Efficient Reinforcement Learning in Probabilistic Reward Machines

Xiaofeng Lin, Xuezhou Zhang

Boston University
{xfl,xuezhouz}@bu.edu

Abstract

In this paper, we study reinforcement learning in Markov Decision Processes with Probabilistic Reward Machines (PRMs), a form of non-Markovian reward commonly found in robotics tasks. We design an algorithm for PRMs that achieves a regret bound of $\widetilde{O}(\sqrt{HOAT} + H^2O^2A^{3/2} + H\sqrt{T})$, where H is the time horizon, O is the number of observations, A is the number of actions, and T is the number of time-steps. This result improves over the best-known bound, $\widetilde{O}(H\sqrt{OAT})$ of Bourel et al. (2023) for MDPs with Deterministic Reward Machines (DRMs), a special case of PRMs. When $T \geq H^3O^3A^2$ and $OA \geq H$, our regret bound leads to a regret of $\widetilde{O}(\sqrt{HOAT})$, which matches the established lower bound of $\Omega(\sqrt{HOAT})$ for MDPs with DRMs up to a logarithmic factor. To the best of our knowledge, this is the first efficient algorithm for PRMs. Additionally, we present a new simulation lemma for non-Markovian rewards, which enables reward-free exploration for any non-Markovian reward given access to an approximate planner. Complementing our theoretical findings, we show through extensive experiment evaluations that our algorithm indeed outperforms prior methods in various PRM environments.

Code — https://github.com/XiaofengLin7/UCBVI_PRM

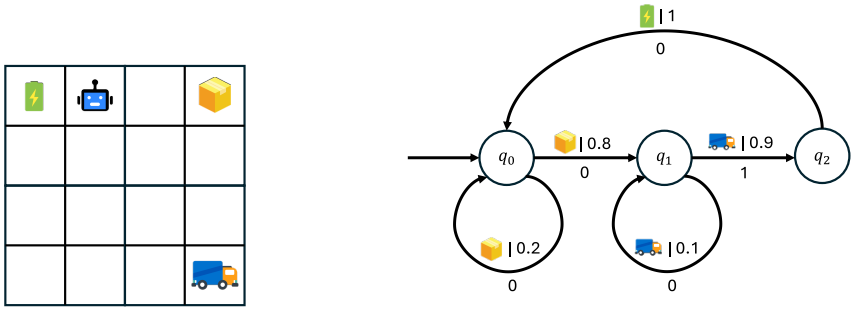
1 Introduction

Reinforcement learning traditionally focuses on the setting where the reward function is Markovian, meaning that it depends solely on the current state and action, and independent of history. However, in many real-world scenarios, the reward is a function of the history of states and actions. For example, consider a robot tasked with patrolling various locations in an industrial park. The performance of robot is measured by how thorough it regularly covers different zones in the park, which cannot easily be represented as a function of its current state and action, but rather would depend on its whole trajectory during the patrol.

One emerging tool to model such performance metrics is called the Reward Machine (RM)(Icarte et al. 2018, 2022), which is a Deterministic Finite-State Automaton (DFA) that can compress the sequence of past events into one single state. Combined with the current observation, the state of RM can fully specify the reward function. Hence, for an MDP with RM, we can obtain an equivalent cross-product MDP by leveraging the information of RM(see Lemma 1) and applying off-the-shelf RL algorithms *e.g.*, Q-learning of Sutton and Barto (2018) to learn an optimal policy. However, as we shall see later, this naive approach will incur a large regret.

One limitation of the classic RM framework is that the transition between the state of RM is restricted to be deterministic, whereas stochastic transitions are much more common in practice, especially with uncertainty in the environment. For instance, suppose a robot working in a warehouse is tasked with managing a warehouse by performing simple tasks of fetching and delivering items (as shown in Figure 1). The robot starts at a charging station, navigates to the item pickup location, collects the item, and then proceeds to the delivery location to deliver the item and receives a reward. Based on past experience and pre-collected data: there is a 20 percent chance that the item at the pickup location is not ready, requiring the robot to wait until the item is ready, and a 10 percent chance that the delivery location is occupied, causing the robot to wait before delivering the item. The robot is rewarded only when it successfully collects and delivers the item in sequence. The setting where the rewards can exhibit non-Markovian and stochastic dynamics can be represented as Probabilistic Reward Machine(PRM)(Dohmen et al. 2022).

In this paper, we investigate RL in Markov decision processes with probabilistic reward machines. We formalize the regret minimization problem within the episodic MDP with PRM setting and introduce an algorithm, `UCBVI-PRM`, a UCB-style model-based RL algorithm with novel steps specifically tailored to PRMs. Our algorithm achieves a regret bound of $\widetilde{O}(\sqrt{HOAT})$ that matches the established lower bound of $\Omega(\sqrt{HOAT})$ for MDPs with PRMs up to a logarithmic factor. Additionally, we present a new simulation lemma that characterizes the difference in policy evaluations between two MDPs with generic non-Markovian rewards. Based on the lemma, we design a reward-free exploration algorithms that can collect data with sufficient coverage to learn a near-optimal policy under any non-Markovian reward in downstream tasks. Finally, we conduct experiments to showcase the efficiency of `UCBVI-PRM`.



(a) Warehouse environment (b) The Warehouse PRM

Figure 1: The Warehouse example and the corresponding PRM. The robot needs to pick up an item and delivers the item to the right location in sequence when the item may not be ready and the delivery location could be busy. **(a)**: a 4×4 grid world in which is our robot, is the charging station, is the item pickup location, is the delivery location;**(b)**: The corresponding PRM, where an edge $q \xrightarrow[r]{\ell|p} q'$ represents that state q transitions to q' on event l with probability p and receives reward r.

