On the Expressivity of Markov Reward

By Abel et.al.

Presenter: Yibin Xiong

December 7, 2021

Reward Engineering as a 2-phase Problem

- Reward hypothesis: “All goals and purposes can be well thought of as *maximization of the expected total reward.*”
- Is it true? How do we figure out the appropriate reward function?
- TaskQ: How do we define/specify a task? (natural language, an optimal policy, etc.)
- ExpressionQ: Given the task definition, can we design a (Markov) reward function that fully expresses the task?

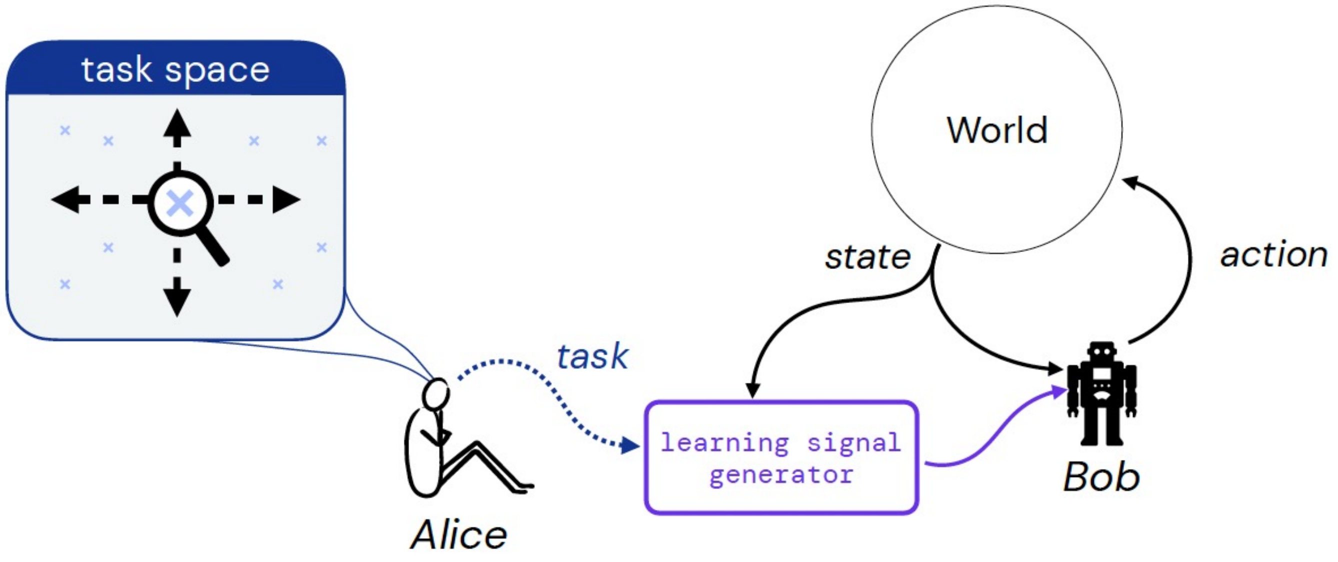


Figure 1: Alice, Bob, and the artifacts of task definition (blue) and task expression (purple).

*In this paper, we discuss deterministic policies and reward functions

TaskQ: Definitions of Tasks

- Task-as-π^*
  - *The most "coarse" specification of tasks; Cannot differentiate all other sub-optimal policies*
- SOAP (Set of Acceptable Policies)
  - Partitioned the policy space into 2 equivalent classes: good policies Π_g and bad policies Π_b
- PO (Partial Ordering on Policies)
  - Partitioned the policy space into several equivalent classes

