Reward-Machine-Guided, Self-Paced Reinforcement Learning

Cevahir Koprulu[1] **Ufuk Topcu**[1]

[1]The University of Texas at Austin, USA

Abstract

Self-paced reinforcement learning (RL) aims to improve the data efficiency of learning by automatically creating sequences, namely *curricula*, of probability distributions over *contexts*. However, existing techniques for self-paced RL fail in long-horizon planning tasks that involve temporally extended behaviors. We hypothesize that taking advantage of prior knowledge about the underlying task structure can improve the effectiveness of self-paced RL. We develop a self-paced RL algorithm guided by reward machines, i.e., a type of finite-state machine that encodes the underlying task structure. The algorithm integrates reward machines in 1) the update of the policy and value functions obtained by any RL algorithm of choice, and 2) the update of the automated curriculum that generates context distributions. Our empirical results evidence that the proposed algorithm achieves optimal behavior reliably even in cases in which existing baselines cannot make any meaningful progress. It also decreases the curriculum length and reduces the variance in the curriculum generation process by up to one-fourth and four orders of magnitude, respectively.

1 INTRODUCTION

The design of task sequences, i.e., curricula [Bengio et al., 2009], aims to reduce the sample complexity of teaching reinforcement learning (RL) agents complex behaviors. Given a *target* task, a common curriculum design approach is to begin with *easier* tasks and increase the difficulty in a gradual manner, which requires domain expertise to define what is easy or hard [Narvekar et al., 2020]. To eliminate the need for manual curriculum design, many studies such as Baranes and Oudeyer [2010], Svetlik et al. [2017], Andrychowicz et al. [2017] focus on automating the process of curriculum generation. Klink et al. [2020a] adopt self-paced learning [Kumar et al., 2010] in RL by developing an algorithm that creates a sequence of probability distributions over *contexts* [Hallak et al., 2015]. The dynamics, the reward function, and the initial state distribution of an environment change with respect to the context. Given a *target context distribution*, a self-paced RL algorithm iteratively generates *context distributions* that maximizes the expected discounted return, regularized by the Kullback-Leibler (KL) divergence from the target context distribution.

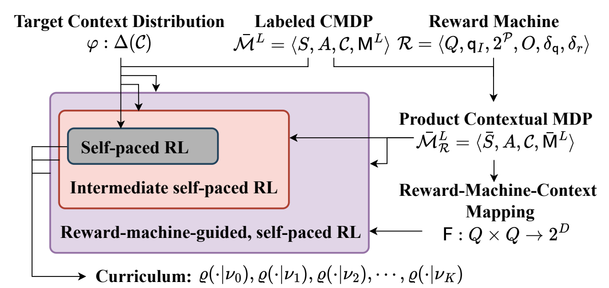


Figure 1: Workflow diagram for an existing self-paced RL approach, and two methods that we propose: intermediate self-paced RL and reward-machine-guided, self-paced RL.

Although empirical evidence by Klink et al. [2021] suggests that self-paced RL outperforms the state-of-the-art curriculum learning methods [Florensa et al., 2018, Portelas et al., 2020], existing self-paced RL approaches work poorly in long-horizon planning tasks, which involve temporally extended behaviors. We focus on tasks where the reward depends on the history of states and actions. In other words, the reward function of such a long-horizon planning task is non-Markovian. A remedy to tasks that require temporally extended behaviors is to expose the high-level structural relationships to the agent [Singh, 1992, Parr and Russell, 1997]. Icarte et al. [2018a] use a type of finite-state machine, called *reward machines*, as the high-level structural

Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023), PMLR 216:1121–1131.

knowledge to encode non-Markovian reward functions in RL.

We claim that exploiting the high-level structural knowledge about a long-horizon planning task can improve self-paced RL. To this end, we study self-paced RL for long-horizon planning tasks in which such knowledge is available a priori to the agent in the form of reward machines. Specifically, we focus on contextual long-horizon planning tasks, where the context parameterizes the dynamics and the non-Markovian reward function. The underlying temporal task structure remains the same irrespective of the context, hence a reward machine can encode all possible non-Markovian reward functions. We define a labeled contextual Markov decision process (MDP) to model such long-horizon planning tasks (see Figure 1).

Contribution. Our contribution is three-fold. 1) We propose an intermediate self-paced RL algorithm that combines a labeled contextual MDP and its reward machine in a product contextual MDP to update the policy and value functions of an RL agent. 2) We establish a mapping that, given a transition in the reward machine, outputs the smallest set of context parameters, that determine whether the transition, namely a high-level event, occurs or not. 3) We develop a reward-machine-guided, self-paced RL algorithm that exploits reward machines not only to update the policy and value functions but also to navigate the generation of curricula via the proposed mapping (see Figure 1).

Our experiments conclude that, first, proposed reward-machine-guided and intermediate self-paced RL algorithms enable RL agents to accomplish long-horizon planning tasks by encoding non-Markovian reward functions as reward machines, whereas state-of-the-art automated curriculum generation methods fail to do so; and, second, guiding curriculum generation via a *reward-machine-context mapping* not only boosts learning speed reliably but also stabilizes the curriculum generation process by reducing curricula variance by up to four orders of magnitude, and thus avoid inefficient exploration of the curriculum space.

2 RELATED WORK

We propose an automated curriculum generation method, that exploits high-level structural knowledge about long-horizon planning tasks. Our work falls under two subjects.

Curriculum learning for RL. Automatically generating curricula in RL modifies the configuration of the environment iteratively to accelerate convergence to optimal policies. As we do, many studies in the literature consider a curriculum as a sequence of distributions over environment configurations. Florensa et al. [2017] proposes the generation of distributions over initial states that iteratively get further away from goal states. Others focus on goal-conditioned RL where a curriculum is a sequence of distributions over goal states that optimize value disagreement [Zhang et al., 2020], feasibility and coverage of goal states [Racaniere et al., 2020], intrinsic motivation [Baranes and Oudeyer, 2010, Portelas et al., 2020], and intermediate goal difficulty [Florensa et al., 2018]. For procedural content generation environments, curricula prioritize levels with higher learning potential [Jiang et al., 2021b,a]. In comparison, self-paced RL is adopted from supervised learning where training samples are automatically ordered in increasing complexity [Kumar et al., 2010, Jiang et al., 2015]. Ren et al. [2018] consider curricula as a sequence of environment interactions and proposes a self-paced mechanism that minimizes coverage penalty. Eimer et al. [2021]'s work generates a sequence of contexts, not distributions, with respect to their capacity of value improvement. Klink et al. [2020a,b, 2021, 2022], Koprulu et al. [2023] formulate the generation of curricula as interpolations between distributions over contexts. Similarly, Chen et al. [2021] study interpolations between task distributions, but not under the self-paced RL framework.

Incorporating high-level structural knowledge into RL. Singh [1992], Parr and Russell [1997], Sutton et al. [1999], Dietterich [2000] propose the idea of incorporating high-level structural knowledge to decompose a task into a hierarchy of subtasks. The proposed hierarchy allows the agent to learn a meta-controller that chooses between subtasks to pursue, and a low-level controller that acts in the chosen subtask. Another way to incorporate such knowledge is to capture temporal abstractions in long-horizon planning tasks via temporal logic [Bacchus et al., 1996, Li et al., 2017, Littman et al., 2017], or reward machines [Icarte et al., 2018a, Camacho et al., 2019], which address MDPs with non-Markovian structures. We investigate a multi-task setting with non-Markovian reward functions and propose an automated curriculum generation approach that uses reward machines, 1) to encode non-Markovian reward functions; and 2) to guide the curriculum generation process. Similar to curriculum learning, Icarte et al. [2018b], Xu and Topcu [2019], Kuo et al. [2020], Zheng et al. [2022], Velasquez et al. [2021] study the use of temporal logic and reward machines in topics such as generalization, transfer learning, and multi-task learning.