2 Related Work

Reward Machines Icarte et al. (2018) introduced the concept of Deterministic Reward Machines (DRMs), a type of finite state machine that specifies reward functions while exposing the structure of these functions to the learner. They also proposed an algorithm called QRM, which outperforms standard Q-learning and hierarchical RL algorithms in environments where the reward can be specified by a DRM. Simultaneously, there has been extensive research on RL with temporal specifications expressed in Linear Temporal Logic (LTL) (Sadigh et al. 2014; Wen, Ehlers, and Topcu 2015; Li, Vasile, and Belta 2017), which can be directly translated into DRMs. Of particular interest to us is the recent work of Bourel et al. (2023), which studies online RL with DRMs in the context of infinite horizon average reward MDPs. They establish a regret lower-bound of $\Omega(\sqrt{HOAT})$ in the episodic setting. They also propose two algorithms differing in the confidence interval design, that achieve a regret bound of order $\widetilde{O}(H\sqrt{OAT})$ and $\widetilde{O}(HO\sqrt{AT})$, respectively. We improve upon this work with an algorithm that outperforms theirs both empirically and theoretically.

Empirically, RL with LTL or DRMs has been successfully applied to complex robotics tasks, such as manipulation (Camacho et al. 2021) and locomotion (DeFazio, Hayamizu, and Zhang 2024). DRMs have also been employed in multi-agent learning scenarios (Neary et al. 2020; Hu et al. 2024; Zheng and Yu 2024). However, all of these works have focused exclusively on Deterministic RMs. Dohmen et al. (2022) introduced the concept of Probabilistic Reward Machines (PRMs), where state transitions are probabilistic rather than deterministic. However, to the best of our knowledge, no algorithms have yet been proposed to solve PRMs with rigorous theoretical guarantees.

Reward-free exploration Reward-free exploration (Jin et al. 2020) studies the problem where an agent needs to collect data by interacting with the environment during the exploration stage, preparing for learning the optimal policy for an unknown reward function during the offline planning stage. The number of potential rewards can be arbitrarily large or even infinite. Jin et al. (2020) proposed an algorithm that can return an ϵ-optimal policies for an arbitrary Markovian reward function, after at most collecting $\widetilde{O}\left(\frac{H^5S^2A}{\epsilon^2}\right)$ samples in the exploration stage. Ménard et al. (2021) further improved the sample complexity to $\widetilde{O}\left(\frac{H^3S^2A}{\epsilon^2}\right)$. Later work extends this problem to MDPs with more general structures, such as linear MDPs(Wang et al. 2020; Zhang, Zhou, and Gu 2021; Wagenmaker et al. 2022) and kernel MDPs(Qiu et al. 2021), and other settings such as multitask RL(Zhang, Ma, and Singla 2020) and multi-agent RL(Bai and Jin 2020; Liu et al. 2021). However, *all* of the above works restrict their attention on Markovian rewards. This is because their analysis crucially relies on the well-known simulation lemma that quantifies the value difference of a policy in two different MDPs. This classic version of simulation lemma only holds for Markovian reward. We provide the first extension of reward-free RL to the non-Markovian reward setting via a new simulation lemma that holds for arbitrary non-Markovian reward (Lemma 2).

3 Problem Formulation

We start with a few definitions.

Labeled MDPs with Probabilistic Reward Machines A labeled MDP(Xu et al. 2022) with PRM(Dohmen et al. 2022) is defined as a tuple $M = (\mathcal{O}, \mathcal{A}, p, \mathcal{R}, \mathcal{P}, L, H)$. $\mathcal{O}$ represents a finite set of observations with cardinality O, and $\mathcal{A}$ represents a finite set of actions available at each observation with cardinality A. The transition function $p : \mathcal{O} \times \mathcal{A} \to \Delta_{\mathcal{O}}$ dictates the probability $p(o'|o, a)$ of transitioning to observation o' when action a is taken in observation o. The set $\mathcal{P}$ consists of atomic propositions, and the labeling function $L : \mathcal{O} \times \mathcal{A} \times \mathcal{O} \to 2^{\mathcal{P}}$ assigns a subset of $\mathcal{P}$ to each transition (o, a, o'). These labels represent high-level events associated with transitions that can be detected from the ob-

servation space. The component $\mathcal{R}$ is a Probabilistic Reward Machine (PRM), which generates history-dependent reward functions. A PRM is defined as a tuple $\mathcal{R} = (\mathcal{Q}, 2^{\mathcal{P}}, \tau, \nu)$, where $\mathcal{Q}$ is a finite set of states with cardinality Q. The probabilistic transition function $\tau : \mathcal{Q} \times 2^{\mathcal{P}} \to \Delta_{\mathcal{Q}}$ determines the probability $\tau(q'|q, \sigma)$ of transitioning to state q' given that the event σ occurs in state q. The function $\nu : \mathcal{Q} \times 2^{\mathcal{P}} \times \mathcal{Q} \to \Delta_{[0,1]}$ maps each transition to a reward. The horizon H defines the length of each episode. Note that MDP with PRM is a special class of Non-Markovian Reward Decision Processes (NMRDPs)(Bacchus, Boutilier, and Grove 1996), which is identical to a MDP except that the reward depends on the history of state and action. For a NMRDP, we define $F(\eta)$ as the expected reward collected by trajectory $\eta \stackrel{\text{def}}{=} \{(o_t, a_t)_{t=1}^{H}\}$ and $G \stackrel{\text{def}}{=} \max_\eta F(\eta)$.

A special class of PRMs is Deterministic Reward Machines (DRMs)(Icarte et al. 2018, 2022), where the transition function τ is deterministic. In a DRM, given a state q and an event σ, the next state q' is uniquely determined, and the rewards are generated deterministically based on these transitions.

The agent interacts with the MDP with PRM M as follows: At each time step t, the agent is in observation $o_t \in \mathcal{O}$ and selects an action $a_t \in \mathcal{A}$ based on the history $h_t = (o_1, a_1, \ldots, o_{t-1}, a_{t-1}, o_t)$. After executing the action a_t in observation o_t, the environment transitions to the next observation $o_{t+1} \sim p(\cdot|o_t, a_t)$ and assigns a label $\sigma_t = L(o_t, a_t, o_{t+1})$. Simultaneously, the PRM, which is in state q_t, transitions to state $q_{t+1} \sim \tau(\cdot|q_t, \sigma_t)$ and outputs a reward $\boldsymbol{r}_t = \nu(q_t, \sigma_t, q_{t+1})$. The agent then receives this reward, and both the environment and the PRM proceed to their next observation o_{t+1} and state q_{t+1}, respectively.