- Ta
- T Traje
  - -horiz
  - *the a e. proc*

Realizability and Task Constraints

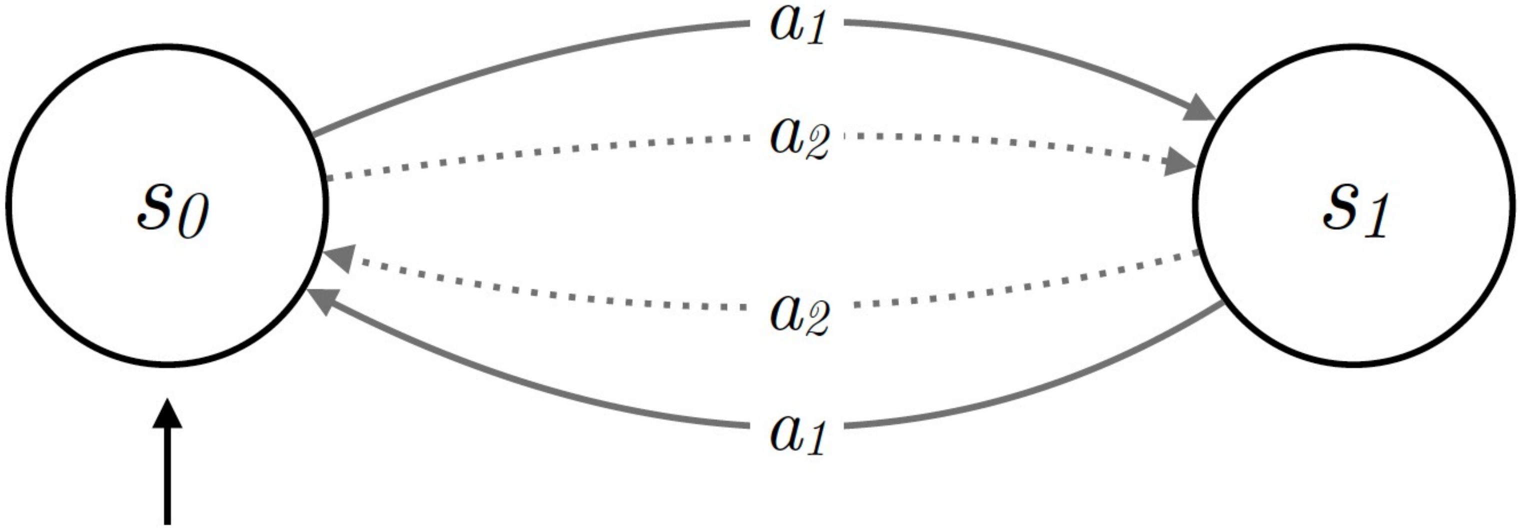
- A reward function *realizes* the given task $\mathcal{T}$ in an environment $E = \{S, A, T, \gamma, s_0\}$ if it satisfies the constraints induced by $\mathcal{T}$.

Name	*Notation*	*Generalizes*	*Constraints Induced by $\mathscr{T}$*
SOAP	Π_G	task-as-π^*	equal: $V^{\pi_g}(s_0) = V^{\pi_{g'}}(s_0) > V^{\pi_b}(s_0), \forall_{\pi_g, \pi_{g'} \in \Pi_G, \pi_b \in \Pi_B}$ range: $V^{\pi_g}(s_0) > V^{\pi_b}(s_0), \forall_{\pi_g \in \Pi_G, \pi_b \in \Pi_B}$
PO	L_Π	SOAP	$(\pi_1 \oplus \pi_2) \in L_\Pi \implies V^{\pi_1}(s_0) \oplus V^{\pi_2}(s_0)$
TO	$L_{\tau,N}$	task-as-goal	$(\tau_1 \oplus \tau_2) \in L_{\tau,N} \implies G(\tau_1; s_0) \oplus G(\tau_2; s_0)$

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 $\oplus$ to be one of '$<$', '$>$', or '$=$', and G is the discounted return of the trajectory.

ExpressionQ: Are SOAP, PO, TO realizable?

- **Theorem 4.1** says No!
- We can find simple counter-examples:



SOAP/PO with $\Pi_g = \{\pi_{12}, \pi_{21}\}, \Pi_b = \{\pi_{11}, \pi_{22}\}$

- **Proposition 4.2** generalizes this to any transition dynamics T and any discount factor γ

Application – An Algorithm

- Determine whether a task is realizable
- If so, find a reward function that realizes the task
- **Theorem 4.3**: The REWARDDESIGN problem can be solved in *polynomial* time, for any *finite* E, and any SOAP, PO, or TO, so long as a reward-function family with infinitely many outputs is used.
- This is because we can formulate the constraints into a *linear programming* (LP) problem

For example, we consider a $SOAP$ task defined by $\Pi_g = \{\pi_{g_1}, \ldots, \pi_{g_m}\}$ and $\Pi_b = \{\pi_{b_1}, \ldots, \pi_{b_n}\}$.

The LP problem can be

$$
\begin{aligned}
\underset{r_1,\ldots,r_{|S||A|}}{min} \quad & 0 \\
\text{s.t.} \quad V^{\pi_{g1}}(s_0) &= V^{\pi_{g2}}(s_0) \\
V^{\pi_{g2}}(s_0) &= V^{\pi_{g3}}(s_0) \\
&\vdots \\
V^{\pi_{gm-1}}(s_0) &= V^{\pi_{gm}}(s_0) \\
V^{\pi_{g1}}(s_0) &> V^{\pi_{b1}}(s_0) \\
&\vdots \\
V^{\pi_{g1}}(s_0) &> V^{\pi_{bn}}(s_0)
\end{aligned}
$$