3 PRELIMINARIES

In this section, we provide the background for our problem of interest. We illustrate a two-door environment, (see Figure 2), which we will revisit throughout the paper. The agent has to complete 4 subtasks in the following order: (1) Passing through *Door 1*, (2) getting a key from *Box*, (3) opening *Door 2* with the key, and (4) arriving at *Goal*. The agent has to avoid hitting the walls that separate the rooms.

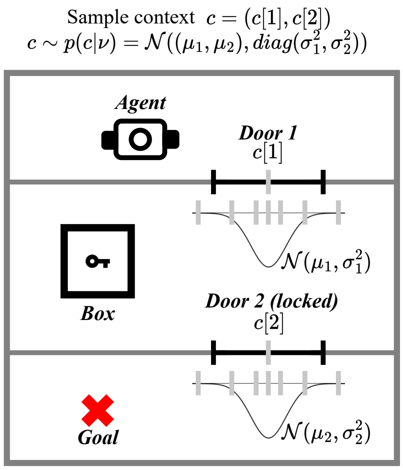


Figure 2: The two-door environment: A context $c = (c[1], c[2])$ determines the positions of *Door 1* and *Door 2* with $c[1]$ and $c[2]$, respectively.

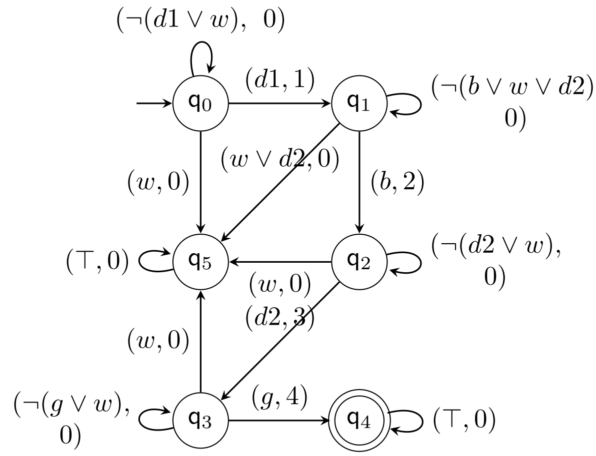


Figure 3: Reward machine of the two-door environment

3.1 LABELED MDPS AND REWARD MACHINES

Definition 1. *A labeled Markov decision process Xu et al. [2020] is a tuple $\mathcal{M} = \langle S, A, p, R, \phi, \gamma, \mathcal{P}, L\rangle$ consisting of a state space S, an action space A, a probabilistic transition function $p\colon S \times A \times S \to [0, 1]$, and an initial state distribution $\phi : S \to [0, 1]$. A reward function $R : (S \times A)^+ \times S \to \mathbb{R}$, and a discount factor $\gamma \in [0, 1)$ specify the returns to the agent. A finite set $\mathcal{P}$ of propositional variables, and a labeling function $L : S \times A \times S \to 2^{\mathcal{P}}$ determine the set of high-level events that the agent sees in the environment.*

A *policy* π is a function mapping states in S to a probability distribution over actions in A. At state $s \in S$, an agent using policy π picks an action a with probability $\pi(s, a)$, and the new state s' is chosen with probability $p(s, a, s')$.

For a fixed context, we can model the two-door environment as a labeled MDP. The states are the coordinates of the agent and the actions are moving in the four cardinal directions, whereas the transitions are deterministic. The agent receives the labels $\{d1\}$, $\{b\}$, $\{d2\}$, $\{g\}$, and $\{w\}$ when it moves onto *Door 1*, *Box*, *Door 2*, *Goal*, and the walls, respectively.

Definition 2. *A reward machine Icarte et al. [2018a] $\mathcal{R} = \langle Q, \mathsf{q}_I, 2^{\mathcal{P}}, O, \delta_{\mathsf{q}}, \delta_r\rangle$ consists of a finite, nonempty set Q of states, an initial state $\mathsf{q}_I \in Q$, an input alphabet $2^{\mathcal{P}}$, an output alphabet $O \subset \mathbb{R}$, a deterministic transition function $\delta_{\mathsf{q}}\colon Q \times 2^{\mathcal{P}} \to Q$, and an output function $\delta_r\colon Q \times 2^{\mathcal{P}} \to O$.*

Reward machines encode non-Markovian reward functions. The run $\mathsf{q}_0(\ell_1, r_1)\mathsf{q}_1(\ell_2, r_2) \ldots (\ell_k, r_k)\mathsf{q}_{k+1}$ of a reward machine $\mathcal{R}$ on a label sequence $\ell_1 \ldots \ell_k \in (2^{\mathcal{P}})^*$ is a sequence of states and label-reward pairs such that $\mathsf{q}_0 = \mathsf{q}_I$, $\delta_{\mathsf{q}}(\mathsf{q}_i, \ell_i) = \mathsf{q}_{i+1}$ and $\delta_r(\mathsf{q}_i, \ell_i) = r_i$ for all $i \in \{0, \ldots, k\}$. The reward machine $\mathcal{R}$ produces a sequence of rewards from an input label sequence as $\mathcal{R}(\ell_1 \ldots \ell_k) = r_1 \ldots r_k$. We say that a reward machine $\mathcal{R}$ *implements* the reward function R of an MDP if for every trajectory $s_0a_0 \ldots s_ka_ks_{k+1}$ and the corresponding label sequence $\ell_1 \ldots \ell_k$, the reward sequence equals $\mathcal{R}(\ell_1 \ldots \ell_k)$.

The reward machine of the two-door environment is in Figure 3. Every node is a state in the reward machine. An edge $(\mathsf{q}_i, \mathsf{q}_j)$ with tuple (ρ, r) indicates that given a label $\ell \in 2^{\mathcal{P}}$ satisfying propositional formula ρ, the transition from state q_i to q_j yields reward $r = \delta_r(\mathsf{q}_i, \ell)$. For instance, the transition $(\mathsf{q}_1, \mathsf{q}_5)$ with $(w \vee d2, 0)$ in Figure 3 occurs if a label ℓ satisfies $\rho = w \vee d2$, i.e., if the agent moves into a wall w or passes the second door $d2$, yielding a reward of 0. The agent receives rewards of 1, 2, 3, and 4 upon completing the four subtasks, respectively. We note that there can be multiple reward machines that encode an MDP's reward function, and reward machines may differ on a label sequence that does not correspond to any trajectory of the MDP.

3.2 CONTEXTUAL MDPS

Definition 3. *A contextual Markov decision process (CMDP) Hallak et al. [2015] $\bar{\mathcal{M}} = \langle S, A, \mathcal{C}, \mathsf{M}\rangle$ is defined by a state space S, an action space A, a context space $\mathcal{C}$ and a mapping M from $\mathcal{C}$ to MDP parameters. $\bar{\mathcal{M}}$ represents a family of MDPs parameterized by contexts $c \in \mathcal{C} \subseteq \mathbb{R}^n$, $n \in \mathbb{Z}^+$. An MDP in this family is a tuple $\mathsf{M}(c) = \langle S, A, p_c, R_c, \phi_c, \gamma\rangle$ that shares the same state space S and action space A with other members, but its probabilistic transition function $p_c : S \times A \times S \to [0, 1]$, reward function $R_c : S \times A \to [0, 1]$ and initial state distribution $\phi_c : S \to [0, 1]$ depend on c.*

CMDPs appear in the multi-task RL literature to model tasks, where transition functions, reward functions, and initial state distributions are parameterized via contexts. Although the two-door environment in Figure 2 has a context that determines the door positions, which in return affects the transition and reward functions, the CMDP framework fails to model such an environment as the two-door environment has a non-Markovian reward function. We present a new MDP formulation to address this limitation in Section 4.