We define the joint state space as the cross-product of $\mathcal{Q}$ and $\mathcal{O}$, i.e. $\mathcal{S} = \mathcal{Q} \times \mathcal{O}$. The transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ dictates the probability $p(s'|s, a)$ of transitioning to state s' when action a is taken in s. The policy is defined as a mapping $\pi : \mathcal{S} \times [H] \to \mathcal{A}$. For every π, the occupancy measure $\mu_h^\pi : \mathcal{S} \times \mathcal{A} \to \Delta_{[0,1]}$ denotes the probability distribution at time h induced by policy π over state action space. Further we denote $\mu^\pi(s, a) \stackrel{\text{def}}{=} \sum_{h=1}^{H} \mu_h^\pi(s, a)$. The value $V_h^\pi : \mathcal{S} \to \mathbb{R}$ denotes the value function at each time step h and state $s = (q, o) \in \mathcal{S}$ is a tuple of $q \in \mathcal{Q}$ and $o \in \mathcal{O}$. $V_h^\pi(s)$ corresponds to the expected sum of $H - h$ rewards received under policy π, starting from $s_h = s \in \mathcal{S}$. We define the state transition kernel P_h^π under the policy π at time step h as P_h^π. $P_h^\pi(s'|s) \stackrel{\text{def}}{=} P(s'|s, \pi(s, h)) = p(o'|o, \pi(s, h))\tau(q'|q, L(o, \pi(s, h), o'))$ where $s' = (q', o')$, and $r_h^\pi = R(s, \pi(s, h))$. For every $V : \mathcal{S} \to \mathbb{R}$ the operators $P\cdot$ and $P_h^\pi\cdot$ are defined as $(PV)(s, a) \stackrel{\text{def}}{=} \sum_{s' \in \mathcal{S}} P(s'|s, a)V(s')$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$ and $(P_h^\pi V)(s) \stackrel{\text{def}}{=} \sum_{s' \in \mathcal{S}} P_h^\pi(s'|s)V(s')$. We define the optimal value function $V_h^*(s) \stackrel{\text{def}}{=} \sup_\pi V_h^\pi(s)$ for all $s \in \mathcal{S}$ and $h \geq 1$. The learner's goal is to minimize their cumulative regret given the knowledge of PRM, defined via

$$\text{Regret}(K) := \sum_{k=1}^{K} \left(V_1^*(s_{k,1}) - V_1^{\pi_k}(s_{k,1})\right).$$

where π_k the policy followed by the learner at episode k. The regret measures the loss of the learner who doesn't follow the optimal policy. Note that in most practical RL applications, human experts specify the reward functions to guide agents in learning the desired behaviors. Therefore, assuming knowledge of PRM is natural.

The following lemma makes it formal that the joint state transforms a PRM into an MDP with Markovian reward and transition.

Lemma 1. *(Bourel et al. 2023) Let $M = (\mathcal{O}, \mathcal{A}, p, \mathcal{R}, \mathcal{P}, L, H)$ be a finite MDP with PRM. Then, an associated cross-product MDP to M is $M_{cp} = (\mathcal{S}, \mathcal{A}, P, R)$, where $\mathcal{S} = \mathcal{Q} \times \mathcal{O}$ and for $s = (q, o)$, $s' = (q', o') \in \mathcal{S}$ and $a \in \mathcal{A}$,*

$$P(s'|s, a) = p(o'|o, a)\tau(q'|q, L(o, a, o'))$$
$$R(s, a) = \sum_{o' \in \mathcal{O}, q' \in \mathcal{Q}} p(o'|o, a)\nu(q, L(o, a, o'), q').$$

This Lemma allows one to apply off-the-shelf RL algorithms to the cross-product MDP given the knowledge of PRM. However, the regret of such an approach will grow not slower than $\Omega(\sqrt{QOAHT})$(Auer, Jaksch, and Ortner 2008) that scales with the joint state space size $|Q| \cdot |O|$. In contrast, as we show next, it is possible to design algorithms whose regret scales only with the observation state space size $|O|$ and independent from the size of the label space, which can be as large as $|O|^H |A|^{H-1}$.

4 Learning Algorithms and Results

In this section, we present our RL algorithm for PRMs, `UCBVI-PRM`. `UCBVI-PRM` follows the algorithmic skeleton of a classic model-based RL algorithm (Azar, Osband, and Munos 2017), while incorporating designs that leverage the structure of PRMs. Our key contribution is a regret bound of $\widetilde{O}(\sqrt{HOAT})$ when T is large enough and $OA \geq H$. The regret bound matches the established lower bound $\Omega(\sqrt{HOAT})$ up to a logarithmic factor for MDP with DRM, and is notably independent of the joint state space size.

Intuitively, `UCBVI-PRM` (Algorithm 1) proceeds in 3 stages: (i) From lines 1 to 7, the algorithm first constructs an empirical transition matrix based on the data collected thus far; (ii) Using this empirical transition matrix, the algorithm then performs value iteration from lines 8 to 23 to update the value function. Notably, between lines 8 and 19, the algorithm computes the new action-value function using the updated empirical transition matrix (line 12) and the exploration bonus (line 13); (iii) Finally, from lines 24 to 28, the agent selects actions based on the updated action-value function and collects new data, which is then incorporated into the dataset.

