







# An Impossibility Result in Automata-Theoretic Reinforcement Learning <sup>\*</sup>

Ernst Moritz Hahn<sup>1</sup>, Mateo Perez<sup>2</sup>, Sven Schewe<sup>3</sup>, Fabio Somenzi<sup>2</sup>,  
Ashutosh Trivedi<sup>2</sup>, and Dominik Wojtczak<sup>3</sup>

<sup>1</sup> University of Twente, The Netherlands

<sup>2</sup> University of Colorado Boulder, USA

<sup>3</sup> University of Liverpool, UK


**Abstract.** The expanding role of reinforcement learning (RL) in safety-critical system design has promoted  $\omega$ -automata as a way to express learning requirements—often non-Markovian—with greater ease of expression and interpretation than scalar reward signals. When  $\omega$ -automata were first proposed in model-free RL, deterministic Rabin acceptance conditions were used in an attempt to provide a direct translation from  $\omega$ -automata to finite state “reward” machines defined over the same automaton structure (a memoryless reward translation). While these initial attempts to provide faithful, memoryless reward translations for Rabin acceptance conditions remained unsuccessful, translations were discovered for other acceptance conditions such as suitable, limit-deterministic Büchi acceptance or more generally, good-for-MDP Büchi acceptance conditions. Yet, the question “whether a *memoryless* translation of Rabin conditions to scalar rewards exists” remained unresolved.

This paper presents an impossibility result implying that any attempt to use Rabin automata directly (without extra memory) for model-free RL is bound to fail. To establish this result, we show a link between a class of automata enabling memoryless reward translation to closure properties of its accepting and rejecting infinity sets, and to the insight that both the property and its complement need to allow for positional strategies for such an approach to work. We believe that such impossibility results will provide foundations for the application of RL to safety-critical systems.

## 1 Introduction

The empirical success of reinforcement learning (RL, [26]) in solving challenging problems, even in the absence of an explicit model of the environment, has made its application to the design of safety-critical systems inevitable. However, traditional RL relies on expert inputs in the form of scalar reward signals that are often designed in intuitive, empirical fashion. A rigorous approach to the

---

<sup>\*</sup>  This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreements 864075 (CAESAR), and 956123 (FOCETA). This work is supported in part by the National Science Foundation (NSF) grant CCF-2009022 and by NSF CAREER award CCF-2146563.

design of safety-critical systems demands formal specifications at every stage of the design process. Consequently, there is an increased interest in formal languages that express learning requirements and in their automatic translation to reward signals for model-free RL. *This paper concerns an impossibility result on the automatic reward translations when requirements are specified using an important class of formal languages: the  $\omega$ -regular objectives.*

Omega-regular languages [27,21] have been used to specify high level objectives in the safety-critical system-design community for decades—often in the form of declarative specifications in Linear Temporal Logic [23]. Omega-regular objectives express qualitative properties for infinite-horizon behaviors, extending what regular languages do for finite-horizon behaviors: they are expressive, robust, and support efficient, automatic analysis. However, unlike regular languages, for which deterministic finite automata provide a *de facto* canonical machine model, the machine models for  $\omega$ -regular objectives are characterized by infinitary *acceptance conditions* to be satisfied by the infinite-horizon behavior. Widely adopted acceptance conditions include Büchi, parity, Rabin, and Streett conditions [2], and the name of the acceptance condition customarily precedes “automata” to specify the kind of  $\omega$ -automata. Deterministic parity, Rabin, Streett, and nondeterministic Büchi automata capture the same class of languages and characterize the class of  $\omega$ -regular languages; while, deterministic Büchi automata are strictly less expressive.

It is known that unrestricted nondeterminism is not compatible with the computation of optimal strategies when they are used to express the properties of probabilistic systems modeled as Markov decision processes (MDPs). This has motivated the study of a restricted form of nondeterminism formalized as the *good-for-MDPs automata* [15,9,30]. Notably, good-for-MDPs Büchi automata are expressive enough to represent all  $\omega$ -regular objectives. The optimal control problem for MDPs against  $\omega$ -automata based specifications has been studied extensively [6,1,22]. These solution methods, however, require a model of the environment dynamics. In the RL framework, the system, specified as an MDP, is unknown. Model-free RL algorithms synthesize a strategy without creating an explicit model of the dynamics. Thus, when designing a model-free translation from an  $\omega$ -automaton to rewards, the rewards should not depend directly on the unknown transition structure of the MDP. Model-free reward translations for good-for-MDPs Büchi automata [13,5] and parity automata [16] have been demonstrated; they assign reward from the transitions of the automaton.

When choosing a type of  $\omega$ -automata to use for a model-free reward translation, one must consider that there are optimal positional strategies in MDPs for discounted and average reward RL [24]. This means that, if a model-free reward translation results in an MDP where optimal strategies for the  $\omega$ -regular objective require additional memory, then this construction is incorrect. For instance, if one forms the synchronous product between a deterministic Streett automaton and an MDP, optimal policies in the resulting product MDP may require finite memory, so there is no faithful reward assignment on this product MDP.

Good-for-MDPs Büchi, deterministic parity, and deterministic Rabin automata admit positional optimal strategies in the product MDP.

Since good-for-MDPs Büchi and deterministic parity automata are known to have faithful model-free reductions, one may consider if a faithful model-free reduction exists for Rabin automata with no additional memory. Attempts at this [25] have later been shown to be incorrect [13]. We show that no such reduction exists. This somewhat surprising result explains why no attempt at using Rabin automata in RL has been successful. We observe connections between the existence of a model-free reward translation and the closure sets of Muller automata. We also show how the positional nature of optimal strategies for the complement of an automaton affects the existence of a model-free reward translation.

We begin the technical discussion by introducing  $\omega$ -automata (Section 2) and their closure properties (Section 3). Section 4 formalizes MDPs and the RL framework with both automata-theoretic rewards and non-Markovian scalar rewards. Section 5 introduces the idea of “memoryless” translations and associated impossibility theorems for Rabin. Section 6 provides concluding remarks.

## 2 Omega-Automata

A finite word over an alphabet  $\Sigma$  is a finite concatenation of symbols from  $\Sigma$ . Similarly, an  $\omega$ -word  $w$  over  $\Sigma$  is a function  $w : \omega \rightarrow \Sigma$  from the natural numbers to  $\Sigma$ . We write  $\Sigma^*$  and  $\Sigma^\omega$  for the set of finite and  $\omega$ -strings over  $\Sigma$ . We write  $\mathbb{B}$  for the binary alphabet  $\{0, 1\}$ .