Given a CMDP $\bar{\mathcal{M}}$, *contextual RL* Hallak et al. [2015] aims to learn a policy that maximizes the expectation of the value of an initial state in context c sampled from a target context distribution $\varphi : \mathcal{C} \rightarrow [0,1]$, namely, $\max_\omega \mathbb{E}_{\varphi(c),\phi_c(s)}[V_\omega(s,c)]$, where $V_\omega(s,c)$ is the value of state s in context c and encodes the expected discounted return obtained by following the policy $\pi_\omega(a|s,c)$ as $V_\omega(s,c) = \mathbb{E}_{\pi_\omega(a|s,c)}[R_c(s,a) + \gamma \mathbb{E}_{p_c(s'|s,c)}[V_\omega(s',c)]]$.

3.3 SELF-PACED REINFORCEMENT LEARNING

Klink et al. [2020a,b, 2021] propose algorithms under the *Self-paced RL* framework to address the contextual RL problem. *Self-paced RL* aims to iteratively generate a sequence of context distributions by maximizing the expected performance with respect to the current distribution, which is regularized by the KL divergence from the target context distribution, namely,

$$\begin{aligned} \max_{\nu,\omega} \quad & \mathbb{E}_{\varrho(c|\nu),\phi_c(s)}[V_\omega(s,c)] - \alpha D_{\mathrm{KL}}(\varrho(c|\nu) \,||\, \varphi(c)) \\ \text{s.t.} \quad & D_{\mathrm{KL}}(\varrho(c|\nu) \,||\, \varrho(c|\nu_{prev})) \leq \epsilon, \end{aligned} \tag{1}$$

where $\varrho(c|\nu)$ and α are the current context distribution parameterized by ν and the regularization coefficient, respectively. Klink et al. [2020a] introduce the constraint in (1) to restrict the divergence of the current context distribution from the previous context distribution parameterized by ν_{prev}. Klink et al. [2020b] propose a way to estimate the expectation in (1). By following policy π_ω, they collect a set $\mathcal{D} = \{(c_i, \tau_i) | c_i \sim \varrho(c|\nu_{prev}), i \in \{1,2,\cdots,M\}\}$ of trajectories $\tau_i = (s_{i,0}, a_{i,0}, r_{i,1}, s_{i,1}), \cdots, (s_{i,T_i-1}, a_{i,T_i-1}, r_{i,T_i}, s_{i,T_i})$, where $r_{i,t+1} = R_{c_i}(s_{i,t}, a_{i,t}, s_{i,t+1})$ is the reward received at time $t+1$ in trajectory τ_i. Then, they use the cumulative sum of discounted rewards collected in trajectories to obtain an unbiased estimator of the expectation as

$$\frac{1}{N} \sum_{i=1}^{N} \frac{\varrho(c_i|\nu)}{\varrho(c_i|\nu_{prev})} \sum_{t=0}^{T_i-1} \gamma^t r_{i,t+1}, \tag{2}$$

where $\frac{\varrho(c_i|\nu)}{\varrho(c_i|\nu_{prev})}$ is an importance weight used to estimate the value of state $s_{i,0}$ in context c_i with respect to the current context distribution $\varrho(\cdot|\nu)$, as c_i is sampled from $\varrho(\cdot|\nu_{prev})$.

4 PROBLEM FORMULATION

We begin with integrating a labeling function into a CMDP to propose a labeled CMDP. We use labeled CMDPs to model long-horizon planning tasks, described via contexts.

Definition 4. *A labeled CMDP $\bar{\mathcal{M}}^L = \langle S, A, \mathcal{C}, \mathsf{M}^L \rangle$ consists of a CMDP $\bar{\mathcal{M}}$ and a labeling function $L_c : S \times A \times S \rightarrow 2^{\mathcal{P}}$. A member of a labeled CMDP $\bar{\mathcal{M}}^L$ is a labeled MDP $\mathsf{M}^L(c) = \langle S, A, p_c, R_c^L, \phi_c, \gamma, \mathcal{P}, L_c \rangle$ parameterized by a context $c \in \mathcal{C}$.*

A labeled MDP $\mathsf{M}^L(c)$ differs from a labeled MDP $\mathcal{M}$, from Definition 1, as the former depends on a context $c \in \mathcal{C}$. However, every labeled MDP $\mathsf{M}^L(c)$ obtained in $\bar{\mathcal{M}}^L$ can use the same reward machine, that encodes the underlying task structure. Throughout this paper, we make Assumption 1 on the context space $\mathcal{C}$ of a labeled CMDP $\bar{\mathcal{M}}^L$.

Assumption 1. *There exists $\Gamma \in \mathbb{Z}^+$ and $\mathcal{C}[1], \cdots, \mathcal{C}[\Gamma]$ such that $\mathcal{C} = \prod_{i=1}^{\Gamma} \mathcal{C}[i]$. For $c = (c[1], \cdots, c[\Gamma]) \in \mathcal{C}$, we call $c[i]$ the i^{th} context parameter of c. We say Γ is the dimension of the context space $\mathcal{C}$, referred to as $dim(\mathcal{C})$.*

The assumption of a box-shaped context space $\mathcal{C}$ of a labeled CMDP $\bar{\mathcal{M}}^L$ allows us to establish a mapping from the transitions in the reward machine to context parameters.

Problem statement. Given a labeled CMDP $\bar{\mathcal{M}}^L$, a reward machine $\mathcal{R}$ that encodes the non-Markovian reward function of $\bar{\mathcal{M}}^L$, and a target context distribution φ, we want to obtain a policy that maximizes the expected discounted return in contexts c drawn from φ, namely,

$$\max_\omega \mathbb{E}_{\varphi(c),\phi_c(s),\pi_\omega(a|s,c)}[\sum_{t=0}^{T-1} \gamma^t R_c^L(h_t)], \tag{3}$$

where $h_t = s_0 a_0 \cdots s_t a_t s_{t+1}$ is the history at time t. Note that as the reward machine $\mathcal{R}$ encodes the reward function R_c^L for any context c, we have $R_c^L(h_t) = \mathcal{R}(\ell_1 \cdots \ell_{t+1})$ with labels $\ell_\tau = L_c(s_{\tau-1}, a_{\tau-1}, s_\tau)$ for $\tau \in [t+1]$.

5 METHOD

We first present an intermediate self-paced RL algorithm by adopting the approach by Icarte et al. [2018a], which runs an RL algorithm using reward machines. Then, we discuss how contexts affect the transitions in a reward machine, and define a reward-machine-context mapping. Finally, integrating the proposed mapping into the intermediate algorithm, we develop a reward-machine-guided self-paced RL algorithm.

5.1 INTERMEDIATE SELF-PACED RL

We construct a product contextual MDP that combines a labeled contextual MDP $\bar{\mathcal{M}}^L$ and its reward machine $\mathcal{R}$.