The main technical novelty lies in how we utilize the PRM structure. Denote $W_h : \mathcal{Q} \times \mathcal{O} \times \mathcal{A} \times \mathcal{O} \to \mathbb{R}$ a function

Algorithm 1: UCBVI-PRM

Input: Bonus algorithm bonus
Initialize: $D = \emptyset$

for episode $k \in [K]$ **do**
  Compute, for all $(o, a, z) \in \mathcal{O} \times \mathcal{A} \times \mathcal{O}$,
  $N_k(o, a, z) = \sum_{(o', a', z') \in \mathcal{H}} \mathbb{I}(o', a', z') = (o, a, z)$
  $N_k(o, a) = \sum_{z \in \mathcal{O}} N_k(o, a, z)$
  $N'_{k,h}(o, a) = \sum_{(o_{i,h}, a_{i,h}) \in \mathcal{H}} \mathbb{I}(o_{i,h} = o, a_{i,h} = a)$
  Let $\mathcal{K} = \{(o, a) \in \mathcal{O} \times \mathcal{A}, N_k(o, a) \geq 0\}$
  Set $\widehat{p}_k(z|o, a) = \frac{N_k(o,a,z)}{N_k(o,a)}$ for all $(o, a) \in \mathcal{K}$
  for timestep $h = H, H-1, \ldots, 1$ **do**
    for all $(o, a) \in \mathcal{O} \times A$ **do**
      if $(o, a) \in \mathcal{K}$ **then**
        for all $s = (q, o), s' = (q', o'), q, q' \in \mathcal{Q}$ **do**

$$\widehat{P}_k(s'|s, a) = \widehat{p}_k(o'|o, a)\tau(q'|q, L(o, a, o'))$$
$$\widehat{R}_k(s, a) = \sum_{s' \in \mathcal{S}} \widehat{P}_k(s'|s, a)\nu(q, L(o, a, o'), q')$$

$$b_{k,h}(s, a) = \texttt{bonus}(\widehat{p}_k(o, a), V_{k,h+1}, N_k, N'_{k,h})$$

$$Q_{k,h}(s, a) = \min\{Q_{k-1,h}(s, a), H, \widehat{R}_k(s, a) + (\widehat{P}_h^k V_{k,h+1})(s, a) + b_{k,h}(s, a)\}$$

        end for
      else
        $Q_{k,h}(s, a) = H$
      end if
    end for
    for all $s \in S$ **do**
      set $V_{k,h}(s) = \max_{a \in A} Q_{k,h}(s, a)$
    end for
  end for
  for timestep $h \in [H]$ **do**
    take greedy action $a_{k,h} = \arg\max_{a \in A} Q_{k,h}(s_{k,h}, a)$
    observe state$(q_{k,h+1}, o_{k,h+1})$
    update dataset $D \leftarrow D \cup \{(h, o_{k,h}, a_{k,h}, o_{k,h+1})\}$
  end for
end for

Algorithm 2: bonus

Require: $\widehat{p}_k(o, a)$, $V_{k,h+1}$, N_k, $N'_{k,h}$

for all $o' \in \mathcal{O}$ **do**
  $W_{k,h+1}(q, o, a, o') = \sum_{q' \in \mathcal{Q}} \tau(q'|q, L(o, a, o'))V_{k,h+1}(q', o')$
end for
$\widehat{\mathbb{W}}_{k,h+1} = \mathrm{Var}_{o' \sim \widehat{p}_k(\cdot|o,a)}(W_{k,h+1}(q, o, a, o'))$
$b_{k,h}(s, a) = \sqrt{\frac{8\iota \widehat{\mathbb{W}}_{k,h+1}}{N_k(o,a)}} + \frac{14H\iota}{3N_k(o,a)} + \sqrt{\frac{2\iota}{N_k(o,a)}} + \sqrt{\frac{8\sum_{o' \in \mathcal{O}} \widehat{p}_k(o'|o,a) \min\left(\frac{100^2 H^3 O^2 A_\iota{}^2}{N'_{k,h+1}(o')}, H^2\right)}{N_k(o,a)}}$
where $\iota = \ln(6QOAT/\rho)$
return $b_{k,h}(s, a)$

that measures the expected return when being in state (q, o), executing action a at time step $h-1$ and observing o' at time step h. W is defined as follows:

$$W_h(q, o, a, o') = \sum_{q' \in \mathcal{Q}} \tau(q'|q, L(o, a, o'))V_h(q', o')$$

W_h is similar to value function V_h in the sense that both are expected returns but condition on different random variables. Consequently, the estimation error can be characterized by W_h instead of V_h, and our bonus will be a function of W_h instead of V_h. More precisely, the estimation error $(\widehat{P}_k^{\pi_k} - P_h^{\pi_k})V_{h+1}^*$ can be translated to the estimation error in the observation space $(\widehat{p}_k^{\pi_k} - p^{\pi_k})W_{h+1}^*$.

We utilize a Bernstein-type reward bonus to ensure that $V_{k,h}$ serves as an upper bound for V_h^* with high probability, a common technique in the literature (Azar, Osband, and Munos 2017; Zanette and Brunskill 2019; Zhang, Ji, and Du 2021). Unlike previous works that directly use $V_{k,h}$, we leverage our knowledge of Probabilistic Reward Machines (PRMs) to construct our bonus using $W_{k,h}$. This approach results in the regret associated with our bonus design growing only in the order of $|O|$ instead of $|S|$.

Theorem 1. *(Regret bound for UCBVI-PRM) Consider a parameter $\rho \in (0, 1)$. Then the regret of UCBVI-PRM is bounded w.p. at least $1 - \rho$, by*

$$\textit{Regret}(K) \leq 72\iota\sqrt{OAHT} + 2500H^2O^2A^{3/2}\iota^2 + 4H\sqrt{T\iota}$$

where $\iota = \ln(6QOAT/\rho)$.

Notice that the leading term of the regret does not scale with $|Q|$. In contrast, if one were to apply an off-the-shelf RL algorithm to the cross-product MDP, it could achieve a regret bound no better than $\Omega(\sqrt{HQOAT})$ (Auer, Jaksch, and Ortner 2008). In the work of Bourel et al. (2023), their algorithms *i.e.*, UCRL2-RM-L and UCRL2-RM-B achieve a regret bound of $\widetilde{O}(H\sqrt{OAT})$ and $\widetilde{O}(HO\sqrt{AT})$ in DRMs, respectively. Compared to their results, we improve the regret bound by a factor of $\sqrt{H}$ and $\sqrt{HO}$ respectively, while generalizes to the PRM setting.

