulas from natural language texts: State of the art and research directions. In *26th International Symposium on Temporal Representation and Reasoning (TIME 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[11] Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*, pages 2107–2116. PMLR, 2018.

[12] Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Anthony Valenzano, and Sheila A. McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *CoRR*, abs/2010.03950, 2020.

[13] Maor Gaon and Ronen Brafman. Reinforcement learning with non-Markovian rewards. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34(04), pages 3980–3987, 2020.

[14] Zhe Xu, Bo Wu, Aditya Ojha, Daniel Neider, and Ufuk Topcu. Active finite reward automaton inference and reinforcement learning using queries and counterexamples. In *Machine Learning and Knowledge Extraction - 5th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2021, Virtual Event, August 17-20, 2021, Proceedings*, volume 12844 of *Lecture Notes in Computer Science*, pages 115–135. Springer, 2021.

[15] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[16] Calin Belta and Sadra Sadraddini. Formal methods for control synthesis: An optimization perspective. *Annual Review of Control, Robotics, and Autonomous Systems*, 2:115–140, 2019.

[17] Pushpak Jagtap, Sadegh Soudjani, and Majid Zamani. Formal synthesis of stochastic systems via control barrier certificates. *IEEE Transactions on Automatic Control*, 2020.

[18] Rupak Majumdar, Kaushik Mallik, Anne-Kathrin Schmuck, and Sadegh Soudjani. Symbolic qualitative control for stochastic systems via finite parity games. *IFAC-PapersOnLine*, 54(5):127–132, 2021.

[19] Hadas Kress-Gazit, Morteza Lahijanian, and Vasumathi Raman. Synthesis for robots: Guarantees and feedback for robot behavior. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:211–236, 2018.

[20] Sridhar Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine learning*, 22(1):159–195, 1996.

[21] Yi Wan, Abhishek Naik, and Richard S Sutton. Learning and planning in average-reward Markov decision processes. In *International Conference on Machine Learning*, pages 10653–10662. PMLR, 2021.

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

[23] Abhishek Naik, R. Shariff, Niko Yasui, and R. Sutton. Discounted reinforcement learning is not an optimization problem. *ArXiv*, abs/1910.02140, 2019.

[24] Luca De Alfaro. *Formal verification of probabilistic systems*. PhD thesis, Stanford University, 1998.

[25] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[26] Ernst Moritz Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. Omega-regular objectives in model-free reinforcement learning. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pages 395–412. Springer, 2019.

[27] Dorsa Sadigh, Eric S Kim, Samuel Coogan, S Shankar Sastry, and Sanjit A Seshia. A learning based approach to control synthesis of Markov decision processes for linear temporal logic specifications. In *Conference on Decision and Control*, pages 1091–1096, 2014.

[28] Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Certified reinforcement learning with logic guidance. *arXiv preprint arXiv:1902.00778*, 2019.

[29] Alper Kamil Bozkurt, Yu Wang, Michael M. Zavlanos, and Miroslav Pajic. Control synthesis from linear temporal logic specifications using model-free reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*, pages 10349–10355. IEEE, 2020.

[30] Milad Kazemi and Sadegh Soudjani. Formal policy synthesis for continuous-state systems via reinforcement learning. In *International Conference on Integrated Formal Methods*, pages 3–21. Springer, 2020.

[31] Ernst Moritz Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. Good-for-mdps automata for probabilistic analysis and reinforcement learning. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 306–323. Springer, 2020.

[32] Salomon Sickert, Javier Esparza, Stefan Jaax, and Jan Křetínskỳ. Limit-deterministic Büchi automata for linear temporal logic. In *International Conference on Computer Aided Verification (CAV)*, pages 312–332. Springer, 2016.

[33] Jan Kretínský, Tobias Meggendorfer, and Salomon Sickert. Owl: A library for ω-words, automata, and LTL. In *International Symposium on Automated Technology for Verification and Analysis (ATVA)*, volume 11138 of *LNCS*, pages 543–550. Springer, 2018.

[34] Ernst Moritz Hahn, Guangyuan Li, Sven Schewe, Andrea Turrini, and Lijun Zhang. Lazy probabilistic model checking without determinisation. In *International Conference on Concurrency Theory (CONCUR)*, pages 354–367, 2015.

[35] Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888, 2006.

[36] D. Silver et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, January 2016.

[37] Yi Wan, Abhishek Naik, and Richard S Sutton. Learning and planning in average-reward Markov decision processes. *arXiv preprint arXiv:2006.16318*, 2020.

[38] A Prasad Sistla. Safety, liveness and fairness in temporal logic. *Formal Aspects of Computing*, 6(5):495–511, 1994.

[39] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287, 1999.

[40] Alper Kamil Bozkurt, Yu Wang, Michael M Zavlanos, and Miroslav Pajic. Control synthesis from linear temporal logic specifications using model-free reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10349–10355. IEEE, 2020.

[41] Giuseppe De Giacomo and Moshe Vardi. Synthesis for LTL and LDL on finite traces. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[42] Alberto Camacho, Jorge A Baier, Christian Muise, and Sheila A McIlraith. Finite LTL synthesis as planning. In *Twenty-Eighth International Conference on Automated Planning and Scheduling*, 2018.

[43] Giuseppe De Giacomo, Luca Iocchi, Marco Favorito, and Fabio Patrizi. Foundations for restraining bolts: Reinforcement learning with LTLf/LDLf restraining specifications. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29(1), pages 128–136, 2019.