Definition 5. *Given a labeled contextual Markov decision process $\bar{\mathcal{M}}^L$ and a reward machine $\mathcal{R}$, we define a product contextual MDP as the tuple $\bar{\mathcal{M}}^L_{\mathcal{R}} = \langle \bar{S}, A, \mathcal{C}, \bar{\mathsf{M}}^L \rangle$*

that has an extended state space $\bar{S} = S \times Q$, an action space A, a context space $\mathcal{C}$, and a mapping $\bar{\mathsf{M}}^L$ from the context space to product MDP parameters. A member of this product contextual MDP is a tuple $\bar{\mathsf{M}}^L(c) = \langle \bar{S}, A, \bar{p}_c, \bar{R}_c^L, \bar{\phi}_c, \gamma, \mathcal{P}, L_c \rangle$ with a probabilistic transition function $\bar{p}_c : \bar{S} \times A \times \bar{S} \to [0,1]$, a reward function $\bar{R}_c^L : \bar{S} \times A \times \bar{S} \to \mathbb{R}$, and an initial state distribution $\bar{\phi}_c : S \times \{\mathsf{q}_I\} \to [0,1]$. We define them as

$$\bar{p}_c((s,\mathsf{q}), a, (s', \mathsf{q}')) = \begin{cases} p_c(s,a,s') & \text{if } \mathsf{q}' = \delta_{\mathsf{q}}(\mathsf{q}, L_c(s,a,s')); \\ 0 & \text{otherwise}, \end{cases} \tag{4}$$

$$\bar{R}_c^L((s,\mathsf{q}), a, (s',\mathsf{q}')) = \delta_r(\mathsf{q}, L_c(s,a,s'))), \tag{5}$$

$$\bar{\phi}_c(s, \mathsf{q}_I) = \phi_c(s), \tag{6}$$

where states $s, s' \in S$ and $\mathsf{q}, \mathsf{q}' \in Q$ come from labeled contextual MDP $\bar{\mathcal{M}}^L$ and reward machine $\mathcal{R}$, respectively.

A trajectory of length T on the product MDP $\bar{\mathsf{M}}^L(c)$ is $\bar{\tau}_i = (\bar{s}_0, a_0, \bar{r}_1, \bar{s}_1), \cdots, (\bar{s}_{T-1}, a_{T-1}, \bar{r}_{i,T}, \bar{s}_T)$, where $\bar{r}_t = \bar{R}_c^L(\bar{s}_{t-1}, a_{t-1}, \bar{s}_t)$, $t \in \{1, 2, \cdots, T\}$. The intermediate self-paced RL algorithm replaces the contextual MDP trajectories with the product contextual MDP trajectories. Therefore, an RL agent can capture the temporal task structure by learning a policy via trajectories rolled out in a product contextual MDP. The intermediate self-paced RL algorithm optimizes the following objective to generate context distribution

$$\begin{aligned} \max_{\nu_k} \quad & \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \gamma^t \frac{\varrho(c_i|\nu_k)}{\varrho(c_i|\nu_{k-1})} \bar{r}_{i,t+1} \\ & - \alpha_k D_{\mathrm{KL}}(\varrho(c|\nu_k) \,||\, \varphi(c)) \\ \text{s.t.} \quad & D_{\mathrm{KL}}(\varrho(c|\nu_k) \,||\, \varrho(c|\nu_{k-1})) \leq \epsilon, \end{aligned} \tag{7}$$

where α_k is the regularization coefficient at the context distribution update k. Appendix B provides the pseudocode for this algorithm.

5.2 FROM REWARD MACHINES TO CONTEXTS

In the two-door environment (see Figure 2), we observe that the first context parameter, i.e., the position of the first door, determines which $\bar{\mathcal{M}}_{\mathcal{R}}^L$ transitions enable the agent to pass the first door, yielding label $\{d1\}$. If we change the value of the first context parameter, then we have a different set of $\bar{\mathcal{M}}_{\mathcal{R}}^L$ transitions that yield label $\{d1\}$. However, this modification has no impact on the transitions that enable the agent to pass the second door, i.e., to obtain label $\{d2\}$. Taking this observation into account, we show how context parameters influence the transitions in a reward machine, then we define *reward machine-context mapping* $\mathsf{F} : Q \times Q \to 2^D$, which outputs the smallest set of context parameters that determines if a transition in the reward machine happens.

Definition 6. *Given a product contextual MDP $\bar{\mathcal{M}}_{\mathcal{R}}^L$, we define a set $\mathcal{G} \subseteq D = \{1, 2, \cdots, dim(\mathcal{C})\}$, as* the set of identifier context parameters on a transition (q, s, a, s') *if*

$$\forall c, c' \in \mathcal{C}, c[i] = c'[i], \forall i \in \mathcal{G} \implies \delta_{\mathsf{q}}(\mathsf{q}, L_c(s,a,s')) = \delta_{\mathsf{q}}(\mathsf{q}, L_{c'}(s,a,s')), \tag{8}$$

where $(\mathsf{q}, s, a, s') \in Q \times S \times A \times S$. That is, $\mathcal{G}$ is the set of indices of the context parameters that identify the next state of the reward machine given a state q of the reward machine and a transition (s, a, s') in the labeled MDP.

Notice that D is a set of identifier context parameters for all $(\mathsf{q}, s, a, s') \in Q \times S \times A \times S$.

Lemma 1. *If $\mathcal{G}_1$ is a set of identifier context parameters on (q, s, a, s'), and $\mathcal{G}_1 \subseteq \mathcal{G}_2 \subseteq D$, then $\mathcal{G}_2$ is also a set of identifier context parameters on (q, s, a, s').*

Proof. Suppose $c[i] = c'[i]$, $\forall i \in \mathcal{G}_2$, then $c[i] = c'[i]$, $\forall i \in \mathcal{G}_1$ by definition. □

We note that if the empty set $\emptyset$ is a set of identifier context parameters on (q, s, a, s'), then the corresponding transition $\delta_{\mathsf{q}}(\mathsf{q}, L_c(s,a,s'))$ in the reward machine does not depend on the choice of context $c \in \mathcal{C}$.

Theorem 2. *Under Assumption 1, $\mathcal{G}_1$ and $\mathcal{G}_2$ are sets of identifier context parameters on (q, s, a, s') if and only if $\mathcal{G}_1 \cap \mathcal{G}_2$ is a set of identifier context parameters on (q, s, a, s').*

Proof Sketch. [1] The backward statement comes from Lemma 1. For the forward statement, let $c_{\mathcal{G}} = [c[i]]_{i \in \mathcal{G}}$. Then, for any $c, c' \in \mathcal{C}$ that satisfy $c_{\mathcal{G}_1 \cap \mathcal{G}_2} = c'_{\mathcal{G}_1 \cap \mathcal{G}_2}$, by Assumption 1 there exists c'' for which $c''_{\mathcal{G}_1} = c_{\mathcal{G}_1}$ and $c''_{\mathcal{G}_2} = c'_{\mathcal{G}_2}$. Therefore, $\delta_{\mathsf{q}}(\mathsf{q}, L_c(s,a,s')) = \delta_{\mathsf{q}}(\mathsf{q}, L_{c''}(s,a,s')) = \delta_{\mathsf{q}}(\mathsf{q}, L_{c'}(s,a,s'))$. □

Corollary 1. *Under Assumption 1, the set Γ containing all sets of identifier context parameters on (q, s, a, s') is closed under arbitrary unions and finite intersections.*

Proof. Lemma 1 and Theorem 2 guarantee that Γ is closed under unions and finite intersections, respectively. □

Corollary 1 guarantees that there is a set of identifier context parameters that is contained by every set of identifier context parameters. In Definition 7, we define a mapping that provides such a set for any transition (q, s, a, s').

