Master Thesis Exposé
Reward Machines: A new Framework for Exploiting Reward Structure in Reinforcement Learning

Maximilian Stollmayer

Introduction

The goal of reinforcement learning is to get an agent to learn a task by maximizing the total rewards over sequences of decisions. The challenge lies in the fact that the consequences of decisions are often delayed, meaking it difficult for the agent to associate specific actions with their long-term outcomes.

In reinforcement learning an agent interacts with an environment. The environment's behavior is unknown to the agent. For each action the agent takes it receives a reward from the environment, but how these rewards are determined is not visible to the agent. This makes the environment a complete black box from the agent's perspective. Figure 1 illustrates this feedback loop: the agent performs an action a_t at time t, receives a new state s_{t+1} and reward r_{t+1} from the environment, and uses this feedback to inform its next action.

For example consider as the agent a self-driving car. The car cannot predict the future, so it does not know in advance what it will encounter on the road, i.e. what the next state will be. Furthermore, at first, the car does not know what we expect from it, like whether it should prioritize reaching the desitination quickly, following speed limits or avoiding accidents. Rewards associated to these goals are unknown to the car initially. We want to make the car learn how to drive us from point A to B. But also adhere to constraints like obeying traffic laws.

Specifying the reward function, i.e. the rule that maps each state-action pair to a reward, is often a complex and error-prone process. In our self-driving car example, we would need to account for numerous cases to design

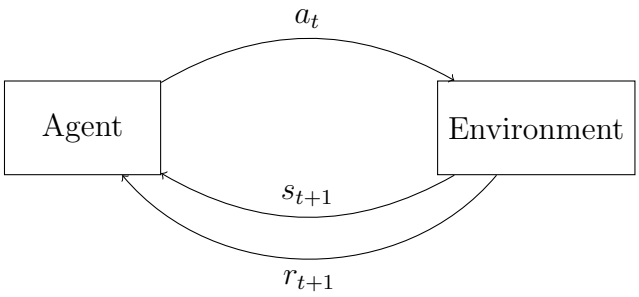

Figure 1: Diagram of the reinforcement learning feedback loop.

rewards that encourage the agent to both achieve the primary task of reaching the destination and satisfy constraints like stopping for pedestrians, etc. Crafting a reward function that balances all these considerations can quickly become unmanagable as tasks increase in complexity.

Additionally sparse rewards present another significant challenge. In many scenarios the agent may receive no feedback for a majority of its actions. For example the self-driving car might only receive a positive reward after coming to a complete halt in front of a passing pedestrian. However it would get no reward for the crucial action of slowing down earlier, even though this behavior is critical for safety and anticipates the pedestrian's presence before they are fully visible.

If we could introduce a layer of abstraction to simplify reward specification, it would make it easier to communicate task objectives to the agent. Furthermore if we exposed the structure of the reward system to the agent rather than treating it as a black box, the agent could exploit this information to accelerate learning.

A reward machine addresses both of these challenges.

Reward Machines

Reward machines are compact, structured representations that encode task objectives and constraints as finite-state machines. By decomposing a task into subtasks and defining rewards at each step, reward machines provide a more interpretable and flexible framework for reward specification. They make it easier to design rewards for complex tasks and enable agents to leverage this structure for faster and more efficient learning. Reward machines were introduced in [1].

Unlike the traditional reinforcement learning feedback loop, this new framework exposes the reward machine to the agent. After the agent takes

action a_t at time t the reward machine receives the new state s_{t+1} from the environment. It then transitions to a new internal state u_{t+1} and assigns a reward r_{t+1}. Compare the figure 2 below to the original feedback loop diagram 1.

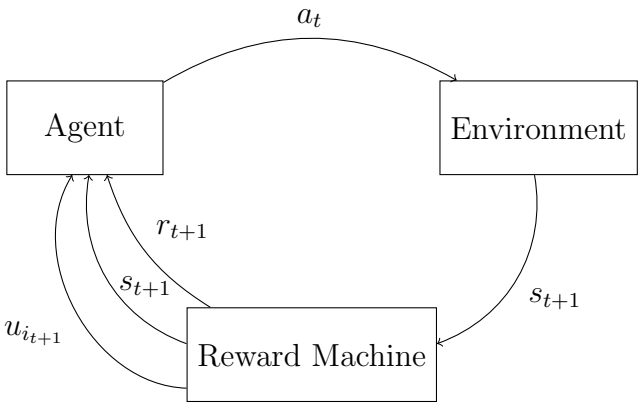

Figure 2: Diagram of the reinforcement learning feedback loop with a reward machine.

