On the Expressivity of Markov Reward (Extended Abstract)

David Abel¹, Will Dabney¹, Anna Harutyunyan¹, Mark K. Ho², Michael L. Littman³, Doina Precup¹, Satinder Singh¹

¹DeepMind
²Princeton University
³Brown University

{dmabel, wdabney, harutyunyan, doinap, baveja}@deepmind.com, mho@princeton.edu, mlittman@cs.brown.edu

Abstract

Reward is the driving force for reinforcement-learning agents. We here set out to understand the expressivity of Markov reward as a way to capture tasks that we would want an agent to perform. We frame this study around three new abstract notions of “task”: (1) a set of acceptable behaviors, (2) a partial ordering over behaviors, or (3) a partial ordering over trajectories. Our main results prove that while reward can express many of these tasks, there exist instances of each task type that no Markov reward function can capture. We then provide a set of polynomial-time algorithms that construct a Markov reward function that allows an agent to perform each task type, and correctly determine when no such reward function exists.

1 Introduction

At the heart of artificial intelligence (AI) is the study of agents that learn to explore, plan, communicate, and pursue goals. The reinforcement-learning (RL) problem puts the study of such agents front and center under the assumption that we focus on agents that learn to maximize reward. In this sense, reward plays a significant role as a general purpose signal in RL: For any desired behavior, task, or other characteristic of agency, there must exist a reward signal that can incentivize an agent to learn to realize these desires. The expressivity of reward is taken as a backdrop assumption that frames RL, sometimes called the reward hypothesis [Littman, 2017; Christian, 2021]: “...all of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received scalar signal (reward)” [Sutton, 2004]. We here provide an extended abstract¹ that establishes first steps toward a systematic study of the reward hypothesis by examining the expressivity of reward as a signal, proceeding in three steps.

An Account of “Task”. As rewards encode tasks, goals, or desires, we first ask, “what is a task?” We frame our study around a thought experiment (Figure 1) involving the interactions between a designer, Alice, and a learning agent, Bob, drawing inspiration from Ackley and Littman [1992], Sorg [2011], and Singh et al. [2009]. In this thought experiment, we draw a distinction between how Alice thinks of a task, and the means by which Alice incentivizes Bob to pursue this task. This distinction allows us to analyze the expressivity of reward as an answer to the latter question, conditioned on how we answer the former. Concretely, we consider three kinds of task in the context of finite Markov Decision Processes (MDPs): A task is either (1) a set of acceptable behaviors (policies), (2) a partial ordering over behaviors, or (3) a partial ordering over trajectories. Given these three task types, we then examine the expressivity of reward.

¹This is an extended abstract of a paper [Abel et al., 2021] that won an outstanding paper award at NeurIPS 2022.
vide guidance into when it is necessary to draw on alternative formulations of a problem. In light of our focus on Markov rewards, we treat a reward function as accurately expressing a task just when the optimal value function it induces in an environment adheres to the constraints of the given task.

Main Results. We find that, for all three task types, there are environment–task pairs for which there is no Markov reward function that realizes the task (Theorem 3.1). In light of this finding, we design polynomial-time algorithms that can determine, for any given task and environment, whether a reward function exists in the environment that captures the task (Theorem 3.2). When such reward functions do exist, the algorithms also return one of them. In this sense, we take the perspective that these algorithms can be used to identify when the given representation of state might be insufficient for learning to solve the desired task. We take these findings to shed light on the nature of reward maximization as a principle, and highlight pathways for further investigation.

Background. RL defines the problem facing an agent that learns to improve its behavior over time by interacting with its environment. We make the typical assumption that the RL problem is well modeled by an agent interacting with a finite Markov Decision Process (MDP), defined by the tuple $(S,A,R,T,\gamma, s_0)$. An MDP gives rise to deterministic behavioral policies, $\pi : S \rightarrow A$, and the value, $V^\pi : S \rightarrow \mathbb{R}$, and action–value, $Q^\pi : S \times A \rightarrow \mathbb{R}$, functions that measure their quality. We will refer to a Controlled Markov Process (CMP) as an MDP without a reward function, which we denote $E$ for environment. We assume that all reward functions are deterministic, and may be a function of either state, state-action pairs, or state-action-state triples, but not history. Henceforth, we simply use “reward function” to refer to a deterministic Markov reward function for brevity, but note that more sophisticated settings beyond MDPs and deterministic Markov reward functions are important directions for future work. For more on MDPs or RL, see the books by Puterman [2014] and Sutton and Barto [2018] respectively. We also note that there is considerable relevant literature that focuses on understanding reward, and refer readers to Abel et al. [2021] for a full exposition of related work.