Definition 7. *Given a product contextual MDP $\bar{\mathcal{M}}_{\mathcal{R}}^L$, we define a mapping $\mathsf{H}_{min} : Q \times S \times A \times S \to 2^D$ such that we call $\mathsf{H}_{min}(\mathsf{q}, s, a, s')$ "the smallest set of identifier context parameters on (q, s, a, s')" if $\mathsf{H}_{min}(\mathsf{q}, s, a, s') =$*

[1] See Appendix A for the complete proof.

$\bigcap_{\mathcal{G}_i \in \Gamma} \mathcal{G}_i$, *where* Γ *is the set containing all possible sets of identifier context parameters on* (q, s, a, s').

For practical applications, the design of H_{min} is not trivial, as one needs to separately analyze every transition in a labeled contextual MDP $\bar{\mathcal{M}}^L$. On the contrary, it is trivial to work with the transitions in a reward machine $\mathcal{R}$, as the number of transitions in $\mathcal{R}$ is smaller than the number of transitions in $\bar{\mathcal{M}}^L$ in general. Therefore, we define a set of identifier context parameters for every transition in $\mathcal{R}$.

Definition 8. *Given a product contextual MDP* $\bar{\mathcal{M}}^L_{\mathcal{R}}$ *and the mapping* H_{min}, *we define a reward machine-context mapping* $\mathsf{F} : Q \times Q \to 2^D$ *that outputs "a set of identifier context parameters for the transition* $(\mathsf{q}, \mathsf{q}')$*" as*

$$\mathsf{F}(\mathsf{q}, \mathsf{q}') = \bigcup_{\mathcal{B}(\mathsf{q},\mathsf{q}')} \mathsf{H}_{min}(\mathsf{q}, s, a, s'), \tag{9}$$

where $\mathcal{B}(\mathsf{q}, \mathsf{q}') = \{(\mathsf{q}, s, a, s') \in Q \times S \times A \times S \,|\, \delta_{\mathsf{q}}(\mathsf{q}, L_c(s, a, s')) = \mathsf{q}'$ *for some* $c \in \mathcal{C}\}$.

Theorem 3. $\mathsf{F}(\mathsf{q}, \mathsf{q}')$ *is the smallest set that is a set of identifier context parameters for all* $(\mathsf{q}, s, a, s') \in \mathcal{B}(\mathsf{q}, \mathsf{q}')$.

Proof. By Corollary 1, $\mathsf{F}(\mathsf{q}, \mathsf{q}')$ is a set of identifier context parameters for all $(\mathsf{q}, s, a, s') \in \mathcal{B}(\mathsf{q}, \mathsf{q}')$. Also, a set $\mathcal{U}$ that is guaranteed to be a set of identifier context parameters for all $(\mathsf{q}, s, a, s') \in \mathcal{B}(\mathsf{q}, \mathsf{q}')$ must contain $\mathsf{F}(\mathsf{q}, \mathsf{q}')$ by construction. Then, $|\mathcal{U}| \geq |\mathsf{F}(\mathsf{q}, \mathsf{q}')|$. □

An expert designs mapping F by asking questions about the task structure. For instance, for the transition $(\mathsf{q}_1, \mathsf{q}_5)$ in Figure 3, the expert should ask: Is there a transition (s, a, s') in the labeled contextual MDP $\bar{\mathcal{M}}^L$ such that it causes the agent to hit the wall for some context c but lets the agent pass through the door, i.e., $(\mathsf{q}_1, \mathsf{q}_2)$, for a different context c'? The idea is to find the context parameters $i \in D$ for which a change of value, e.g. $c[i] \neq c'[i]$, prevents a transition $(\mathsf{q}, \mathsf{q}')$ in the reward machine from happening. For $(\mathsf{q}_1, \mathsf{q}_5)$, the mapping outputs the first context parameter, $\mathsf{F}(\mathsf{q}_1, \mathsf{q}_5) = \{1\}$, as the identifier, since it determines the position of the first door. In short, when the agent is in the second room and moves into the first door/wall with an upward action, then the position of the first door determines whether it moves into the door or the wall. Here, the position of the second door does not identify which transition will happen.

5.3 REWARD-MACHINE-GUIDED, SELF-PACED REINFORCEMENT LEARNING

Klink et al. [2020b]'s self-paced RL algorithm uses an importance weight in (2) as the ratio between probabilities of a context with respect to the current and previous contexts distributions. In other words, the algorithm assumes that every context parameter has an effect on the reward that an environment interaction yields. On the other hand, by Theorem 3, a reward machine-context mapping F outputs the smallest set of identifier context parameters for a transition $(\mathsf{q}, \mathsf{q}')$ in the reward machine $\mathcal{R}$. Therefore, we can remove the naive assumption of Klink et al. [2020b] and use the context parameters that the mapping provides to compute the importance weight of a reward received in a transition $(\mathsf{q}, \mathsf{q}')$. We achieve this by utilizing the marginal context distributions for the context parameters in the set $\mathsf{F}(\mathsf{q}, \mathsf{q}')$ as $\frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T_i-1}\gamma^t \frac{\varrho_{\mathsf{f}_t}(c_i|\nu_k)}{\varrho_{\mathsf{f}_t}(c_i|\nu_{k-1})}\bar{r}_{i,t+1}$, where $\bar{r}_{i,t} = \bar{R}^L_{c_i}(\bar{s}_{i,t-1}, a_{i,t-1}, \bar{s}_{i,t})$ and $\mathsf{f}_t = \mathsf{F}(\mathsf{q}_t, \mathsf{q}_{t+1})$. Here, we introduce $\varrho_{\mathsf{f}_t}(\cdot|\nu_k)$ and $\varrho_{\mathsf{f}_t}(\cdot|\nu_{k-1})$, that are the current and previous marginal context distributions, where the marginal variables are the identifier context parameters in f_t, respectively. We note that for the case $\mathsf{f}_t = \emptyset$, we assign $\frac{\varrho_{\mathsf{f}_t}(c_i|\nu_k)}{\varrho_{\mathsf{f}_t}(c_i|\nu_{k-1})} = 1$. Consequently, the reward-machine-guided, self-paced RL algorithm optimizes the following objective for context distribution updates, namely,

$$\begin{aligned}\max_{\nu_k} \quad & \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T_i-1}\gamma^t \frac{\varrho_{\mathsf{f}_t}(c_i|\nu_k)}{\varrho_{\mathsf{f}_t}(c_i|\nu_{k-1})}\bar{r}_{i,t+1} \\ & - \alpha_k D_{\mathrm{KL}}(\varrho(c|\nu_k) \,||\, \varphi(c)) \\ \text{s.t.} \quad & D_{\mathrm{KL}}(\varrho(c|\nu_k) \,||\, \varrho(c|\nu_{k-1})) \leq \epsilon, \end{aligned} \tag{10}$$

where α_k is the regularization coefficient at the context distribution update k. Similar to the intermediate self-paced RL, the reward-machine-guided, self-paced RL algorithm runs on a product contextual MDP $\bar{\mathcal{M}}^L_{\mathcal{R}}$, as well. We outline the complete algorithm in Algorithm 1. Lines 3-5 update the policy π using trajectories in the sampled contexts via an RL algorithm Ψ. Line 6 generates context distributions.

6 EMPIRICAL RESULTS

We evaluate the proposed RM-guided SPRL and Intermediate SPRL with three state-of-the-art automated curriculum generation methods[2]: SPDL [Klink et al., 2020b], GoalGAN [Florensa et al., 2018], and ALP-GMM [Portelas et al., 2020]. We also include two baseline approaches: Default, which draws samples from the target context distribution without generating a curriculum, and Default*, which extends Default by running the RL algorithm on a product contextual MDP, hence we observe the effect of capturing temporal abstractions. Appendix B includes more details.