5 RL with Non-Markovian Rewards

In this section, we introduce an explore-then-commit style algorithm for MDPs with generic non-Markovian rewards: in the exploration stage, the agent collects trajectories from the environment without the information of rewards; in the planning stage, the agent computes a near-optimal policy given the data gathered in the exploration stage, assuming access to an approximate planner. We give an efficient algorithm that conducts $\widetilde{O}(\frac{O^5A^3H^2G^2}{\epsilon^2})$ episodes of exploration

and returns an $(\epsilon + \alpha)$-optimal policy for any general non-Markovian rewards, given an α-approximate planner, formally stated below.

Definition 1. *A planner is α-approximate if given any NMRDP $M = (\mathcal{O}, \mathcal{A}, p, \mathcal{R}, H)$, the planner returns a policy π that satisfies*

$$J(\pi) \geq J(\pi^*) - \alpha$$

where $J(\pi)$ is the expected return of executing policy π in M and π^* is the optimal policy in M.

Theorem 2. *The exists an absolute constant $c > 0$, such that, for any $p \in (0, 1)$, with probability at least $1 - \rho$, given the information collected by algorithm 3, algorithm 4 can output $(\epsilon + \alpha)$-optimal policies for any non-Markovian rewards assuming access to α-approximate planner. The number of episodes in algorithm 3 is bounded by*

$$c \cdot \left[\frac{O^5 A^3 H^2 G^2 \iota'}{\epsilon^2} + \frac{O^4 A H^4 G \iota'^3}{\epsilon} \right]$$

where $\iota' \stackrel{def}{=} \ln\left(\frac{OAHG}{\rho\epsilon}\right)$.

This result is made possible by a new *simulation lemma* that can be applied to generic non-Markovian rewards and non-Markovian policies.

Lemma 2. *For any two NMRDPs $M = (\mathcal{O}, \mathcal{A}, p, \mathcal{R}, H)$ and $\widehat{M} = (\mathcal{O}, \mathcal{A}, \widehat{p}, \mathcal{R}, H)$, for any policy π*

$$\left|\widehat{J}(\pi) - J(\pi)\right| \leq \sum_{m=1}^{H} \sum_{o_m, a_m, o_{m+1}} \epsilon(o_{m+1}|o_m, a_m) \mu_m^\pi(o_m, a_m) G$$

where

$$\epsilon(o_{m+1}|o_m, a_m) = |\widehat{p}(o_{m+1}|o_m, a_m) - p(o_{m+1}|o_m, a_m)|$$

The proof can be found in the Appendix D.1. This lemma characterizes the performance difference of the same policy applied to two Non-Markovian Reward Decision Processes (NMRDPs) that differ in their transition kernels. The performance gap is determined by the divergence in the transition kernel ϵ, the occupancy measure μ_h^π induced by the policy in M, and the return upperbound G. This lemma is key to establish our result, because it can be applied to any non-Markovian reward and non-Markovian policy, including Markovian rewards and PRMs as special cases. Intuitively, this lemma provides a concrete goal for the exploration stage, i.e. the gap $\epsilon(o_{m+1}|o_m, a_m)$ must be small for any (o_m, a_m) pair that is visited significantly often under π. In the following, we show how to achieve this goal.

5.1 Exploration stage

It turns out that a procedure (Algorithm 3) similar to the Markovian reward case suffices for our purpose (Jin et al. 2020). Intuitively, algorithm 3 perform two steps. In the first step, from lines 2 to 7, the algorithm constructs a set of exploratory policies each designed to visit an observation state $o \in \mathcal{O}$ as often as possible. To accomplish this, for each observation o, the algorithm creates a reward function that is 0 everywhere except at observation o, where it is set to 1 (line 3). The algorithm then employs a no-regret RL algorithm (e.g. EULER of Zanette and Brunskill (2019)) to find a policy that maximizes this reward, which is equivalent to maximizing the probability of visiting o. In the second stage, from lines 8 to 12, the algorithm collects new data by sampling and executing policies from this exploratory policy set. We prove that, with this framework, the collected data can be used to learn a transition kernel that is sufficiently close to the true transition characterized by the divergence in Lemma 2. Towards this, we introduce the notion of significant observation:

Algorithm 3: Reward-free RL-Explore

Require: iteration number N_0, N.
1: set policy class $\Psi \leftarrow \emptyset$, and dataset $D \leftarrow \emptyset$.
2: **for all** $o \in \mathcal{O}$ **do**
3: $r'(o', a') \leftarrow \mathbb{I}[o' = o]$ for all $(o', a') \in \mathcal{O} \times \mathcal{A}$.
4: $\Phi_o \leftarrow \texttt{EULER}(r, N_0)$.
5: $\pi(\cdot \mid o) \leftarrow \text{Uniform}(\mathcal{A})$ for all $\pi \in \Phi_o$.
6: $\Psi \leftarrow \Psi \cup \Phi_o$.
7: **end for**
8: **for** $n = 1$ to N **do**
9: sample policy $\pi \sim \text{Uniform}(\Psi)$.
10: play $\mathcal{M}$ using policy π, and observe the trajectory $z_n = (o_1, a_1, \ldots, o_H, a_H, o_{H+1})$.
11: $D \leftarrow D \cup \{z_n\}$
12: **end for**
13: **return** dataset D.

Definition 2 (Significant Observation)**.** *A observation o is δ-significant if there exists a policy π, so that the probability to reach o following policy π is greater than δ. In symbol:*

$$\max_\pi \mu^\pi(o) \geq \delta$$

where $\mu^\pi(o) = \sum_a \mu^\pi(o, a)$.

Intuitively, since insignificant observations are rarely reachable by any policy, their contributions to the divergence in Lemma 2 will be limited. Thus, it suffices to only visit significant observations. Algorithm 3 is designed specifically for this purpose, and achieves the following guarantee.