2 Three Task Types: SOAPs, POs, and TOs

Consider an onlooker, Alice, and a learning agent, Bob, engaged in the interaction pictured in Figure 1. Suppose that Alice has a particular task in mind that she would like Bob to learn to solve, and that Alice constructs a reward function to incentivize Bob to pursue this task. Here, Alice is playing the role of “all of what we mean by goals and purposes” for Bob to pursue, with Bob playing the role of the standard reward-maximizing RL agent. That is, we suppose Alice might think of one of three kinds of task: 1) A set of acceptable policies, 2) A partial ordering over policies, or 3) A partial ordering over trajectories. We adopt these three as they can capture many kinds of task while also allowing a great deal of flexibility in the level of detail of the specification.

Set Of Acceptable Policies. A classical view of the equivalence of two reward functions is based on the optimal policies they induce. For instance, Ng et al. [1999] develop potential-based reward shaping by inspecting which shaped reward signals will ensure that the optimal policy is unchanged. Extrapolating, it is natural to say that for any environment $E$, two reward functions are equivalent if the optimal policies they induce in $E$ are the same. In this way, a task is viewed as a choice of optimal policy. However, this notion of task fails to allow for the specification of the quality of other behaviors. For this reason, we generalize task-as-optimal-policy to a set of acceptable policies, defined as follows.

Definition 2.1. A set of acceptable policies (SOAP) is a non-empty subset of the deterministic policies, $\Pi_G \subseteq \Pi$, with $\Pi$ the set of all mappings from $S$ to $A$ for a given $E$.

With one task type defined, it is important to address what it means for a reward function to properly realize or express a task in a given environment. We offer the following account.

Definition 2.2. A reward function is said to realize a task $\mathcal{T}$ in an environment $E$ just when the start-state value (or trajectory-return) induced by the reward function exactly adheres to the constraints of $\mathcal{T}$.

Precise conditions for the realization of each task type are provided alongside each task definition, with a summary presented in column four of Table 1. For SOAPs, we take the start-state value $V^\pi(s_0)$ to be the mechanism by which a reward function realizes a SOAP. That is, for a given $E$ and $\Pi_G$, a reward function $R$ is said to realize $\Pi_G$ in $E$ when the start-state value function is optimal for all good policies, and strictly higher than the start-state value of all other policies. It is clear that SOAP strictly generalizes a task in terms of a choice of optimal policy, as captured by the SOAP $\Pi_G = \{\pi^*_G\}$.

We note that there are two natural ways for a reward function to realize a SOAP: First, each $\pi_g \in \Pi_G$ has optimal start-state value and all other policies are sub-optimal. We call this type equal-SOAP, or just SOAP for brevity. Alternatively, we might only require that the acceptable policies are each near-optimal, but are allowed to differ in start-state value so long as they are all better than every bad policy $\pi_b \in \Pi_B$. That is, in this second kind, there exists an $\epsilon > 0$ such that every $\pi_g \in \Pi_G$ is $\epsilon$-optimal in start-state value, $V^\pi(s_0) - V^{\pi_g}(s_0) \leq \epsilon$, while all other policies are worse. We call this second realization condition range-SOAP. We note that the range realization generalizes the equal one: Every equal-SOAP is a range-SOAP (by letting $\epsilon = 0$). However, there exist range-SOAPs that are expressible by Markov rewards that are not realizable as an equal-SOAP. We illustrate this fact with the following proposition. All proofs are presented in the appendix of the full version of the paper.

Proposition 2.1. There exists a CMP, $E$, and choice of $\Pi_G$ such that $\Pi_G$ can be realized under the range-SOAP criterion, but cannot be realized under the equal-SOAP criterion.