Two-door environment. The two-door environment is a variation of the point-mass environment Klink et al. [2020a,b, 2021, 2022] with a temporal structure. Similar to commonly studied domains such as the office world, craft world, and water world Icarte et al. [2018a], Camacho et al. [2019], Icarte et al. [2022], the two-door environment has discrete state and action spaces as a 40-by-40 grid

[2]See `https://github.com/cevahir-koprulu/rm-guided-sprl` to access the code repository of this work.

Algorithm 1 Reward-Machine-Guided, Self-Paced RL

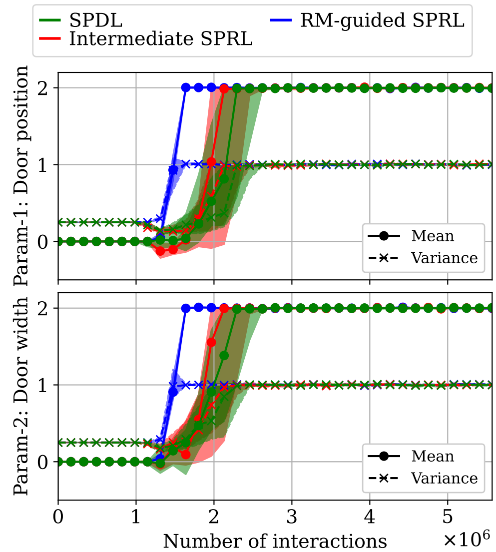
Input: Product MDP $\bar{\mathcal{M}}^L_{\mathcal{R}}$, reward machine-context mapping F, target context distribution φ, initial context distribution $\varrho(\cdot|\nu_0)$,
Parameter: KL penalty proportion ζ, relative entropy bound ϵ, KL penalty offset offset K_α, number K of iterations, number N of rollouts
Output: Final policy π_{ω_K}

Initialize policy π_{ω_0}.
for $k = 1$ to K **do**
  $c_i \sim \varrho(c|\nu_{k-1}), i \in [N]$, ▷ *sample contexts*
  $\mathcal{D}_k \leftarrow \{(c_i, \bar{\tau}_i)|\bar{\tau}_i = (\bar{s}_{i,0}, a_{i,0}, \bar{r}_{i,1}, \bar{s}_{i,1}), \cdots, (\bar{s}_{i,T_i-1}, a_{i,T_i-1}, \bar{r}_{i,T_i}, \bar{s}_{i,T_i}), i \in [N]\}$, ▷ *collect trajectories*
  $\pi_{\omega_k} \leftarrow \Psi(\mathcal{D}_k, \pi_{\omega_{k-1}})$ ▷ *update policy with RL algorithm* Ψ
  Compute next context distribution parameter ν_k by optimizing (10) with
$$\alpha_k = \begin{cases} 0 & \text{if } k \le K_\alpha; \\ \mathsf{B}(\nu_{k-1}, \mathcal{D}_k) & \text{otherwise,} \end{cases}$$
  where $\mathsf{B}(\nu_{k-1}, \mathcal{D}_k) = \zeta \frac{\max\left(0, \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T_i} \gamma^t \bar{r}_{i,t}\right)}{D_{\mathrm{KL}}(\varrho(c|\nu_{k-1})||\varphi(c))}$.
end for
return π_{ω_K}

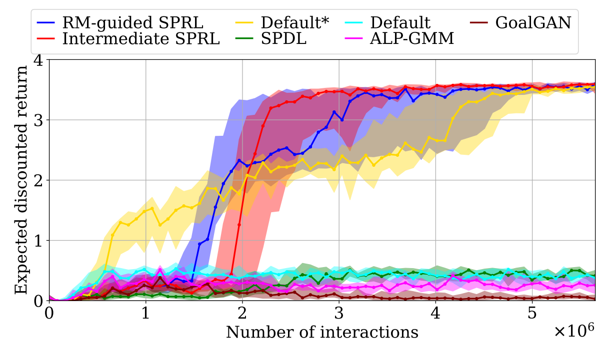
world. The context space $\mathcal{C} = [-4, 4]^2$ includes all available horizontal positions for two doors. The target context distribution $\varphi(c)$ is a normal distribution $\mathcal{N}(\mu, \Sigma)$, where $\mu = (\mu_1, \mu_2) = (2, 2)$ and $\Sigma = diag((\sigma_1^2, \sigma_2^2)) = \boldsymbol{I}_2$, where $\boldsymbol{I}_2$ is the identity matrix.

Figure 4a demonstrates how the curricula generated by each method evolve over the training. Here, we exclude GoalGAN, Default*, and Default, since they do not have a notion of a target context distribution. RM-guided SPRL produces sequences of context distributions that vary less over the same curriculum updates and converges faster, by one-fourth, than Intermediate SPRL and SPDL. Table 1 demonstrates the average variance of the statistics of the context distributions, i.e., mean and variance of normal distributions, generated until convergence to the target distribution. RM-guided SPRL has the lowest variances for all statistics, up-to four orders of magnitude, with statistical significance $p < 0.001$. Intuitively, guiding the curriculum generation process via a reward-machine-context mapping F allows for avoiding redundant exploration of the curriculum space.

Figure 4b shows the expected discounted return progression. Although Default* obtains a higher return early on, it lags behind RM-guided SPRL and Intermediate SPRL since it does not generate a curriculum, but samples from the target context distribution directly. As we set K_α to 70 for all self-paced algorithms, the agent draws easy contexts from the initial context distribution $\varrho(\cdot|\nu_0)$. RM-guided SPRL surpasses Default* quickly, but the agent seems to get stuck in the final phase, finding the goal. Intermediate SPRL do



(a) Progression of the statistics (mean and variance) of context distributions generated in the curriculum.



(b) Progression of the expected discounted return with respect to the target context distribution.

Figure 4: Two-door environment: Progression of the curricula and performance during the training. Shaded regions cover the quartiles. Bold lines indicate the median values.

not experience this as its curricula converge later (see red lines in Figure 4a). Other approaches cannot learn a policy that accomplishes the task because they do not capture the temporal structure described by the reward machine.

Customized Swimmer-v3 environment. We customize the Swimmer-v3 environment Brockman et al. [2016] by adding two flags, namely checkpoints, to the left, f_1, and the right, f_2, of the initial position of the swimmer. In comparison to the two-door environment, customized Swimmer-v3 has continuous state (8-dimensional) and action (2-dimensional) spaces. However, the underlying task is the simplest in our experimental setup, with a reward machine of 3 states (see Figure 5): The swimmer has to visit flag 2 f_2 first, and then flag 1, obtaining a reward of 100 and 1000, respectively. Inspired by Icarte et al. [2022], we use a control penalty, noted as CP, for rewards received by the agent

Table 1: Curricula variance: Average variance of the statistics of the context distributions generated by self-paced RL methods in three case studies. μ_i and σ_i^2 correspond to the statistics of a normal distribution, i.e., the mean and the variance for the i^{th} context parameter, respectively. The variances that are highlighted in bold are significantly better results (lower variance of a statistic) with $p < 0.001$.