Theorem 3. *(Theorem 3 of (Jin et al. 2020)) There exists absolute constant $c > 0$ such that for any $\varepsilon > 0$ and $\rho \in (0, 1)$, if we set $N_0 \geq cO^2AH^4\iota_0^3/\delta$ where $\iota_0 = \log(OAH/\rho\delta)$, then with probability at least $1 - \rho$, that Algorithm 3 will return a dataset D consisting of N trajectories $\{z_n\}_{n=1}^N$, which are i.i.d sampled from a distribution λ satisfying:*

$$\forall \delta - \textit{significant } o, \quad \max_{a,\pi} \frac{\mu^\pi(o, a)}{\lambda(o, a)} \leq 2OAH. \qquad (1)$$

As we can see from theorem 3, all significant observations can be visited by distribution λ with reasonable probability. Hence, with algorithm 3, we can learn our model by visiting significant observations without the guidance of any rewards.

5.2 Planning stage

Algorithm 4: Rewards-free-Plan

Input: a dataset of transition D, Rewards $\mathcal{R}$.
for all $(o, a, o') \in \mathcal{O} \times \mathcal{A} \times \mathcal{O}$ **do**
  $N(o, a, o') \leftarrow \sum_{(o_h, a_h, o_{h+1}) \in D} \mathbb{I}[o_h = o, a_h = a, o_{h+1} = o']$.
  $N(o, a) \leftarrow \sum_{o'} N(o, a, o')$.
  $\widehat{p}(o'|o, a) = \frac{N(o,a,o')}{N(o,a)}$.
end for
$\widehat{\pi} \leftarrow \alpha$-approximate planner$(\widehat{p}, \mathcal{R})$.
Return: policy $\widehat{\pi}$.

After collecting enough data in the exploration stage, algorithm 4 use the data to compute an empirical transition matrix $\widehat{p}$, on which the approximate planner is employed. We prove that (see Appendix D.2), any policy will have small value gap in the learned transition under $\widehat{p}$ vs. the ground truth transition p.

Lemma 3. *There exists an absolute constant $c > 0$, for any $\epsilon > 0$, $\rho \in (0, 1)$, assume the dataset $\mathcal{D}$ has N i.i.d. samples from distribution λ which satisfies equation 1 with $\delta = \frac{\epsilon}{8OG}$, and $N \geq c\frac{O^5A^3H^2G^2\iota'}{\epsilon^2}$, where $\iota' = \ln\left(\frac{2H}{\rho}\right)$, then w.p. at least $1 - 2\rho$:*

$$\left|\widehat{J}(\pi) - J(\pi)\right| \leq \frac{\epsilon}{2}$$

The reason for the increased sample complexity compared to the original analysis by Jin et al. (2020) lies in the fact that more samples are required to reduce the model error associated with significant observations than in the Markovian setting. Specifically, in our analysis, it is necessary to account for model errors across every (o, a, o') triplet. In contrast, in the standard Markovian setting, the modeling error can be further decomposed into the error of the empirical next-state value function (see the proof of Lemma 3.6 in Jin et al. (2020)), which allows for tighter bounds. After we obtain our empirical transition matrix $\widehat{p}$, given any non-Markovian rewards $\mathcal{R}$, we can find a near optimal policy by running α-approximate planner, as a result of our simulation Lemma.

Lemma 4. *Under the preconditions of lemma 3, with probability of $1 - 2\rho$ for any rewards $\mathcal{R}$, the output policy of algorithm 4 is $\epsilon + \alpha$-optimal, that is*

$$|J(\pi^*) - J(\widehat{\pi})| \leq \epsilon + \alpha$$

where π^* is the optimal policy.

Note that, for general non-Markovian rewards, the optimization error won't be reduced to 0, but for any PRMs, the optimization can be reduced to 0, since we can run value iteration given the cross-product MDP and solve it optimally. In addition, there are some cases where the rewards possess special structural properties, for which performance with guarantees can be achieved(Prajapat et al. 2023; De Santi, Prajapat, and Krause 2024).

6 Experiments

In this section, we present a series of experiments comparing the empirical performance of our algorithm, `UCBVI-PRM`, with existing baselines. We evaluate our algorithm in MDPs with both DRM and PRM against different baselines. For DRMs, we compare with `UCRL-RM-L` and `UCRL-RM-B` of Bourel et al. (2023). For PRM, since there is no existing algorithm, we compare with the naive approach of directly applying `UCBVI`(Azar, Osband, and Munos 2017) onto the cross-product MDP. In our experiment, we tune the exploration coefficient for all algorithms by selecting from a equally large set of options (see Appendix E.2). This is to make sure that an algorithm with a larger hyper-parameter set does not get an unfair advantage. In addition, we apply the doubling trick (detailed in Appendix E.1) to speed up `UCBVI-PRM`, which is a common technique in the literature(Auer, Jaksch, and Ortner 2008; Dann and Brunskill 2015) and won't affect the $\widetilde{O}$ regret.

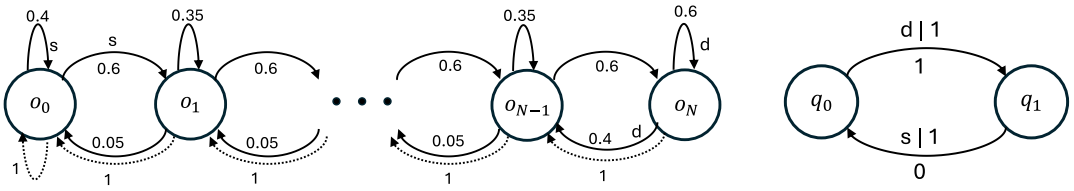
6.1 DRM Experiments

In the RiverSwim environment, shown in Figure 2, the agent has two actions corresponding to swimming left or right. Going right results in stochastic transitions, as shown by the solid lines in Figure 2(a). Going left results in deterministic transitions as shown by the dashed lines in Figure 2(a). The agent receives reward when visit two extreme locations in RiverSwim(*i.e.*, o_1 and o_N) in sequence.