One such CMP is pictured in Figure 2b. Consider the SOAP $\Pi_G = \{\pi_{11},\pi_{12},\pi_{21}\}$, where $\pi_{21}$ denotes the policy $\{s_0 \mapsto a_2, s_1 \mapsto a_1\}$: Under the equal-SOAP criterion, if each of these three policies are made optimal, any reward function will also make $\pi_{22}$ (the only bad policy) optimal as well. In contrast, for the range criterion, we can choose a
Table 1: A summary of the three proposed task types. We further list the constraints that determine whether a reward function realizes each task type in an MDP, where we take ⊕ to be one of ’<’, ’>’, or ’=’, and G is the discounted return of the trajectory.

<table>
<thead>
<tr>
<th>Name</th>
<th>Notation</th>
<th>Generalizes</th>
<th>Constraints Induced by $\mathcal{F}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOAP</td>
<td>$\Pi_G$</td>
<td>task-as-$\pi^*$</td>
<td>equal: $V^{\pi_1}(s_0) = V^{\pi_2}(s_0) &gt; V^{\pi_3}(s_0)$, $\forall \pi_1, \pi_2, \pi_3 \in \Pi_G, \pi_1 \neq \Pi_B$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>range: $V^{\pi_1}(s_0) &gt; V^{\pi_2}(s_0)$, $\forall \pi_1, \pi_2 \in \Pi_G, \pi_1 \neq \Pi_B$</td>
</tr>
<tr>
<td>PO</td>
<td>$L_{\Pi}$</td>
<td>SOAP</td>
<td>$(\pi_1 + \pi_2) \in L_{\Pi} \implies V^{\pi_1}(s_0) + V^{\pi_2}(s_0)$</td>
</tr>
<tr>
<td>TO</td>
<td>$L_{\tau,N}$</td>
<td>task-as-goal</td>
<td>$(\tau_1 + \tau_2) \in L_{\tau,N} \implies G(\tau_1; s_0) + G(\tau_2; s_0)$</td>
</tr>
</tbody>
</table>

3 Results

With our definitions and objectives in place, we now present our main results.

3.1 Expressing SOAPs, POs, and TOs

We first ask whether Markov reward can always realize a given SOAP, PO, or TO, for an arbitrary $E$. Our first result states that the answer is “no”.

Theorem 3.1. For each task type, there exist $(E, \mathcal{F})$ pairs for which no Markov reward function realizes $\mathcal{F}$ in $E$.

Thus, Markov reward is incapable of capturing certain tasks. What tasks are they, precisely? Intuitively, inexpressible tasks involve policies or trajectories that are correlated in value in an MDP. That is, if two policies are nearly identical in behavior, it is unlikely that reward can capture the PO that places them at opposite ends of the ordering. A simple example is the “always move the same direction” task in a grid world, with state defined as an $(x, y)$ pair. The SOAP $\Pi_G = \{\pi_{\rightarrow}, \pi_{\uparrow}, \pi_{\rightarrow}, \pi_{\downarrow}\}$ conveys this task, but no Markov reward function can make these policies strictly higher in value than all others.

Inexpressible SOAPs. Observe the two CMPs pictured in Figure 2a and Figure 2b, depicting two kinds of inexpressible SOAPs. In the top figure, we consider the SOAP $\Pi_G = \{\pi_{21}\}$, containing only the policy that executes $a_2$ in the left state $(s_0)$, and $a_1$ in the right $(s_1)$. This SOAP is inexpressible through reward, but only because reward cannot distinguish the start-state value of $\pi_{21}$ and $\pi_{22}$ since the policies differ only in an unreachable state. In the bottom figure, we find a more interesting case: The chosen SOAP is similar to the XOR function, $\Pi_G = \{\pi_{12}, \pi_{21}\}$. Here, the task requires that the agent choose each action in exactly one state. However, there cannot exist a reward function that makes only these policies optimal, as by consequence, both policies $\pi_{11}$ and $\pi_{22}$ must be optimal as well.

3.2 Constructive Algorithms: Task to Reward

We now analyze how to determine whether an appropriate reward function can be constructed for any $(E, \mathcal{F})$ pair. We pose a general form of the reward-design problem [Mataric, 1994; Sorg et al., 2010; Dewey, 2014] as follows.

Definition 3.1. The Reward Design problem is: Given $E = (S, A, T, \gamma, s_0)$, and a $\mathcal{F}$ output a reward function $R$ such that $E, R$ and $\mathcal{F}$.