	Stat	RM-guided	Intermediate	SPDL
Two-door	μ_1	$\mathbf{4.09 \cdot 10^{-3}}$	$2.06 \cdot 10^{-1}$	$1.21 \cdot 10^{-1}$
	μ_2	$\mathbf{4.61 \cdot 10^{-3}}$	$1.69 \cdot 10^{-1}$	$8,78 \cdot 10^{-2}$
	σ_1^2	$\mathbf{2.23 \cdot 10^{-3}}$	$3.75 \cdot 10^{-2}$	$2.46 \cdot 10^{-2}$
	σ_2^2	$\mathbf{2.31 \cdot 10^{-3}}$	$2.59 \cdot 10^{-2}$	$2.42 \cdot 10^{-2}$
Swimmer	μ_1	$3.02 \cdot 10^{-4}$	$8.18 \cdot 10^{-4}$	$2.37 \cdot 10^{-2}$
	μ_2	$5.07 \cdot 10^{-4}$	$1.13 \cdot 10^{-4}$	$2.90 \cdot 10^{-2}$
	σ_1^2	$2.83 \cdot 10^{-6}$	$9.55 \cdot 10^{-7}$	$8.40 \cdot 10^{-6}$
	σ_2^2	$2.68 \cdot 10^{-6}$	$2.77 \cdot 10^{-6}$	$6.16 \cdot 10^{-5}$
HalfCheetah	μ_1	$\mathbf{2.90 \cdot 10^{-2}}$	$6.23 \cdot 10^{-1}$	-
	μ_2	$\mathbf{2.39 \cdot 10^{-2}}$	$9.84 \cdot 10^{-1}$	-
	μ_3	$\mathbf{9.85 \cdot 10^{-2}}$	$1.27 \cdot 10^{-1}$	-
	σ_1^2	$\mathbf{9.13 \cdot 10^{-4}}$	$5.80 \cdot 10^{-3}$	-
	σ_2^2	$1.92 \cdot 10^{-3}$	$1.95 \cdot 10^{-4}$	-
	σ_3^2	$\mathbf{2.86 \cdot 10^{-3}}$	$5.35 \cdot 10^{-2}$	-

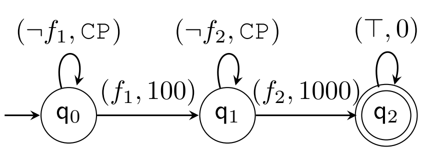


Figure 5: Reward machine of the Swimmer environment

following the self-loop transitions in the reward machine, to discourage the agent from applying large forces to the joints. The context space is 2-dimensional and determines the positions of the flags: $\mathcal{C}_2 = [-0.6, 0] \times [1, 1.6]$. The target context distribution is $\mathcal{N}((0.6, 1.6, \boldsymbol{I}_3 \cdot 1.6 \cdot 10^{-7})$.

Figure 6 shows that only RM-guided-SPRL and Default* achieve 100% success ratio in median. RM-guided SPRL converges faster, and is more reliable as the quartiles converge before the training ends, as well. Default*'s performance evidence that this task does not require a curriculum as much as the two-door environment. Intermediate SPRL's failure supports this argument, as it cannot achieve a success ratio of more than 20%, in the median. The other algorithms, again, fail to accomplish the task. Table 1 indicates that the curricula variance of RM-guided SPRL is not significantly different than Intermediate SPRL. Nevertheless, RM-guided SPRL is reliable as it is 100% successful (median) while avoiding the redundant exploration of the curriculum space.

Customized HalfCheetah-v3 environment. We also customize the HalfCheetah-v3 environment Brockman et al.

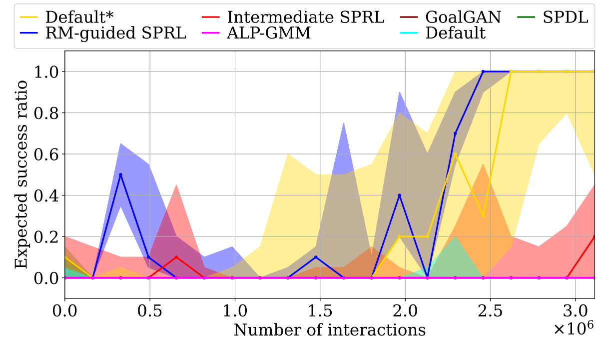


Figure 6: Swimmer-v3 environment: Progression of the successful episodes ratio in contexts drawn from the target context distribution over curriculum updates.

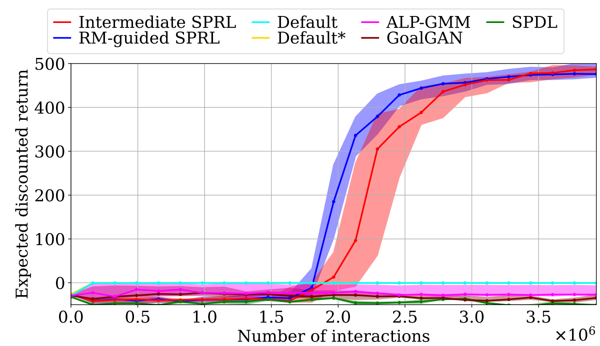


Figure 7: HalfCheetah-v3 environment: Progression of the expected discounted return with respect to the target context distribution over curriculum updates.

[2016] by adding three flags, f_1, f_2, and f_3, ordered in ascending distance to the right of the cheetah's initial position. The reward machine in Figure 8 describes the following task: The cheetah has to visit flags 1 and 2, then go back to flag 1, and finally pass flag 3. The underlying task requires the cheetah to change direction 3 times. The original HalfCheetah-v3 Brockman et al. [2016] and its variation in Icarte et al. [2022] are single-task environments and reward the cheetah for running forward, only. In comparison, customized HalfCheetah-v3 has a running backward subtask (the transition (q_2, q_3) in the reward machine in Figure Figure 8), which is challenging for the agent. Similar to customized Swimmer-v3, customized HalfCheetah-v3 has continuous state (17-dimensional) and action (6-dimensional) spaces. The 3-dimensional context space determines the flag positions: $\mathcal{C}_3 = [0.5, 4] \times [2, 7] \times [3.5, 10]$. The target context distribution is $\mathcal{N}((4, 7, 10), \boldsymbol{I}_3 \cdot 1.6 \cdot 10^{-7})$.

Figure 7 shows that RM-guided SPRL is the only algorithm that can learn a policy that accomplishes the target contexts in every independent training run. Intermediate SPRL fails in one run, where the generated curricula cannot converge to the target context distribution during the training (see Appendix B). SPDL suffers from a similar issue because they

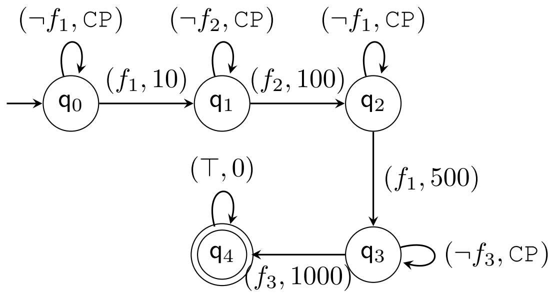
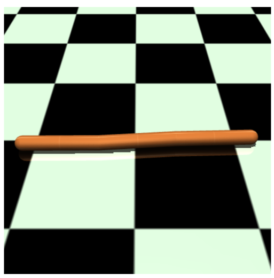
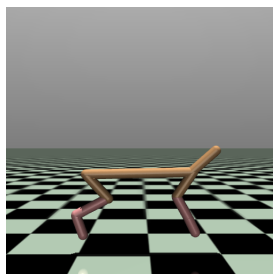


Figure 8: Reward machine of the HalfCheetah environment



(a) Swimmer-v3



(b) HalfCheetah-v3

Figure 9: Images from Customized Swimmer-v3 and Customized HalfCheetah-v3 environments.

both obtain a negative expected discounted return, which sets α_k in (10) to zero, hence the generated curricula do not approach, even diverge from, the target context distribution. Default*, Default, GoalGAN, and ALP-GMM are unsuccessful in this domain. In Table 1, we exclude SPDL, as none of its curricula converge to the target before the training ends. Similar to Case-2, RM-guided SPRL generates curricula whose variance is significantly smaller with $p < 0.001$.