Reward machines abstract events in the environment by using propositional symbols. For instance in our self-driving car example a propositional symbol could be 🏠 for the destination, i.e. home, and 🛒 for an intermediate destination, like a shopping center, while ⚠ represents a traffic law violation.

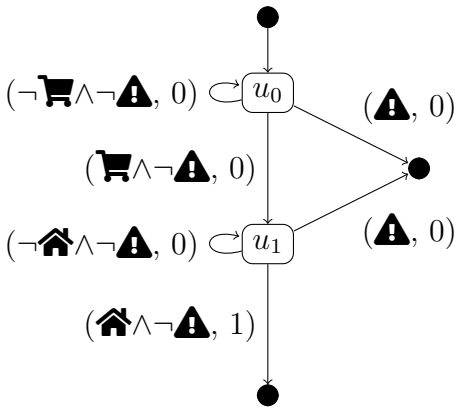

Figure 3: A simple reward machine describing a route for a self-driving car.

Definition. *Given a set of propositional symbols $\mathcal{P}$, a (simple)* reward machine (RM) *is a tuple $\mathcal{R}_\mathcal{P} = (U, u_0, F, \delta_u, \delta_r)$, where*

- *U is a finite set of* states

- $u_0 \in U$ *is an* initial state
- F *is a finite set of* terminal states *s.t.* $U \cap F = \emptyset$
- δ_u *is the* state-transition function $\delta_u : U \times 2^{\mathcal{P}} \to U \cup F$
- δ_r *is the* state-reward function $\delta_r : U \times 2^{\mathcal{P}} \to \mathbb{R}$

[8]

Note that there is a more general version that assigns whole reward functions instead of just values, but this simpler version of a reward machine suffices for this work.

Like in traditional reinforcement learning, where a Markov decision process (MDP) describes the feedback framework, we extend the MDP with a reward machine and a labelling function, which assigns truth values to the defined propositional symbols given the previous state, the agents action and the next state.

Definition. *A* Markov decision process with a reward machine (MDPRM) *is a tuple* $\mathcal{T} = (S, A, P, \gamma, \mathcal{R}_{\mathcal{P}}, L)$*, where*

- S*,* A*,* P *and* γ *are defined as in an MDP*
- $\mathcal{R}_{\mathcal{P}}$ *is a reward machine as defined above*
- L *is a* labelling function $L : S \times A \times S \to 2^{\mathcal{P}}$*.*

[8]

In an MDPRM $\mathcal{T}$ the reward machine $\mathcal{R}_{\mathcal{P}}$ updates at each step of the agent. If the RM is in state u_{i_t} and the agent performs action a_t to move from state s_t to s_{t+1}, then the RM moves to state $u_{t+1} = \delta_u(u_t, L(s_t, a_t, s_{t+1}))$ and the agent receives reward $\delta_r(u_t, L(s_t, a_t, s_{t+1}))$.

In the running example with the reward machine from 3 the RM starts in u_0. As long as the car has not reached its intermediate destination and has not violated any traffic laws, the RM remains in u_0. A traffic law violation transitions the RM to a terminal state with zero reward for the whole episode. Reaching the shopping center transitions the RM to state u_1. Upon arriving home safely, the RM terminates, rewarding the agent with a reward of 1.

By exposing the reward machine's state to the agent, the framework allows the agent to learn a policy $\pi(a|(s, u))$ over pairs in $S \times U$ instead of just over S. An algorithms for learning such policies will be explored later.

Goals of the Master's Thesis

Theorem: Construction of RMs

The first goal of the Master's thesis will be to explore the relationship between formal languages like Linear Temporal Logic (LTL) and reward machines, specifically demonstrating that any task specified using LTL can be represented as a reward machine.

LTL is a formalism for specifying temporal properties and constraints in tasks, commonly used in domains like robotics and verification. Reward machines offer a compact and structured representation of task objectives and constraints, which can incorporate the same kinds of temporal properties specified by LTL. LTL is equivalent in expressiveness to Deterministic Finite Automata (DFA) over infinite sequences. A DFA is a finite-state machine that accepts or rejects sequences of symbols based on a deterministic transition function. Formal languages like regular expressions and other temporal logics are also equivalent to DFAs. Since reward machines are also finite-state machines these logics can be encoded as reward machines, making reward machines a sort of "lingua franca" for reward specification.