Indeed, for all three task types, there is an efficient algorithm for solving the reward-design problem.
Figure 2: (Top Row) Two CMPs in which there is a SOAP that is not expressible under any Markov reward function. On the left, $\Pi_G = \{\pi_{21}\}$ is not realizable, as $\pi_{21}$ can not be made better than $\pi_{22}$ because $s_1$ is never reached. On the right, the XOR-like-SOAP, $\Pi_G = \{\pi_{12}, \pi_{21}\}$ is not realizable: To make these two policies optimal, it is entailed that $\pi_{22}$ and $\pi_{11}$ must be optimal, too. (Bottom Row) The approximate fraction of SOAPs that are expressible by reward in CMPs with a handful of states and actions, with 95% confidence intervals. In each plot, we vary a different parameter of the environment or task to illustrate how this change impacts the expressivity of reward, showing both equal (color) and range (grey) realization of SOAP.

**Theorem 3.2.** The REWARDDESIGN problem can be solved in polynomial time, for any finite $E$, and any $T$, so long as reward functions with infinitely many outputs are considered.

Therefore, for any choice of $E$ and a SOAP, PO, or TO, we can find a reward function that perfectly realizes the task in the given environment, if such a reward function exists. Each of the three algorithms are based on forming a linear program that matches the constraints of the given task type, which is why reward functions with infinitely many outputs are required.

### 3.3 Experiments

Lastly, we conduct an experiment to shed further light on the analysis. Our focus is on SOAPs, though we anticipate the insights extend to POs and TOs with little complication.

**SOAP Expressivity.** Concretely, we estimate the fraction of SOAPs that are expressible by Markov reward in small CMPs as we vary expects of the environment or task. For each data point, we sample 200 random SOAPs and run the algorithm mentioned in Theorem 3.2 to determine whether each SOAP is realizable in the given CMP. We ask this question for both the equal (color) and range (grey) SOAP realization. We inspect SOAP expressivity as we vary three different characteristics of $E$ or $\Pi_G$: The number of good policies in each SOAP, the Shannon entropy of $T$ at each $(s, a)$ pair, and the “spread” of each SOAP. The spread approximates average edit distance among policies in $\Pi_G$ determined by randomly permuting actions of a reference policy by a coin weighted according to the value on the x-axis. We use the same set of CMPs for each environment up to any deviations explicitly made by the varied parameter (such as entropy). Unless otherwise stated, each CMP has four states and three actions, with a fixed but randomly-chosen transition function.

Results are presented in Figure 2. We find that our theory is borne out in a number of ways. First, as Theorem 3.1 suggests, we observe that SOAP expressivity is strictly less than one in nearly all cases. This is evidence that inexpressible tasks are not only found in manufactured corner cases, but rather that expressivity is a spectrum. We further observe—as predicted by Proposition 2.1—separation between the expressivity of range-SOAP (grey) vs. equal-SOAP (color); there are many cases where we can find a reward function that makes the good policies near-optimal and better than the bad, but cannot make those good policies all exactly-optimal. Additionally, several trends emerge as we vary the parameter of environment or task, though we note that such trends are likely specific to the choice of CMP and may not hold in general. Perhaps the most striking trend is in Figure 2e, which shows a decrease in expressivity as the SOAPs become more spread-out.

### 4 Conclusion

We here examine the expressivity of Markov reward, framed around three accounts of task. Our main results show that there exist choices of task and environment in which Markov reward cannot express the chosen task, but there are efficient algorithms that decide whether a task is expressible and construct a realizing reward function when it exists. We take these to be first steps toward understanding the full scope of the reward hypothesis, and hope this work provides new conceptual perspectives on reward and its place in RL and AI.
Acknowledgments

The authors would like to thank André Barreto, Diana Borsa, Michael Bowling, Wilka Carvalho, Brian Christian, Jess Hamrick, Steven Hansen, Zac Kenton, Ramana Kumar, Katrina McKinney, Rémi Munos, Brendan O’Donoghue, Matt Overlan, Mark Rowland, Hado van Hasselt, and Ben Van Roy for helpful discussions. We would also like to thank the anonymous NeurIPS reviewers for their thoughtful feedback.

References