**Definition 1.** *An  $\omega$ -automaton  $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$  consists of a finite alphabet  $\Sigma$ , a finite set of states  $Q$ , an initial state  $q_0 \in Q$ , a transition function  $\delta : Q \times \Sigma \rightarrow 2^Q$ , and an acceptance condition  $\alpha \subseteq Q^\omega$ . A deterministic automaton is such that  $\delta(q, \sigma)$  is a singleton for every state  $q$  and alphabet letter  $\sigma$ . For deterministic automata, we write  $\delta(q, \sigma) = q'$  instead of  $\delta(q, \sigma) = \{q'\}$ .*

A *run* of an automaton  $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$  on word  $w \in \Sigma^\omega$  is a function  $\rho : \omega \rightarrow Q$ , such that  $\rho(0) = q_0$  and  $\rho(i+1) \in \delta(\rho(i), w(i))$  holds. A run  $\rho$  is *accepting* if  $\rho \in \alpha$ . A word  $w$  is *accepted* by  $\mathcal{A}$  if there exists an accepting run of  $\mathcal{A}$  on  $w$ . The language of  $\mathcal{A}$ , written  $L(\mathcal{A})$ , is the set of words accepted by  $\mathcal{A}$ .

The set of states that appear infinitely often in  $\rho$  is written  $\text{Inf}(\rho)$ . A deterministic automaton  $\mathcal{D}$  has exactly one run for each word in  $\Sigma^\omega$ . We write  $\text{Inf}^{\mathcal{D}}(w)$  for the set of states that appear infinitely often in the unique run of  $\mathcal{D}$  on  $w$ . When the deterministic automaton  $\mathcal{D}$  is clear from the context, we drop the superscript and simply write  $\text{Inf}(w)$ .

**Definition 2 (Acceptance Conditions.).** *Several ways to give finite presentations of  $\alpha$  acceptance conditions<sup>4</sup> are in use. We recall the most common ones so as to fix notation. They are all defined in terms of  $\text{Inf}(\rho)$ .*

<sup>4</sup> Abusing notation, we sometimes use  $\alpha$  to denote the indicator function  $\alpha : Q^\omega \rightarrow \mathbb{B}$ .

- A Büchi acceptance condition is specified by a set  $F \subseteq Q$  such that

$$\alpha = \{\rho \in Q^\omega : \text{Inf}(\rho) \cap F \neq \emptyset\}.$$

- A co-Büchi acceptance condition is specified by a set  $F \subseteq Q$  such that

$$\alpha = \{\rho \in Q^\omega : \text{Inf}(\rho) \cap F = \emptyset\}.$$

- A generalized Büchi condition is specified by a set of sets  $\mathcal{F} \subseteq 2^Q$  such that

$$\alpha = \{\rho \in Q^\omega : \forall F \in \mathcal{F}. \text{Inf}(\rho) \cap F \neq \emptyset\}.$$

- A generalized co-Büchi condition is specified by a set of sets  $\mathcal{F} \subseteq 2^Q$  such that

$$\alpha = \{\rho \in Q^\omega : \exists F \in \mathcal{F}. \text{Inf}(\rho) \cap F = \emptyset\}.$$

- A parity acceptance condition of index  $k$  is specified by a function  $\pi : Q \rightarrow \{0, \dots, k-1\}$  that assigns a priority to each state of the automaton, so that

$$\alpha = \{\rho \in Q^\omega : \pi(\text{Inf}(\rho)) \text{ is odd}\},$$

where  $\pi(S) = \max\{\pi(s) : s \in S\}$  is the maximum priority of states in  $S \subseteq Q$ .

- A Rabin acceptance condition of index  $k$  is specified by  $k$  pairs of sets of states,  $\{\langle R_i, G_i \rangle\}_{1 \leq i \leq k}$ . Intuitively, a run should visit at least one set of Red (ruinous) states finitely often and its matching Green (good) set of states infinitely often. Formally,

$$\alpha = \{\rho \in Q^\omega : \exists i. 1 \leq i \leq k \text{ and } \text{Inf}(\rho) \cap R_i = \emptyset \text{ and } \text{Inf}(\rho) \cap G_i \neq \emptyset\}.$$

- A Streett acceptance condition of index  $k$  is specified by  $k$  pairs of sets of states,  $\{\langle G_i, R_i \rangle\}_{1 \leq i \leq k}$ . Intuitively, a run should visit each Red set of states finitely often or its matching Green set of states infinitely often. Formally,

$$\alpha = \{\rho \in Q^\omega : \forall i. 1 \leq i \leq k \rightarrow \text{Inf}(\rho) \cap R_i = \emptyset \text{ or } \text{Inf}(\rho) \cap G_i \neq \emptyset\}.$$

- A Muller acceptance condition is specified by a collection of sets of states  $\mathcal{C} \subseteq 2^Q$  such that

$$\alpha = \{\rho \in Q^\omega : \text{Inf}(\rho) \in \mathcal{C}\}.$$

The name of the acceptance condition customarily precedes automata to specify the kind of  $\omega$ -automata, e.g.  $\omega$ -automata with Büchi acceptance conditions are called Büchi automata. A Büchi (or co-Büchi) automaton where, in every strongly connected component (SCC) of the automaton, either all or none of the states is in  $F$ , is called *weak*.

### 3 Closure Properties of Acceptance Conditions

**Definition 3 (Eventual Sets).** An eventual set of a deterministic automaton  $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$  is a set of states  $E \subseteq Q$  such that  $E = \text{Inf}(w)$  for some  $w \in \Sigma^\omega$ . An eventual set is accepting if it satisfies the acceptance condition  $\alpha$ ; otherwise, it is rejecting. We denote the set of all eventual sets of  $\mathcal{A}$  by  $\mathcal{E}^{\mathcal{A}}$ . The set of accepting (rejecting) eventual sets of  $\mathcal{A}$  is written  $\mathcal{E}_a^{\mathcal{A}}$  ( $\mathcal{E}_r^{\mathcal{A}}$ ). Note that  $\mathcal{E}_a^{\mathcal{A}} \cup \mathcal{E}_r^{\mathcal{A}} = \mathcal{E}^{\mathcal{A}}$  and  $\mathcal{E}_a^{\mathcal{A}} \cap \mathcal{E}_r^{\mathcal{A}} = \emptyset$ . When the automaton  $\mathcal{A}$  is clear from the context, we drop the superscript and simply write,  $\mathcal{E}$ ,  $\mathcal{E}_a$ ,  $\mathcal{E}_r$ .

For a Muller automaton with acceptance condition  $\mathcal{C}$ , for example,  $\mathcal{E}_a = \mathcal{E} \cap \mathcal{C}$  and  $\mathcal{E}_r = \mathcal{E} \setminus \mathcal{C}$ . For a Büchi automaton with acceptance condition  $F$ ,  $\mathcal{E}_a = \{E \in \mathcal{E} : E \cap F \neq \emptyset\}$  and so on.