A reward specification is a set temporal formulas associated to rewards $\{(r_1 : \varphi_1), \ldots, (r_n : \varphi_n)\}$, which yields reward $r_i \in \mathbb{R}$ when formula φ_i holds. We will show that any LTL-based reward specification can be encoded as a reward machine.

Theorem. *Let $R = \{(r_1 : \varphi_1), \ldots, (r_n : \varphi_n)\}$ be a reward specification, where each φ_i can be transformed into a DFA. Then there exists a reward machine $\mathcal{R}_\mathcal{P}$ that induces the same reward function as R.* [2]

Theorem: Convergence of CRM

To exploit the reward specification provided by a reward machine in the learning process of an agent, we can update the policy by considering the experiences of all possible RM states given the current environment state s, action a and next state s. These are called counterfactual experiences because they simulate how the agent would have behaved if the RM were in a different state. By leveraging these counterfactual updates, we can efficiently utilize the structure of the RM to accelerate learning. This process is formalized in the algorithm (CRM) below. Note that in the algorithm below the update $x \overset{\alpha}{\leftarrow} y$ is defined as $x \leftarrow x + \alpha \cdot (y - x)$.

Theorem. *Given an MDPRM $\mathcal{T}$, CRM with tabular Q-learning converges to an optimal policy for $\mathcal{T}$ in the limit as long as every state-action pair is visited infinitely often.* [8]

Algorithm. Q-learning with counterfactual experiences for RM (CRM) [8]

Require: MDPRM $(S, A, P, \gamma, \mathcal{R}_\mathcal{P}, L)$ with RM $\mathcal{R}_\mathcal{P} = (U, u_0, F, \delta_u, \delta_r)$, initial environment state s_0, learning rate $\alpha \in (0, 1]$, exploration probability $\varepsilon \in (0, 1]$ and number of episodes $n \in \mathbb{N}$.

For all $s \in S$, $u \in U$ and $a \in A$ initialize $\tilde{q}(s, u, a)$ arbitrarily.
for $i \leftarrow 0$ to n **do**
    $u \leftarrow u_0$ and $s \leftarrow s_0$
    while s is not terminal and $u \notin F$ **do**
        Choose ε-greedy action a from (s, u)
        Take action a and observe next state s'
        for $\tilde{u} \in U$ **do**
            $\tilde{r} \leftarrow \delta_r(\tilde{u}, L(s, a, s'))$ and $\tilde{u}' \leftarrow \delta_u(\tilde{u}, L(s, a, s'))$
            if s' is terminal or $\tilde{u} \in F$ **then**
                $\tilde{q}(s, \tilde{u}, a) \overset{\alpha}{\leftarrow} \tilde{r}$
            else
                $\tilde{q}(s, \tilde{u}, a) \overset{\alpha}{\leftarrow} \tilde{r} + \gamma \cdot \max_{a' \in A} \tilde{q}(s', \tilde{u}', a')$
            end if
        end for
        $s \leftarrow s'$ and $u \leftarrow \delta_u(u, L(s, a, s'))$
    end while
end for

The second goal in this thesis will be to prove the convergence of standard tabular Q-learning and then show how CRM as an extension also converges.

The CRM algorithm is not limited to tabular Q-learning. It can be extended to more modern reinforcement learning techniques, such as Deep Q-Networks (DQN) and Deep Deterministic Policy Gradient (DDPG). In these cases, the state-action value function $\tilde{q}(s, u, a)$ is replaced by a neural network to approximate the value function.

Comparison: CRM vs. Q-Learning

A third goal of this thesis is to gain practical experience with reward machines by comparing the CRM algorithm to traditional Q-learning in a simple grid-based environment. This comparisong will help highlight the benefits of CRM in terms of learning efficiency, convergence speed and policy quality when leveraging the RM based reward specification. Furthermore it will show the ease of use of specifying rewards using LTL.

If time permits, the study will extend this comparison to continuous state-action spaces by examining CRM's extension to DDPG. By incorporating

reward machines into the policy gradient framework, the goal will be to evaluate wheter CRM maintains its advantages in more complex settings. A potential task to compare on could be that of autonomous driving in a simplified scenario, such as reaching a destination while adhering to traffic rules.