7 CONCLUSIONS

We propose two self-paced RL algorithms that exploit the high-level structural knowledge about long-horizon planning tasks via reward machines. First, we present an intermediate self-paced RL algorithm that uses reward machines to update the policy and value functions of an RL agent. Then, we establish a mapping, called reward-machine-context mapping, that, given a transition in the reward machine, outputs the smallest set of identifier context parameters that determines whether the transition occurs or not. Lastly, we develop a reward-machine-guided, self-paced RL algorithm that builds on the intermediate algorithm and navigates the automated curriculum generation via reward-machine-context mapping. We evaluate the proposed algorithms in three domains. We empirically show that existing approaches fail to accomplish the given long-horizon planning tasks, whereas the proposed algorithms can capture the temporal structure of such tasks. Compared to the intermediate algorithm, the reward-machine-guided, self-paced RL algorithm is more reliable, as it achieves successful completion of the task in every use case, and it also reduces curricula variance by up to four orders of magnitude.

Limitations. The limitations come from the self-paced RL algorithm used in the proposed approaches, the assumption of a priori available reward machine, and task knowledge to construct a reward-machine-context mapping: 1) Intermediate SPRL and RM-guided SPRL employ a self-paced RL algorithm, SPDL Klink et al. [2020b], which uses a parametric family of context distributions to generate a curriculum. Similar to Klink et al. [2020b, 2021], we study Gaussian context distributions. Hence, SPDL does not address settings with arbitrary target context distributions. 2) We focus on long-horizon planning tasks with a priori available reward machines. The proposed approaches require a reward machine to construct a product contextual MDP, which captures the temporal task structure. 3) Task knowledge about the connection between the reward machine and the context space enables the design of a reward-machine-context mapping. Unless such knowledge is available, RM-guided SPRL and Intermediate SPRL become equivalent, as the latter do not utilize the mapping.

Future Work. Taking into account the limitations of the proposed approaches, we will study how to infer a reward machine and a reward-machine-context mapping of a domain online to remove the need for a priori available task knowledge. In addition, we will extend RM-guided SPRL to address arbitrary context distributions, which Klink et al. [2022] studies without integrating high-level structural task knowledge.

Acknowledgements

This work is supported by the Office of Naval Research (ONR) under grant number N00014-22-1-2254, and the National Science Foundation (NSF) under grant number 1646522.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *NeurIPS*, 2017.

Fahiem Bacchus, Craig Boutilier, and Adam Grove. Rewarding behaviors. In *National Conference on Artificial Intelligence*, pages 1160–1167, 1996.

Adrien Baranes and Pierre-Yves Oudeyer. Intrinsically motivated goal exploration for active motor learning in robots: A case study. In *IROS*, pages 1766–1773, 2010.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, pages 41–48, 2009.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

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*, pages 6065–6073, 2019.

Jiayu Chen, Yuanxin Zhang, Yuanfan Xu, Huimin Ma, Huazhong Yang, Jiaming Song, Yu Wang, and Yi Wu. Variational automatic curriculum learning for sparse-reward cooperative multi-agent problems. In *NeurIPS*, 2021.

Thomas G Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *JAIR*, pages 227–303, 2000.

Theresa Eimer, André Biedenkapp, Frank Hutter, and Marius Lindauer. Self-paced context evaluation for contextual reinforcement learning. In *ICML*, pages 2948–2958, 2021.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse Curriculum Generation for Reinforcement Learning. In *CoRL*, pages 482–495. PMLR, 2017.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *ICML*, pages 1515–1528, 2018.

Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *ICML*, pages 2107–2116, 2018a.

Rodrigo Toro Icarte, Toryn Q Klassen, Richard Valenzano, and Sheila A McIlraith. Teaching multiple tasks to an rl agent using ltl. In *AAMAS*, pages 452–461, 2018b.

Rodrigo Toro Icarte, Toryn Q Klassen, Richard Valenzano, and Sheila A McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *JAIR*, pages 173–208, 2022.

Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *AAAI*, 2015.

Minqi Jiang, Michael Dennis, Jack Parker-Holder, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Replay-guided adversarial environment design. *NeurIPS*, pages 1884–1897, 2021a.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. In *ICML*, pages 4940–4950. PMLR, 2021b.

Pascal Klink, Hany Abdulsamad, Boris Belousov, and Jan Peters. Self-paced contextual reinforcement learning. In *CoRL*, pages 513–529, 2020a.

Pascal Klink, Carlo D' Eramo, Jan R Peters, and Joni Pajarinen. Self-paced deep reinforcement learning. In *NeurIPS*, pages 9216–9227, 2020b.

Pascal Klink, Hany Abdulsamad, Boris Belousov, Carlo D'Eramo, Jan Peters, and Joni Pajarinen. A probabilistic interpretation of self-paced learning with applications to reinforcement learning. *JMLR*, 22:182:1–182:52, 2021.

Pascal Klink, Haoyi Yang, Carlo D'Eramo, Jan Peters, and Joni Pajarinen. Curriculum reinforcement learning via constrained optimal transport. In *ICML*, pages 11341–11358, 2022.

Cevahir Koprulu, Thiago D. Simão, Nils Jansen, and Ufuk Topcu. Risk-aware curriculum generation for heavy-tailed task distributions. In *UAI*, 2023.

M. Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *NeurIPS*, 2010.

Yen-Ling Kuo, Boris Katz, and Andrei Barbu. Encoding formulas as deep networks: Reinforcement learning for zero-shot execution of ltl formulas. In *IROS*, pages 5604–5610, 2020.

Xiao Li, Cristian-Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards. In *IROS*, pages 3834–3839, 2017.

Michael L Littman, Ufuk Topcu, Jie Fu, Charles Isbell, Min Wen, and James MacGlashan. Environment-independent task specifications via gltl. *arXiv preprint arXiv:1704.04341*, 2017.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *JMLR*, pages 1–50, 2020.

Ronald Parr and Stuart Russell. Reinforcement learning with hierarchies of machines. *NeurIPS*, 1997.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *CoRL*, pages 835–853, 2020.

Sebastien Racaniere, Andrew K Lampinen, Adam Santoro, David P Reichert, Vlad Firoiu, and Timothy P Lillicrap. Automated curricula through setter-solver interactions. In *ICLR*, 2020.

Zhipeng Ren, Daoyi Dong, Huaxiong Li, and Chunlin Chen. Self-paced prioritized curriculum learning with coverage penalty in deep reinforcement learning. *IEEE TNNLS*, pages 2216–2226, 2018.

Satinder P Singh. Reinforcement learning with a hierarchy of abstract models. In *National Conference on Artificial Intelligence*, pages 202–207, 1992.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Maxwell Svetlik, Matteo Leonetti, Jivko Sinapov, Rishi Shah, Nick Walker, and Peter Stone. Automatic curriculum graph generation for reinforcement learning agents. In *AAAI*, 2017.

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 *AAAI*, 2021.

Zhe Xu and Ufuk Topcu. Transfer of temporal logic formulas in reinforcement learning. In *IJCAI*, page 4010, 2019.

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 *ICAPS*, pages 590–598, 2020.

Yunzhi Zhang, Pieter Abbeel, and Lerrel Pinto. Automatic curriculum learning through value disagreement. In *NeurIPS*, pages 7648–7659, 2020.

Xuejing Zheng, Chao Yu, and Minjie Zhang. Lifelong reinforcement learning with temporal logic formulas and reward machines. *Knowledge-Based Systems*, page 109650, 2022.