**Definition 4 (Upward-Closure and Closure under Union).** A set  $S \subseteq \mathcal{E}$  of eventual sets is upward-closed if, whenever  $E_1 \in S$ ,  $E_2 \in \mathcal{E}$ , and  $E_1 \subseteq E_2$ , it is also the case that  $E_2 \in S$ . The set  $S$  is closed under union if, whenever  $E_1, E_2, \dots, E_n$  are in  $S$ , and  $E_1 \cup E_2 \cup \dots \cup E_n \in \mathcal{E}$ , then it is also the case that  $E_1 \cup E_2 \cup \dots \cup E_n \in S$ . Note that upward closure implies closure under union because  $E_1 \cup E_2 \cup \dots \cup E_n \supseteq E_1$ .

We define “closure under union” in terms of union of  $n$  sets instead of just pairs to account for cases like this: we have rejecting eventual sets  $E_1, E_2, E_3$  such that their pairwise unions are not in  $\mathcal{E}$ , but the union of all three is. We still want the union to be in  $\mathcal{E}_r$ .  $E_1, E_2, E_3$  may form a ring that is only strongly connected when all three are taken together.

Table 1 summarizes the requirements that  $\mathcal{E}_a$  and  $\mathcal{E}_r$  must satisfy for the acceptance condition of a *deterministic* Muller automaton to be translated into an equivalent acceptance condition of different type for the same transition structure. Dual types of acceptance condition (e.g., Streett and Rabin) must satisfy dual requirements. A weak automaton may be regarded as both a Büchi automaton and a co-Büchi automaton. Hence, it has the most restrictive conditions. Likewise, a parity automaton may be seen as both a Rabin automaton and a Streett automaton. Hence the constraints imposed on parity conditions combine those of Rabin and Streett conditions.

*Remark 1.* The result for generalized Büchi acceptance extends [18, Theorem 4.2], because it says that, with generalized Büchi acceptance, one is not only guaranteed the existence of a deterministic Büchi automaton equivalent to the given Muller automaton, but is also told that one exists with the same transition structure as long as a generalized acceptance condition is used. Any standard technique to “degeneralize” that automaton may be applied to recover an automaton with plain Büchi acceptance.

As an example of how Table 1 is arrived at, we prove the following result. Analogous results are in [32, Lemma 13] and [21, Proposition 4.4.5].

**Table 1.** Closure conditions that  $\mathcal{E}_a$  and  $\mathcal{E}_r$  must satisfy for a Muller condition to be expressed as another type of condition. Positional and co-positional refer to the (guaranteed) existence of positional optimal policies for maximizing the chance that a run in  $\mathcal{E}_a$  and  $\mathcal{E}_r$ , respectively, is produced. The final column lists which of these target types permit memoryless reward translations (MRT) to scalar values with convex aggregator functions, which allows for using them in reinforcement learning.

target type	$\mathcal{E}_a$	$\mathcal{E}_r$	positional	co-positional	MRT
weak	upward	upward	yes	yes	yes
Büchi	upward	union	yes	yes	yes
co-Büchi	union	upward	yes	yes	yes
generalized Büchi	upward		no	yes	no
generalized co-Büchi		upward	yes	no	no
parity	union	union	yes	yes	yes
Streett	union		no	yes	no
Rabin		union	yes	no	no

**Theorem 1 (Muller to Rabin).** *A Muller condition is expressible in Rabin form if, and only if,  $\mathcal{E}_r$  is closed under union.*

*Proof.* To see that, in a Rabin condition  $\{(R_i, G_i)\}_{1 \leq i \leq k}$ ,  $\mathcal{E}_r$  is closed under union, let, for  $j = 1, \dots, n$ ,  $E_j$  be an element of  $\mathcal{E}_r$ . Then, for every  $i \in \{1, \dots, k\}$ ,  $E_j \cap R_i \neq \emptyset$  or  $E_j \cap G_i = \emptyset$  holds. If any  $E_j$  intersects  $G_i$ , so does  $E_1 \cup E_2 \cup \dots \cup E_n$ . In the remaining case, no  $E_j$  intersects  $R_i$ , but then neither does  $E_1 \cup E_2 \cup \dots \cup E_n$ .

To prove the other direction, i.e., that closure under union of  $\mathcal{E}_r$  guarantees the existence of an equivalent Rabin condition, we observe that closure under union of  $\mathcal{E}_r$  implies that, for  $E \in \mathcal{E}_a$ , the set

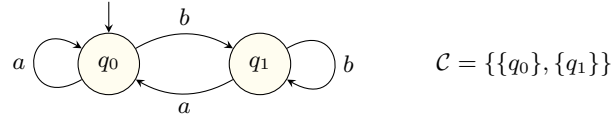
$$S_E = \bigcup \{S \in \mathcal{E}_r : S \subseteq E\}$$

is a proper subset of  $E$ . (Otherwise,  $E$  would belong to both  $\mathcal{E}_a$  and  $\mathcal{E}_r$ .) Therefore, a Rabin condition equivalent to the given Muller condition such that  $\mathcal{E}_r$  is closed under union consists of one pair  $\langle Q \setminus E, E \setminus S_E \rangle$  for each element  $E$  of  $\mathcal{E}_a$ . Every set  $E \in \mathcal{E}_a$  is accepting in the Rabin automaton thanks to the pair  $\langle Q \setminus E, E \setminus S_E \rangle$  because  $E \setminus S_E$  is not empty. For a set  $D \in \mathcal{E}_r$  and a generic pair  $\langle Q \setminus E, E \setminus S_E \rangle$  of the Rabin condition, we consider two cases. If  $D \subset E$ , then  $D \cap (E \setminus S_E) = \emptyset$ . If, however,  $D \not\subset E$ , given that  $D \neq E$  and  $D \neq \emptyset$ , it must be  $D \cap (Q \setminus E) \neq \emptyset$ . Hence, no pair in the Rabin condition makes  $D$  accepting.  $\square$

*Example 1.* For the deterministic Muller automaton of Figure 1,

$$\begin{aligned} \mathcal{E}_a &= \{\{q_0\}, \{q_1\}\} \\ \mathcal{E}_r &= \{\{q_0, q_1\}\} . \end{aligned}$$

The automaton accepts the language  $L = (\Sigma^* a^\omega) \cup (\Sigma^* b^\omega)$ . Since  $\mathcal{E}_r$  is (trivially) closed under union, the acceptance condition can be written in Rabin form. The construction of Theorem 1 yields  $\{\langle \{q_1\}, \{q_0\} \rangle, \langle \{q_0\}, \{q_1\} \rangle\}$ . Since  $\mathcal{E}_a$  is not



**Fig. 1.** A deterministic Muller automaton on the alphabet  $\Sigma = \{a, b\}$ .

closed under union, the automaton of Figure 1 cannot be equipped with a parity condition so that it still accepts  $L$ .