Figure 3(a), 3(b), and 3(c) show the regret over time in the RiverSwim domain, with the results averaged over 16 runs. The shaded area shows the standard variance of the corresponding quantity. Specifically, Figures 3(a), 3(b), and 3(c) present the regrets of the agent running in a RiverSwim MDP with 5 observations and a horizon length of 10, a RiverSwim MDP with 10 observations and a horizon length of 20, and a RiverSwim MDP with 15 observations and a horizon length of 30, respectively. As we can see from the figures, in simpler environments (fewer observations and shorter horizons), the advantage of `UCBVI-PRM` is not obvious (see Figure 3(a)). However, with longer horizons and more observations, the gap between `UCBVI-PRM` and the baselines of Bourel et al. (2023) becomes larger. These results align with our regret analysis, where the regret of `UCBVI-PRM` grows slower than `UCRL-RM-L` in the order of H and slower than `UCRL-RM-B` in the order of H and O.

6.2 PRM Experiments

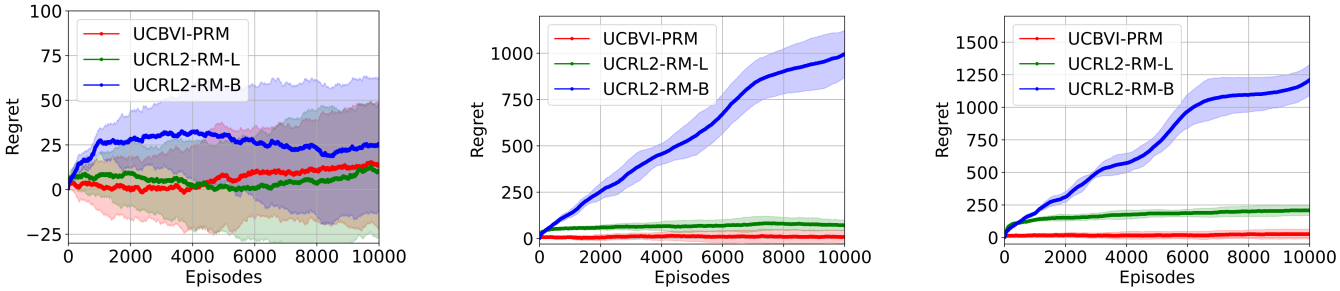
In the warehouse environment(see Figure 1), the robot has five actions corresponding to moving up, right, down, left, and stay. Moving up, right, down, or left leads to moving in the intended direction with probability 0.7, in each perpendicular direction with probability 0.1, or staying in the same place with probability 0.1. The stay action will result in the robot staying in the same place deterministically. The robot receives reward when successfully picks up an item and delivers it to the delivery location in sequence. Figures 4 show the regret over time, with the results averaged over 16 runs. Specifically, Figures 4(a), 4(b), and 4(c) present the



(a) Labeled RiverSwim MDP

(b) The Patrol DRM

Figure 2: The labeled RiverSwim and the corresponding DRM.

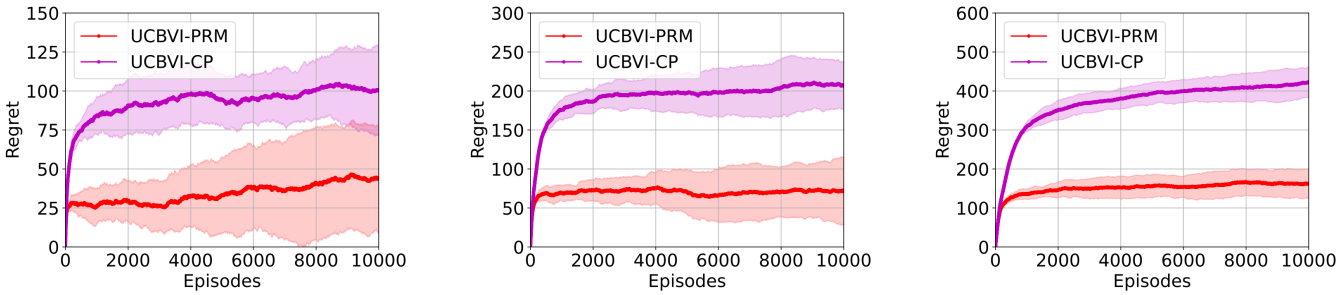


(a) : $H = 10, O = 5$

(b) : $H = 20, O = 10$

(c) : $H = 30, O = 15$

Figure 3: Experimental results in RiverSwim



(a) : $H = 9$, 3×3 warehouse

(b) : $H = 12$, 4×4 warehouse

(c) : $H = 15$, 5×5 warehouse

Figure 4: Experimental results in Warehouse

results of the agent running in a 3×3 warehouse with a horizon length of 9, a 4×4 warehouse with a horizon length of 12, and a 5×5 warehouse with a horizon length of 15, respectively. In all experiments, `UCBVI-PRM` outperforms the baseline. In addition, as the horizon becomes longer and with larger warehouse, `UCBVI-PRM` beats the baseline with a larger margin.

7 Conclusion

We study sample-efficient reinforcement learning in episodic Markov decision processes with probabilistic reward machines. We introduce an algorithm tailored for PRMs that matches the established lower bound when $T \geq H^3O^3A^2$ and $OA \geq H$. We also present a lemma that characterizes the difference in policy evaluations between two MDPs with non-Markovian rewards. Building upon the new lemma, we establish the reward-free learning result for non-Markovian reward. Finally, we validate our algorithms through a series of experiments. Interesting future direction includes designing effective and efficient algorithms for the multi-agent setting, and exploring connections with reward structures such as submodular rewards.

References

Auer, P.; Jaksch, T.; and Ortner, R. 2008. Near-optimal regret bounds for reinforcement learning. *Advances in neural information processing systems*, 21.

Azar, M. G.; Osband, I.; and Munos, R. 2017. Minimax Regret Bounds for Reinforcement Learning. arXiv:1703.05449.

Bacchus, F.; Boutilier, C.; and Grove, A. 1996. Rewarding behaviors. In *Proceedings of the National Conference on Artificial Intelligence*, 1160–1167.