Algorithm

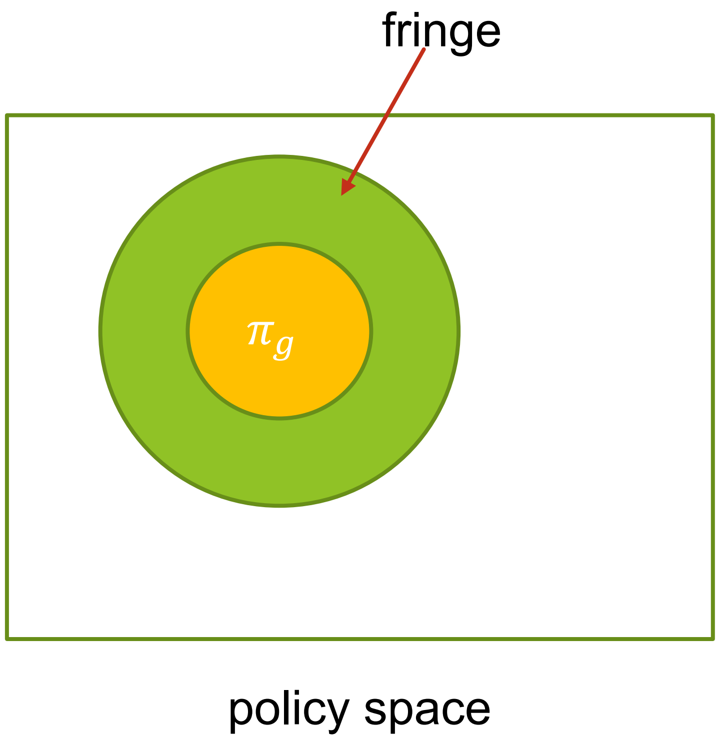
- Trick 1: only "fringe constraints" (for SOAP)
  - Fringe policies deviates from the optimal policies by only 1 action
  - By "policy improvement theorem," policies outside the fringe

  have lower start-state values
- Trick 2: estimate start-state values
  - Define the discounted expected state-action visitation distribution

$$\rho_i(s,a) := \sum_{t=0}^{\infty} \gamma^t \Pr(s_t = s, a_t = a \mid s_0, \pi_i).$$

  - Then $V^{\pi_i}(s_0) = R^T \rho_i$
  - If the state space and/or action space are continuous, R is a vector with infinitely many entries, i.e. a continuous function
- Trick 3: add slack variables to convert $<$ to $\leq$

$$R^T\rho_i < R^T\rho_j \;\Rightarrow\; R^T\rho_i + \epsilon_k \leq R^T\rho_j,\; \epsilon_k \geq 0$$



Algorithm 1 SOAP Reward Design

INPUT: $E = (\mathcal{S}, \mathcal{A}, T, \gamma, s_0), \Pi_G$.
OUTPUT: R, or $\perp$.

$\Pi_{\text{fringe}} = \texttt{compute\_fringe}(\Pi_G)$
for $\pi_{g,i} \in \Pi_G$ **do** ▷ Compute state-visitation distributions.
    $\rho_{g,i} = \texttt{compute\_exp\_visit}(\pi_{g,i}, E)$

for $\pi_{f,i} \in \Pi_{\texttt{fringe}}$ **do**
    $\rho_{f,i} = \texttt{compute\_exp\_visit}(\pi_{f,i}, E)$

$C_{\text{eq}} = \{\}$ ▷ Make Equality Constraints.
for $\pi_{g,i} \in \Pi_G$ **do**
    $C_{\text{eq}}.\texttt{add}(\rho_{g,0}(s_0) \cdot X = \rho_{g,i}(s_0) \cdot X)$

$C_{\text{ineq}} = \{\}$ ▷ Make Inequality Constraints.
for $\pi_{f,j} \in \Pi_{\text{fringe}}$ **do**
    $C_{\text{ineq}}.\texttt{add}(\rho_{f,j}(s_0) \cdot X + \epsilon \leq \rho_{g,0}(s_0) \cdot X)$

$R_{\text{out}}, \epsilon_{\text{out}} = \texttt{linear\_programming}(\text{obj.} = \max \epsilon, \text{constraints} = C_{\text{ineq}}, C_{\text{eq}})$ ▷ Solve LP.

if $\epsilon_{\text{out}} > 0$ **then** ▷ Check if successful.
    return R_{out}
else
    return $\perp$

- Runtime: $\mathcal{O}(N^3)$, where $N \leq |A|^{|S|}$ or $N = \max\{|S|, |A|\}$

More Theoretical Results

- **Theorem 4.5**: When we require the reward function has *finitely* many outputs (i.e. the number of possible values that each entry can take is finite), the problem becomes NP-hard.

- **Proposition 4.6**: For any SOAP, PO, or TO, given a finite set of CMPs, $\mathcal{E} = \{E_1, \ldots, E_n\}$ with shared state–action space, there exists a *polynomial* time algorithm that outputs one reward function that realizes the task (when possible) in all CMPs in $\mathcal{E}$.