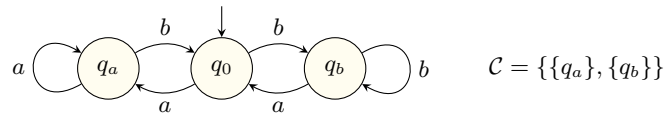
A deterministic Muller automaton for  $L$  that may be equipped with a parity acceptance condition is shown in Figure 2. State  $q_0$  is given priority 2, while the other two states are given priority 1.

*Remark 2.* If acceptance is defined in terms of transitions instead of states, the smallest deterministic Rabin automaton for the language  $L$  of Example 1 has one state, while the smallest deterministic parity automaton has two states. In general, translation from Rabin to parity may incur a factorial blow-up [19].

### 4 Markov Decision Processes

Let  $\mathbb{R}$  be the set of real numbers. Let  $\mathcal{D}(S)$  be the set of distributions over the 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,  $T : S \times A \rightarrow \mathcal{D}(S)$  is the probabilistic transition function,  $AP$  is the set of atomic propositions, and  $L : S \rightarrow 2^{AP}$  is the labeling function.

For any state  $s \in S$ ,  $A(s)$  denotes the set of actions that may 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  $p(s'|s, a)$ , that is, the probability that the MDP moves from state  $s$  to state  $s'$  if action  $a$  is chosen. A run of  $\mathcal{M}$  is an  $\omega$ -word  $\langle s_0, a_1, s_1, \dots \rangle \in S \times (A \times S)^\omega$  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  $\langle s_0, a_1, s_1, \dots \rangle \in S \times (A \times S)^*$ . For a run  $r = \langle s_0, a_1, s_1, \dots \rangle$  we define the corresponding labeled run as  $L(r) = \langle L(s_0), L(s_1), \dots \rangle \in (2^{AP})^\omega$ . We write  $Runs^{\mathcal{M}}(FRuns^{\mathcal{M}})$  for the set of runs (finite runs) of the MDP  $\mathcal{M}$  from its initial state 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  $s$ . When



**Fig. 2.** A deterministic Muller automaton, equivalent to the one of Figure 1, that may be equipped with a parity condition.

the context resolves ambiguity, we drop the superscript and simply write *Runs* and *FRuns*. We write  $last(r)$  for the last state of a finite run  $r$ .

A *strategy* for  $\mathcal{M}$  is a function  $\sigma : FRuns^{\mathcal{M}} \rightarrow \mathcal{D}(A)$  such that  $supp(\sigma(r)) \subseteq A(last(r))$ , where  $supp(d)$  denotes the support of the distribution  $d$ . A strategy  $\sigma$  is *pure* if  $\sigma(r)$  is a point distribution for all runs  $r \in FRuns^{\mathcal{M}}$  and is *mixed* otherwise. Let  $Runs_{\sigma}^{\mathcal{M}}(s)$  denote the subset of runs  $Runs^{\mathcal{M}}(s)$  that are possible under strategy  $\sigma$  from state  $s$ . We say that  $\sigma$  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  $\sigma : S \rightarrow \mathcal{D}(A)$ . A strategy is *positional* if it is both pure and stationary. We write  $\Sigma_{\mathcal{M}}$  for the set of all strategies on  $\mathcal{M}$  and  $\Pi_{\mathcal{M}}$  for the set of positional strategies on  $\mathcal{M}$ .

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

#### 4.1 Optimal Strategies Against $\omega$ -Automata

Given an MDP  $\mathcal{M} = (S, s_0, A, T, AP, L)$  and deterministic automaton  $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$  with  $\Sigma = 2^{AP}$ , a strategy  $\sigma$  determines sequences  $X_i, Q_i$ , and  $Y_i$  of random variables denoting the  $i^{th}$  state of the MDP, state of the automaton, and action, respectively, where  $Q_0 = q_0$  and, for  $i > 0$ ,  $Q_i = \delta(Q_{i-1}, L(X_{i-1}))$ . We define the optimal satisfaction probability  $PSem_{\mathcal{A}}^{\mathcal{M}}(s)$  as

$$PSem_{\mathcal{A}}^{\mathcal{M}}(s) = \sup_{\sigma \in \Sigma_{\mathcal{M}}} \mathbb{E}_{\sigma}^{\mathcal{M}}(s) \{ \alpha(\langle Q_0, Q_1, \dots \rangle) \} .$$

We say that a strategy  $\sigma \in \Sigma_{\mathcal{M}}$  is optimal for  $\mathcal{A}$  if

$$PSem_{\mathcal{A}}^{\mathcal{M}}(s) = \mathbb{E}_{\sigma}^{\mathcal{M}}(s) \{ \alpha(\langle Q_0, Q_1, \dots \rangle) \} .$$

#### 4.2 Optimal Strategies Against Scalar Rewards

Reinforcement learning [26] is a sequential optimization approach where a decision maker learns to optimally resolve a sequence of choices from feedback provided by an unknown or partially known environment. This feedback takes the form of rewards and punishments with strength proportional to the fitness of the decisions taken by the agent as judged by the environment towards some higher-level objectives. RL is inspired by the way dopamine-driven organisms latch on to past rewarding actions. Historically, RL paradigms integrated a myopic way of looking at the reward sequences in the form of discounted reward, with additional notions such as average reward being introduced more recently.

For simplicity, we take an abstract interpretation of RL as a sampling-based optimization approach that asymptotically converges to the optimal values and



policies for a given cost objective  $\text{Agg}$ . We will make some assumptions on RL that are known to hold for popular RL algorithms based on total reward, discounted reward, multi-discounted reward, and average reward RL.

A *rewardful* MDP is a tuple  $\mathcal{M} = (S, s_0, A, T, \rho, \text{Agg})$  where  $S, s_0, A$ , and  $T$  are defined in a similar way as for MDPs,  $\rho : S \times A \times S \rightarrow \mathcal{D}(\mathcal{C})$  is a (stochastic) reward function with values in a given set  $\mathcal{C}$  of colors, and the *aggregator* function  $\text{Agg} : \mathcal{C}^\omega \rightarrow \mathbb{R}$  is a real-valued optimization objective that converts infinite sequences of colors to real numbers. Traditionally, the set of colors  $\mathcal{C}$  is often the set of real numbers encoding a scalar reward signal; however permitting more general sets of colors has paved the way for more expressive reward aggregations, such as state-dependent discounting, reachability reward, and average reward-per-cost objectives. We extend  $\rho$  from transitions to runs by  $\hat{\rho} : \text{Runs} \rightarrow \mathcal{D}(\mathcal{C})^\omega$  in a straightforward manner. Some common aggregator functions are listed below.