Outlook

The reward machine framework is a powerful tool for integrating task structure into reinforcement learning. However the field is still evolving and recent extensions have opened new avenues for exploration. So a fourth goal of this thesis is to review these advencements and also consider potential new dircetion for future research.

A list of papers that build upon the reward machine framework we outlined here:

- Recent work [3] explores how reward machines can be integrated with symbolic planning techniques to solve tasks requiring high-level reasoning and decision-making.
- Reward machines have been extended to handle partially observable settings, where the agent does not have full access to the environment's state. [4]
- They have also been applied to vision-based robotic tasks, where they augment DQN for more dense rewards and thus achieving a more robust policy with higher success rates, while still requiring fewer training steps. [5]
- Handling environments with uncertanty and noise in rewards is another active area of research. Recent methods [7] propose robust reward machines to cope with imperfect feedback. Similarly [6] develop stochastic reward machines that are learned via constraint solving.
- In another work a formalism has been proposed that endows RMs the ability to call other RMs, thus building a hierarchy of RMs and exploiting this in the learning process, which leads to faster convergence. [9]

Possible new directions:

- Optimize a reward machine like DFA minimization to reduce the state space while still being equivalent to the reward specification.

- Learn optimal rewards for a reward machine adaptively based on agent performance or infer optimal reward by observing successful behaviors. In [2] and [8] there has been some work in this regard using value iteration to shape the existing rewards.
- Extending reward machines to incorporate a stack-based memory structure, akin to pushdown automata, could enable agents to handle tasks requiring context-sensitive decision-making. Furthermore investigating whether a reward machine framework could be extended to a Turing-complete model would be interesting.
- Translating task specifications directly from natural language into reward machines presents another interesting angle of research.

References

[1] Rodrigo Toro Icarte et al. "Using Reward Machines for High-Level Task Specification and Decomposition in Reinforcement Learning". In: *Proceedings of the 35th International Conference on Machine Learning (ICML)*. 2018, pp. 2112–2121. URL: http://proceedings.mlr.press/v80/icarte18a.html.

[2] Alberto Camacho et al. "LTL and Beyond: Formal Languages for Reward Function Specification in Reinforcement Learning". In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*. 2019, pp. 6065–6073. URL: https://www.ijcai.org/proceedings/2019/0840.pdf.

[3] León Illanes et al. "Leveraging Symbolic Planning Models in Hierarchical Reinforcement Learning". In: *Proceedings of the Knowledge Representation and Reasoning Meets Machine Learning workshop at NeurIPS 2019*. 2019. URL: https://www.cs.toronto.edu/~%20sheila/publications/illanes-et-al-rldm19.pdf.

[4] Rodrigo Toro Icarte et al. "Learning Reward Machines for Partially Observable Reinforcement Learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019, pp. 15497–15508. URL: https://www.cs.toronto.edu/~%20sheila/publications/toro-et-al-neurips19.pdf.

[5] Alberto Camacho et al. "Reward Machines for Vision-Based Robotic Manipulation". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 14284–14290. DOI: 10.1109/ICRA48506.2021.9561927.

[6] Jan Corazza, Ivan Gavran, and Daniel Neider. "Reinforcement Learning with Stochastic Reward Machines". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 36.6 (June 2022), pp. 6429–6436. DOI: 10.1609/aaai.v36i6.20594. URL: https://ojs.aaai.org/index.php/AAAI/article/view/20594.

[7] Andrew C. Li et al. "Noisy Symbolic Abstractions for Deep RL: A case study with Reward Machines". In: *Deep Reinforcement Learning Workshop, NeurIPS 2022*. 2022. URL: https://openreview.net/pdf?id=2Ak703L8umA.

[8] Rodrigo Toro Icarte et al. "Reward Machines: Exploiting Reward Function Structure in Reinforcement Learning". In: *Journal of Artificial Intelligence Research (JAIR)* 73 (2022), pp. 173–208. URL: https://doi.org/10.1613/jair.1.12440.

[9] Daniel Furelos-Blanco et al. *Hierarchies of Reward Machines*. 2023. arXiv: 2205.15752 [cs.LG]. URL: https://arxiv.org/abs/2205.15752.