[44] Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*, pages 2107–2116, 2018.

[45] Alvaro Velasquez, Brett Bissey, Lior Barak, Andre Beckus, Ismail Alkhouri, Daniel Melcer, and George Atia. Dynamic automaton-guided reward shaping for monte carlo tree search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35(13), pages 12015–12023, 2021.

[46] Zhe Xu, Ivan Gavran, Yousef Ahmad, Rupak Majumdar, Daniel Neider, Ufuk Topcu, and Bo Wu. Joint inference of reward machines and policies for reinforcement learning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 590–598, 2020.

[47] H.A.P. Blom and J. Lygeros. *Stochastic Hybrid Systems: Theory and Safety Critical Applications*. Number 337 in Lecture Notes in Control and Information Sciences. Springer Verlag, Berlin Heidelberg, 2006.

[48] Rupak Majumdar, Kaushik Mallik, and Sadegh Soudjani. Symbolic controller synthesis for Büchi specifications on stochastic systems. In *Hybrid Systems: Computation and Control (HSCC)*, New York, NY, USA, 2020. ACM.

[49] Abolfazl Lavaei, Fabio Somenzi, Sadegh Soudjani, Ashutosh Trivedi, and Majid Zamani. Formal controller synthesis for continuous-space MDPs via model-free reinforcement learning. In *International Conference on Cyber-Physical Systems (ICCPS)*, pages 98–107, 2020.

[50] Hansol Yoon, Yi Chou, Xin Chen, Eric Frew, and Sriram Sankaranarayanan. Predictive runtime monitoring for linear stochastic systems and applications to geofence enforcement for UAVs. In *International Conference on Runtime Verification*. Springer, 2019.

[51] Sofie Haesaert and Sadegh Soudjani. Robust dynamic programming for temporal logic control of stochastic systems. *IEEE Transactions on Automatic Control*, 66(6):2496–2511, 2020.

[52] Xuchu Ding, Stephen L Smith, Calin Belta, and Daniela Rus. Optimal control of Markov decision processes with linear temporal logic constraints. *IEEE Transactions on Automatic Control*, 59(5):1244–1257, 2014.

[53] Pranav Ashok, Krishnendu Chatterjee, Przemysław Daca, Jan Křetínskỳ, and Tobias Meggendorfer. Value iteration for long-run average reward in Markov decision processes. In *International Conference on Computer Aided Verification*, pages 201–221. Springer, 2017.

[54] George K Atia, Andre Beckus, Ismail Alkhouri, and Alvaro Velasquez. Verifiable planning in expected reward multichain MDPs. *arXiv preprint arXiv:2012.02178*, 2020.

[55] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[56] Benjamin Recht. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, pages 253–279, 2018.

[57] Yonathan Efroni, Shie Mannor, and Matteo Pirotta. Exploration-exploitation in constrained MDPs. *arXiv preprint arXiv:2003.02189*, 2020.

[58] Tomáš Brázdil, Krishnendu Chatterjee, Martin Chmelik, Vojtěch Forejt, Jan Křetínskỳ, Marta Kwiatkowska, David Parker, and Mateusz Ujma. Verification of Markov decision processes using learning algorithms. In *Automated Technology for Verification and Analysis (ATVA)*, pages 98–114. Springer, 2014.

[59] Jie Fu and Ufuk Topcu. Probably approximately correct MDP learning and control with temporal logic constraints. In *Proceedings of Robotics: Science and Systems*, 2014.

[60] 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. In *IEEE Conference on Decision and Control (CDC)*, pages 5338–5343. IEEE, 2019.

[61] Ryohei Oura, Ami Sakakibara, and Toshimitsu Ushio. Reinforcement learning of control policy for linear temporal logic specifications using limit-deterministic Büchi automata. *IEEE Control Systems Letters*, 4(3):761–766, July 2020.

[62] Yuqian Jiang, Suda Bharadwaj, Bo Wu, Rishi Shah, Ufuk Topcu, and Peter Stone. Temporal-logic-based reward shaping for continuing reinforcement learning tasks. *Good Systems-Published Research*, 2021.

[63] Vektor Dewanto, George Dunn, Ali Eshragh, Marcus Gallagher, and Fred Roosta. Average-reward model-free reinforcement learning: a systematic review and literature mapping. *arXiv preprint arXiv:2010.08920*, 2020.

[64] Yasin Abbasi-Yadkori, Peter Bartlett, Kush Bhatia, Nevena Lazic, Csaba Szepesvari, and Gellert Weisz. POLITEX: Regret bounds for policy iteration using expert prediction. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 3692–3702. PMLR, 09–15 Jun 2019.

[65] Chen-Yu Wei, Mehdi Jafarnia Jahromi, Haipeng Luo, Hiteshi Sharma, and Rahul Jain. Model-free reinforcement learning in infinite-horizon average-reward Markov decision processes. In *International conference on machine learning*, pages 10170–10180. PMLR, 2020.

[66] Rahul Singh, Abhishek Gupta, and Ness B Shroff. Learning in Markov decision processes under constraints. *arXiv preprint arXiv:2002.12435*, 2020.

[67] Jan Kretínský, Fabian Michel, Lukas Michel, and Guillermo Perez. Finite-memory near-optimal learning for Markov decision processes with long-run average reward. In *Conference on Uncertainty in Artificial Intelligence*, pages 1149–1158. PMLR, 2020.