- The *reachability-sum* (also, stochastic shortest path [3]) aggregator  $\text{Reach} : (\mathbb{R} \times \{0, 1\})^\omega \rightarrow \mathbb{R}$  is defined as

$$\text{Reach} : \langle (r_0, b_0), (r_1, b_1), \dots \rangle \mapsto \liminf_{n \rightarrow \infty} \sum_{0 \leq i < \min\{j \leq n : b_j = 1\}} r_i,$$

and provides the sum of rewards until the first appearance of  $b_i=1$  (reachability). If  $b_i=0$  for all  $i$ , one obtains the *total-sum* aggregator  $\text{Total} : \mathbb{R}^\omega \rightarrow \mathbb{R}$ .

- The *variable-discounted-sum* (also, state-dependent discounted) [11] aggregator  $D : (\mathbb{R} \times [0, 1])^\omega \rightarrow \mathbb{R}$  is defined as

$$D : \langle (r_0, \lambda_0), (r_1, \lambda_1), \dots \rangle \mapsto \lim_{n \rightarrow \infty} \sum_{i < n} \left[ \prod_{0 \leq j < i} \lambda_j \right] r_i.$$

When the discount is a constant  $\lambda$ , variable-discounted-sum recovers the classical discounted-sum aggregator  $D_\lambda : \mathbb{R}^\omega \rightarrow \mathbb{R}$ .

- The *reward-per-cost* [4] aggregator  $\text{RpC} : (\mathbb{R} \times \mathbb{R})^\omega \rightarrow \mathbb{R}$  is defined as

$$\text{RpC} : \langle (r_0, c_0), (r_1, c_1), \dots \rangle \mapsto \liminf_{n \rightarrow \infty} \frac{\sum_{0 \leq i < n} r_i}{\sum_{0 \leq i < n} c_i},$$

with suitable assumptions on the cost sequence  $\langle c_0, c_1, \dots \rangle$  to avoid division by zero. For the cost sequence  $\langle 1, 1, \dots \rangle$ , reward-per-cost reduces to the *average* aggregator  $\text{Avg} : \mathbb{R}^\omega \rightarrow \mathbb{R}$ .

- The *limit inferior* [8]  $\text{LimInf} : \mathbb{R}^\omega \rightarrow \mathbb{R}$  is defined as

$$\text{LimInf} : \langle r_0, r_1, \dots \rangle \mapsto \liminf_{n \rightarrow \infty} r_i.$$

The *limit superior* aggregator  $\text{LimSup} : \mathbb{R}^\omega \rightarrow \mathbb{R}$  is defined analogously.

Since an optimization problem over an MDP is specified by a reward function and an aggregator, we refer to aggregators as optimization objectives (or simply objectives). A rewardful MDP  $\mathcal{M}$  under a strategy  $\sigma$  determines a sequence

of random colors  $\rho(X_{i-1}, Y_i, X_i)_{i \geq 1}$ , where  $X_i$  and  $Y_i$  are the random variables denoting the  $i^{\text{th}}$  state and action, respectively. For an objective  $\text{Agg}$  and an initial state  $s$ , we define the optimal reward as

$$\text{Agg}_*^{\mathcal{M}}(s) = \sup_{\sigma \in \Sigma} \mathbb{E}_{\sigma}^{\mathcal{M}}(s) \{ \text{Agg}(\hat{\rho}(\langle X_0, Y_1, X_1, \dots \rangle)) \} .$$

We say that a strategy  $\sigma$  is optimal for the objective  $\text{Agg}$  if, for all  $s \in S$ ,

$$\text{Agg}_*^{\mathcal{M}}(s) = \mathbb{E}_{\sigma}^{\mathcal{M}}(s) \{ \text{Agg}(\hat{\rho}(\langle X_0, Y_1, X_1, \dots \rangle)) \} .$$

**Definition 5 (Positional objectives).** *An optimization objective  $\text{Agg}$  is positional if, for every MDP  $\mathcal{M}$ , there exists a positional optimal strategy for  $\text{Agg}$ .*

**Definition 6 (Convex combination of strategies).** *We say that a strategy  $\sigma \in \Sigma$  is a convex combination of strategies  $\sigma_1, \sigma_2 \in \Sigma$  if, for some  $\beta \in [0, 1]$ , for every run  $r \in \text{FRuns}$  and every  $a \in A$ ,  $\sigma(r)(a) = \beta\sigma_1(r)(a) + (1-\beta)\sigma_2(r)(a)$ .*

**Definition 7 (Convex objectives).** *An objective  $\text{Agg}$  is convex, if the set of optimal stationary strategies, for every MDP  $\mathcal{M}$ , is convex. In particular, an objective  $\text{Agg}$  is convex if any convex combination of positional optimal strategies for  $\text{Agg}$  is a stationary optimal strategy.*

The objectives listed above are both positional and convex. (See, e.g., [24] for the discounted and average objectives.)

## 5 Memoryless Reward Translations for RL

We are interested in memoryless reward translation from  $\omega$ -regular acceptance conditions to optimization problems with various aggregation semantics. Our notion of a memoryless reward translation is similar in spirit to the notion of Blackwell optimality [24] in reducing the average reward to discounted-reward that continues to operate on the same MDP state space, but accommodates the presence of a hyperparameter (the discount factor). Another example of such memoryless translation is the reduction of [13] from limit-deterministic Büchi automata to average-reward objectives with an unknown hyperparameter  $\zeta$ .

### 5.1 Memoryless Reward Translation

To compute strategies that maximize the probability that an MDP  $\mathcal{M}$  satisfies an  $\omega$ -regular objective by reinforcement learning, one defines a rewardful MDP  $\mathcal{M}^{\times}$  such that, from the optimal strategies for  $\mathcal{M}^{\times}$ , strategies may be derived for  $\mathcal{M}$  that maximize the probability that a run of  $\mathcal{M}$  satisfies the objective. The construction of  $\mathcal{M}^{\times}$  that we consider builds a product of the transition structures of the MDP  $\mathcal{M}$  and the automaton  $\mathcal{A}$  that accepts the objective. The reward for a state-action pair of  $\mathcal{M}^{\times}$  depends on the state of the automaton and, possibly, on a fixed set of parameters whose values depend on  $\mathcal{M}$ . Finally, a

suitable objective must be chosen. This way of converting an  $\omega$ -regular objective into an RL objective is quite general and encompasses the approaches adopted in the literature, e.g., [25,13,16,5]. The fixed set of parameters stand for the hyperparameters of reinforcement learning.

With this scheme, even if the strategies computed for  $\mathcal{M}^\times$  are positional, the strategies for  $\mathcal{M}$  require the memory supplied by the transition structure of the automaton. Since the reward only depends on the state of the automaton, and on no other memory, we call this translation *memoryless*.