Bai, Y.; and Jin, C. 2020. Provable self-play algorithms for

competitive reinforcement learning. In *International conference on machine learning*, 551–560. PMLR.

Bourel, H.; Jonsson, A.; Maillard, O.-A.; and Talebi, M. S. 2023. Exploration in Reward Machines with Low Regret. In Ruiz, F.; Dy, J.; and van de Meent, J.-W., eds., *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, 4114–4146. PMLR.

Bubeck, S.; and Cesa-Bianchi, N. 2012. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. arXiv:1204.5721.

Camacho, A.; Varley, J.; Zeng, A.; Jain, D.; Iscen, A.; and Kalashnikov, D. 2021. Reward machines for vision-based robotic manipulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 14284–14290. IEEE.

Cesa-Bianchi, N.; and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge university press.

Dann, C.; and Brunskill, E. 2015. Sample complexity of episodic fixed-horizon reinforcement learning. *Advances in Neural Information Processing Systems*, 28.

De Santi, R.; Prajapat, M.; and Krause, A. 2024. Global Reinforcement Learning: Beyond Linear and Convex Rewards via Submodular Semi-gradient Methods. *arXiv preprint arXiv:2407.09905*.

DeFazio, D.; Hayamizu, Y.; and Zhang, S. 2024. Learning quadruped locomotion policies using logical rules. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 34, 142–150.

Dohmen, T.; Topper, N.; Atia, G.; Beckus, A.; Trivedi, A.; and Velasquez, A. 2022. Inferring probabilistic reward machines from non-markovian reward signals for reinforcement learning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 32, 574–582.

Freedman, D. A. 1975. On Tail Probabilities for Martingales. *The Annals of Probability*, 3(1): 100 – 118.

Gheshlaghi Azar, M.; Munos, R.; and Kappen, H. J. 2013. Minimax PAC bounds on the sample complexity of reinforcement learning with a generative model. *Machine learning*, 91: 325–349.

Hu, J.; Xu, Z.; Wang, W.; Qu, G.; Pang, Y.; and Liu, Y. 2024. Decentralized graph-based multi-agent reinforcement learning using reward machines. *Neurocomputing*, 564: 126974.

Icarte, R. T.; Klassen, T.; Valenzano, R.; and McIlraith, S. 2018. Using Reward Machines for High-Level Task Specification and Decomposition in Reinforcement Learning. In Dy, J.; and Krause, A., eds., *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, 2107–2116. PMLR.

Icarte, R. T.; Klassen, T. Q.; Valenzano, R.; and McIlraith, S. A. 2022. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73: 173–208.

Jin, C.; Krishnamurthy, A.; Simchowitz, M.; and Yu, T. 2020. Reward-Free Exploration for Reinforcement Learning. arXiv:2002.02794.

Li, X.; Vasile, C.-I.; and Belta, C. 2017. Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3834–3839. IEEE.

Liu, Q.; Yu, T.; Bai, Y.; and Jin, C. 2021. A sharp analysis of model-based reinforcement learning with self-play. In *International Conference on Machine Learning*, 7001–7010. PMLR.

Maurer, A.; and Pontil, M. 2009. Empirical Bernstein Bounds and Sample Variance Penalization. arXiv:0907.3740.

Ménard, P.; Domingues, O. D.; Jonsson, A.; Kaufmann, E.; Leurent, E.; and Valko, M. 2021. Fast active learning for pure exploration in reinforcement learning. In *International Conference on Machine Learning*, 7599–7608. PMLR.

Neary, C.; Xu, Z.; Wu, B.; and Topcu, U. 2020. Reward machines for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2007.01962*.

Prajapat, M.; Mutnỳ, M.; Zeilinger, M. N.; and Krause, A. 2023. Submodular reinforcement learning. *arXiv preprint arXiv:2307.13372*.

Qiu, S.; Ye, J.; Wang, Z.; and Yang, Z. 2021. On reward-free rl with kernel and neural function approximations: Single-agent mdp and markov game. In *International Conference on Machine Learning*, 8737–8747. PMLR.

Sadigh, D.; Kim, E. S.; Coogan, S.; Sastry, S. S.; and Seshia, S. A. 2014. A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, 1091–1096. IEEE.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Wagenmaker, A. J.; Chen, Y.; Simchowitz, M.; Du, S.; and Jamieson, K. 2022. Reward-free rl is no harder than reward-aware rl in linear markov decision processes. In *International Conference on Machine Learning*, 22430–22456. PMLR.

Wang, R.; Du, S. S.; Yang, L.; and Salakhutdinov, R. R. 2020. On reward-free reinforcement learning with linear function approximation. *Advances in neural information processing systems*, 33: 17816–17826.

Wen, M.; Ehlers, R.; and Topcu, U. 2015. Correct-by-synthesis reinforcement learning with temporal logic constraints. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4983–4990. IEEE.

Xu, Z.; Gavran, I.; Ahmad, Y.; Majumdar, R.; Neider, D.; Topcu, U.; and Wu, B. 2022. Joint Inference of Reward Machines and Policies for Reinforcement Learning. arXiv:1909.05912.

Zanette, A.; and Brunskill, E. 2019. Tighter problem-dependent regret bounds in reinforcement learning without domain knowledge using value function bounds. In *International Conference on Machine Learning*, 7304–7312. PMLR.

Zhang, W.; Zhou, D.; and Gu, Q. 2021. Reward-free model-based reinforcement learning with linear function approximation. *Advances in Neural Information Processing Systems*, 34: 1582–1593.

Zhang, X.; Ma, Y.; and Singla, A. 2020. Task-agnostic exploration in reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 11734–11743.

Zhang, Z.; Ji, X.; and Du, S. 2021. Is reinforcement learning more difficult than bandits? a near-optimal algorithm escaping the curse of horizon. In *Conference on Learning Theory*, 4528–4531. PMLR.

Zheng, X.; and Yu, C. 2024. Multi-Agent Reinforcement Learning with a Hierarchy of Reward Machines. *arXiv preprint arXiv:2403.07005*.