In detail, let  $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$  be an automaton and  $P$  a set of parameter values. A *reward assignment function* for  $\mathcal{A}$  and  $P$  is a function  $f : Q \times \Sigma \times Q \times P \rightarrow \mathcal{D}(\mathcal{C})$ . Given an MDP  $\mathcal{M}$ , an automaton  $\mathcal{A}$ , a reward assignment function  $f$ , values for the parameters  $\hat{P}$ , and an aggregator  $\mathbf{Agg}$ , we define the *rewardful product* MDP

$$\mathcal{M}^\times = (S \times Q, (s_0, q_0), A \times Q, T^\times, \rho^\times, \mathbf{Agg}) ,$$

where  $T^\times : (S \times Q) \times (A \times Q) \rightarrow \mathcal{D}(S \times Q)$  is such that

$$T^\times((s, q), (a, q'))((s', q')) = T(s, a)(s')$$

if  $q' \in \delta(q, L(s))$  and 0 otherwise. This definition of  $T^\times$  delegates to the strategy for  $\mathcal{M}^\times$  the resolution of nondeterminism for nondeterministic automata. The reward function  $\rho^\times : (S \times Q) \times (A \times Q) \times (S \times Q) \rightarrow \mathcal{D}(\mathcal{C})$  is defined by

$$\rho^\times((s, q), (a, q'), (s', q')) = f(q, a, q', \hat{P})$$

if  $q' \in \delta(q, L(s))$  and 0 otherwise.

**Definition 8 (Memoryless Translation).** *Given  $\mathcal{A} = \langle 2^{AP}, Q, q_0, \delta, \alpha \rangle$  and payoff  $\mathbf{Agg}$ , we say that a memoryless translation exists from the acceptance condition  $\alpha$  to the aggregator function  $\mathbf{Agg}$ , and we write  $\alpha \hookrightarrow \mathbf{Agg}$ , when there exists a reward function  $f : Q \times 2^{AP} \times Q \times P \rightarrow \mathcal{D}(\mathcal{C})$  such that, for any MDP  $\mathcal{M}$  with atomic propositions  $AP$  and for any two strategies  $\sigma_1, \sigma_2 \in \Sigma_{\mathcal{M}}$ ,*

$$\mathbb{E}_{\sigma_1}^{\mathcal{M}^\times}(s)\{\alpha(\langle Q_0, Q_1, \dots \rangle)\} < \mathbb{E}_{\sigma_2}^{\mathcal{M}^\times}(s)\{\alpha(\langle Q_0, Q_1, \dots \rangle)\}$$

if, and only if, for every  $\varepsilon > 0$  there exists  $\hat{P} \in P$  (that may depend on the MDP) such that

$$\mathbb{E}_{\sigma_1}^{\mathcal{M}^\times}(s)\{\mathbf{Agg}(\hat{\rho}^\times(\langle X_0, Y_1, X_1 \dots \rangle))\} < \mathbb{E}_{\sigma_2}^{\mathcal{M}^\times}(s)\{\mathbf{Agg}(\hat{\rho}^\times(\langle X_0, Y_1, X_1 \dots \rangle))\} + \varepsilon$$

on the product  $\mathcal{M}^\times$  defined by  $\mathcal{M}$ ,  $\mathcal{A}$ , and  $f(\cdot, \cdot, \cdot, \hat{P})$ .

**Theorem 2.** *The following translations are memoryless:*

1. *good-for-MDPs Büchi objective to reachability-sum aggregator [13];*
2. *good-for-MDPs Büchi objective to variable-discounted-sum aggregator [5];*
3. *good-for-MDPs Büchi objective to average aggregator [14];*

4. *good-for-MDPs Büchi objective to total-sum aggregator [14]; and*
5. *parity objective to reachability-sum aggregator [16].*

Note that, although the reduction of [25] for Rabin automata does not use additional memory, it is incorrect [13], so it is not a memoryless translation. We will show in the next subsection that a memoryless translation for deterministic Rabin automata is impossible if the aggregator is convex.

## 5.2 Conditions for Memoryless Reductions

We are now in a position to discuss what the properties of acceptance conditions of  $\omega$ -automata discussed in Section 2 entail for a memoryless translation to exist.

**Definition 9 (Test MDP).** *A test MDP for the set of atomic propositions  $AP$  is*

$$\mathcal{M}_{AP} = (2^{AP}, s_0, 2^{AP}, T, AP, L) ,$$

*with one state and one action for each subset of  $AP$ , such that, for all states,  $L(s) = s$ , and  $p(s'|s, s') = 1$ . The initial state  $s_0$  is a subset of  $AP$ .*

All sequences of labels in  $2^{AP}$  that start with  $s_0$  may be produced by this test MDP.

**Theorem 3.** *Let  $\mathcal{A} = \langle 2^{AP}, Q, q_0, \delta, \alpha \rangle$  be a deterministic automaton and let  $\text{Agg}$  be a convex objective. Then  $\alpha \hookrightarrow \text{Agg}$  implies that  $\mathcal{E}_a^{\mathcal{A}}$  and  $\mathcal{E}_r^{\mathcal{A}}$  are closed under union.*

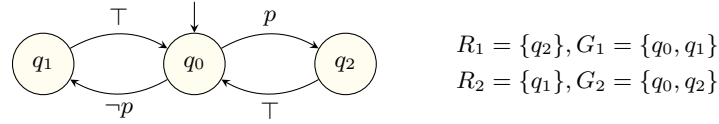
*Proof.* Suppose  $E_1, \dots, E_n \in \mathcal{E}_a$  (resp.  $\in \mathcal{E}_r$ ) are accepting (resp. rejecting) eventual sets of  $\mathcal{A}$  such that their union is in  $\mathcal{E}$ . Let  $\mathcal{M}_T$  be a test MDP for  $\mathcal{A}$  such that  $\bigcup_i E_i$  is reachable from  $\delta(q_0, s_0)$ . Such an  $s_0$  exists because all eventual sets are reachable from the automaton's initial state. Let  $\mathcal{M}^\times$  be the rewardful product defined by  $\mathcal{M}_T$ ,  $\mathcal{A}$ , and a reward function  $f$  for which  $\alpha \hookrightarrow \text{Agg}$ .

For each  $E_i$  there exists a stationary strategy  $\xi_i$  such that a run from any state in  $2^{AP} \times (\{q_0\} \cup \bigcup_i E_i)$  eventually dwells in  $E_i$ . Since  $\alpha \hookrightarrow \text{Agg}$  and  $\text{Agg}$  is convex,  $\xi_1, \dots, \xi_n$  are  $\varepsilon$ -maximal (resp.  $\varepsilon$ -minimal), so are their convex combination, and such combination strategies also maximize (resp. minimize) the probability of satisfying  $\alpha$ . Since there are combination strategies that visit all states in  $E_1 \cup \dots \cup E_n$  infinitely often,  $\mathcal{E}_a$  (resp.  $\mathcal{E}_r$ ) is closed under union.  $\square$

**Corollary 1.** *There exist a Rabin automaton with acceptance conditions  $\alpha$  such that, if  $\alpha \hookrightarrow \text{Agg}$ , then  $\text{Agg}$  is not convex.*

*Proof.* The deterministic Rabin automaton in Figure 3 is such that  $\mathcal{E}_a$  is not closed under union. Application of Theorem 3 yields the desired result.  $\square$

*Remark 3.* Note that the complement of  $\alpha$  for the Rabin automaton in Figure 3—a Streett condition that requires that both  $q_1$  and  $q_2$  be visited infinitely often—is not positional. Since  $\mathcal{E}_a$  is not closed under union and  $\mathcal{E}_r = \{\{q_0, q_1, q_2\}\}$  is upward closed, the acceptance condition of the automaton of



**Fig. 3.** A Rabin automaton on the alphabet  $\Sigma = 2^{\{p\}}$ .

Figure 3 may be expressed as a generalized co-Büchi condition, but not as a plain co-Büchi condition. Correspondingly, the complementary condition may be expressed as a generalized Büchi condition, but not as a plain Büchi condition for the transition structure of Figure 3.

The connection between closure under union of *both*  $\mathcal{E}_a$  and  $\mathcal{E}_r$  (a property called ‘ $\mathcal{E}_a$  has no splits’ there) and the existence of positional optimal strategies for *both* players was first established in [20, Theorem 6.2].

To be self-contained, we provide a proof for the direction we need, namely that closedness under union of  $\mathcal{E}_r$  entails memoryless strategies for a maximizer, while closedness under union of  $\mathcal{E}_a$  entails memoryless strategies for a minimizer.

**Lemma 1.** *Let  $\mathcal{A} = \langle 2^{AP}, Q, q_0, \delta, \alpha \rangle$  be a deterministic automaton and let  $\mathcal{A}^c$  be the automaton obtained from  $\mathcal{A}$  by complementing the acceptance condition (so that  $\mathcal{E}_a^{\mathcal{A}} = \mathcal{E}_r^{\mathcal{A}^c}$  and  $\mathcal{E}_r^{\mathcal{A}} = \mathcal{E}_a^{\mathcal{A}^c}$ ). If no strategy to produce an accepting word on  $\mathcal{A}$  ( $\mathcal{A}^c$ ) is positional, then  $\mathcal{E}_r^{\mathcal{A}}$  ( $\mathcal{E}_a^{\mathcal{A}}$ ) is not closed under union.*

*Proof.* Suppose that there is no positional strategy for  $\mathcal{A}$  ( $\mathcal{A}^c$ ). Then, for every  $E \in \mathcal{E}_a^{\mathcal{A}}$  ( $E \in \mathcal{E}_a^{\mathcal{A}^c}$ ), there exist states in  $E$  for which a pure strategy that visits all of  $E$  chooses different successors depending on the run up to that point. For at least one such state, the strategy depends on memory essentially, in the sense that, if the strategy is made positional at that state by choosing one successors among all successors chosen by the strategy, the resulting eventual set is a proper subset of  $E$  that belongs to  $\mathcal{E}_r^{\mathcal{A}}$  ( $\mathcal{E}_r^{\mathcal{A}^c}$ ). Moreover, depending on which successor is chosen, there must be more than one distinct restriction of  $E$  that is contained in  $\mathcal{E}_r^{\mathcal{A}}$  ( $\mathcal{E}_r^{\mathcal{A}^c}$ ), and the union of all such restrictions must be  $E$ . This shows that  $\mathcal{E}_r^{\mathcal{A}}$  ( $\mathcal{E}_r^{\mathcal{A}^c}$ ) is not closed under union. Observing that  $\mathcal{E}_a^{\mathcal{A}^c} = \mathcal{E}_r^{\mathcal{A}}$  ( $\mathcal{E}_a^{\mathcal{A}} = \mathcal{E}_r^{\mathcal{A}^c}$ ) completes the proof.  $\square$

Together with Lemma 1, we obtain the following corollary.

**Corollary 2.** *Let  $\mathcal{A} = \langle 2^{AP}, Q, q_0, \delta, \alpha \rangle$  be a deterministic automaton, let  $\text{Agg}$  be a convex objective, and let  $\mathcal{A}^c$  be the automaton obtained from  $\mathcal{A}$  by complementing the acceptance condition (so that  $\mathcal{E}_a^{\mathcal{A}} = \mathcal{E}_r^{\mathcal{A}^c}$  and  $\mathcal{E}_r^{\mathcal{A}} = \mathcal{E}_a^{\mathcal{A}^c}$ ). Then  $\mathcal{A}$  and  $\mathcal{A}^c$  have optimal positional strategies.*

Lemma 1 shows that a memoryless translation with a convex aggregator does not exist for deterministic Rabin and Streett automata. Lemma 1 provides a sufficient condition to check for the application of Theorem 3. Intuitively, if we

could compute the strategy for Rabin automata with RL, then we could compute positional strategies for the complement of the Rabin condition by maximizing the negative reward. Since the complement of a Rabin condition does not admit positional strategies in general, this is impossible. This impossibility also applies to generalized co-Büchi automata, as observed in Remark 3, as well as to their duals (Streett as the dual to Rabin and generalized Büchi automata as the dual to generalized co-Büchi), with the dual argument: here, it is the strategy of the maximizer itself that requires memory.

Theorem 3, and Corollary 2 provide the entries for the last three columns of Table 1.

## 6 Conclusion

The study of  $\omega$ -regular specifications for systems modeled as Markov decision processes was initiated in [30] and [9] resulting in a thriving research community (probabilistic model checking) developing principled techniques [2] and analysis tools [17] for the analysis of probabilistic systems. Reinforcement learning is a classical area of machine learning undergoing remarkable transformation under the deep learning revolution [12]; [26] provide a snapshot of both classical and recent results.

A combination of  $\omega$ -regular specifications with RL has potential to positively impact both research fields: for probabilistic model checking, RL offers the twinned advantages of scalability and an ability to reason with system without an explicit model; while for RL,  $\omega$ -regular objectives provide a rich specification language to express learning requirements instead of scalar rewards. As a result, the study of integrating formal requirements in RL in a way that is correct and efficient has attracted considerable interest [13,5,16]. At the same time, the machine learning community has advocated the need for non-Markovian reward signals in RL [28,10]. These representations often take the form of weighted automata called reward machines [28]. Automata-based rewards also serve as a memory mechanism for reasoning over partially observable environments [29], are useful for defining reward shaping functions to mitigate sparse reward signals [7], and can facilitate explanations of RL systems [31].

When  $\omega$ -regular objectives were first used in model checking MDPs, deterministic Rabin automata were used to represent the objectives. The same was attempted by the reinforcement learning community when they first turned to  $\omega$ -regular objectives: they tried the tested route through deterministic Rabin [25], but that translation fails as shown in [13]. This paper answers why any translation of Rabin conditions, even from a deterministic automaton, directly into scalar values is not possible for common reward aggregators used in reinforcement learning, like discounted and average reward. In a broader sense, this paper highlights the existence of positional optimal strategies that may not be computed by reinforcement learning. In so doing, the paper underscores the need for theoretical machine learning research at the promising intersection of the overlapping fields of probabilistic model checking and reinforcement learning.

## References

1. de Alfaro, L.: Formal Verification of Probabilistic Systems. Ph.D. thesis, Stanford University (1998)
2. Baier, C., Katoen, J.P.: Principles of Model Checking (Representation and Mind Series). The MIT Press (2008)
3. Bertsekas, D.: Reinforcement learning and optimal control. Athena Scientific (2019)
4. Bouyer, P., Brinksma, E., Larsen, K.G.: Optimal infinite scheduling for multi-priced timed automata. *Formal Methods in System Design* **32**(1), 3–23 (2008)
5. Bozkurt, A.K., Wang, Y., Zavlanos, M.M., Pajic, M.: Control synthesis from linear temporal logic specifications using model-free reinforcement learning. In: *International Conference on Robotics and Automation (ICRA)*. pp. 10349–10355 (2020)
6. Buhrke, N., Lescow, H., Vöge, J.: Strategy construction in infinite games with Streett and Rabin chain winning conditions. In: *Tools and Algorithms for the Construction and Analysis of Systems (TACAS 96)*. pp. 207–225 (1996), LNCS 1055
7. Camacho, A., Toro Icarte, R., Klassen, T.Q., Valenzano, R.A., McIlraith, S.A.: LTL and beyond: Formal languages for reward function specification in reinforcement learning. In: *IJCAI*. vol. 19, pp. 6065–6073 (2019)
8. Chatterjee, K., Doyen, L., Henzinger, T.A.: A survey of stochastic games with limsup and liminf objectives. In: *36th International Colloquium on Automata, Languages and Programming: Part II*. p. 1–15 (2009)
9. Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. *J. ACM* **42**(4), 857–907 (Jul 1995)
10. Gaon, M., Brafman, R.: Reinforcement learning with non-Markovian rewards. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 34-04, pp. 3980–3987 (2020)
11. Gimbert, H., Zielonka, W.: Limits of multi-discounted Markov decision processes. In: *Symposium on Logic in Computer Science (LICS 2007)*. pp. 89–98 (2007)
12. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: *Deep learning*, vol. 1. MIT Press (2016)
13. Hahn, E.M., Perez, M., Schewe, S., Somenzi, F., Trivedi, A., Wojtczak, D.: Omega-regular objectives in model-free reinforcement learning. In: *Tools and Algorithms for the Construction and Analysis of Systems*. pp. 395–412 (2019), LNCS 11427
14. Hahn, E.M., Perez, M., Schewe, S., Somenzi, F., Trivedi, A., Wojtczak, D.: Faithful and effective reward schemes for model-free reinforcement learning of omega-regular objectives. In: *ATVA: Automated Technology for Verification and Analysis*. pp. 108–124 (2020), LNCS 12302
15. Hahn, E.M., Perez, M., Schewe, S., Somenzi, F., Trivedi, A., Wojtczak, D.: Good-for-MDPs automata for probabilistic analysis and reinforcement learning. In: *Tools and Algorithms for the Construction and Analysis of Systems*. pp. 306–323 (2020), LNCS 12078
16. Hahn, E.M., Perez, M., Schewe, S., Somenzi, F., Trivedi, A., Wojtczak, D.: Model-Free Reinforcement Learning for Stochastic Parity Games. In: *CONCUR: International Conference on Concurrency Theory*. pp. 21:1–21:16 (Sep 2020), LIPIcs 171
17. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: *Computer Aided Verification*. pp. 585–591 (Jul 2011), LNCS 6806

18. Landweber, L.H.: Decision problems for  $\omega$ -automata. *Mathematical Systems Theory* **3**(4), 376–384 (1969)
19. Löding, C.: Optimal bounds for the transformation of omega-automata. In: *Foundations of Software Technology and Theoretical Computer Science*. pp. 97–109 (1999), INCS 1738
20. McNaughton, R.: Infinite games played on finite graphs. *Annals of Pure and Applied Logic* **65**, 149–184 (1993)
21. Perrin, D., Pin, J.É.: *Infinite Words: Automata, Semigroups, Logic and Games*. Elsevier (2004)
22. Piterman, N., Pnueli, A.: Faster solutions of Rabin and Streett games. In: *Symposium on Logic in Computer Science*. pp. 275–284 (2006)
23. Pnueli, A.: The temporal logic of programs. In: *IEEE Symposium on Foundations of Computer Science*. pp. 46–57 (1977)
24. Puterman, M.L.: *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New York, NY, USA (1994)
25. Sadigh, D., Kim, E., Coogan, S., Sastry, S.S., Seshia, S.A.: A learning based approach to control synthesis of Markov decision processes for linear temporal logic specifications. In: *Conference on Decision and Control (CDC)*. pp. 1091–1096 (Dec 2014)
26. Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press, second edn. (2018)
27. Thomas, W.: *Handbook of Theoretical Computer Science*, chap. Automata on Infinite Objects, pp. 133–191. The MIT Press/Elsevier (1990)
28. Toro Icarte, R., Klassen, T., Valenzano, R., McIlraith, S.: Using reward machines for high-level task specification and decomposition in reinforcement learning. In: *International Conference on Machine Learning*. pp. 2107–2116 (2018)
29. Toro Icarte, R., Waldie, E., Klassen, T., Valenzano, R., Castro, M., McIlraith, S.: Learning reward machines for partially observable reinforcement learning. *Advances in Neural Information Processing Systems* **32**, 15523–15534 (2019)
30. Vardi, M.Y.: Automatic verification of probabilistic concurrent finite state programs. In: *Foundations of Computer Science*. pp. 327–338 (1985)
31. Xu, Z., Wu, B., Ojha, A., Neider, D., Topcu, U.: Active finite reward automaton inference and reinforcement learning using queries and counterexamples. In: *Machine Learning and Knowledge Extraction*. pp. 115–135 (2021), INCS 12844
32. Zielonka, W.: Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science* **200**(1-2), 135–183 (1998)