In other words, transition dynamics and γ do not affect the realizability much.

- **Theorem 4.7**: Task realization is not closed under sets of CMPs with shared state-action space.

That is, there exist choices of $\mathcal{T}$ and $\mathcal{E} = \{E_1, \ldots, E_n\}$ such that $\mathcal{T}$ is realizable in each $E_i \in \mathcal{E}$ *independently*, but there is not a single reward function that realizes $\mathcal{T}$ in all $E_i \in \mathcal{E}$ *simultaneously*.

Experiments: 4 state 3 action MDP

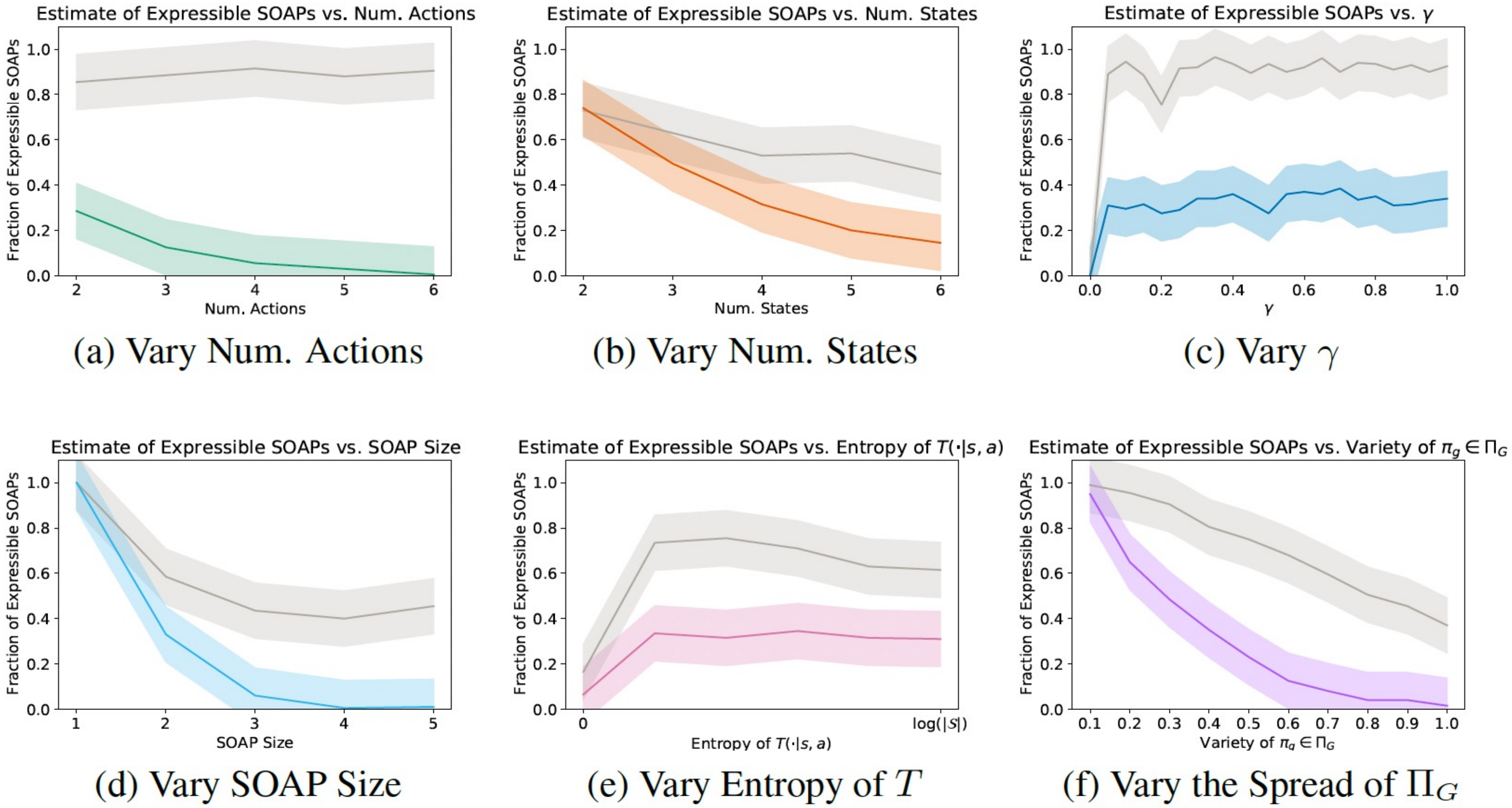
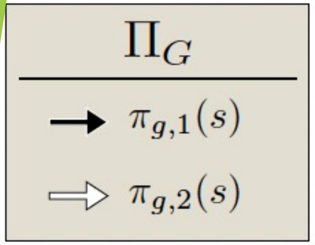
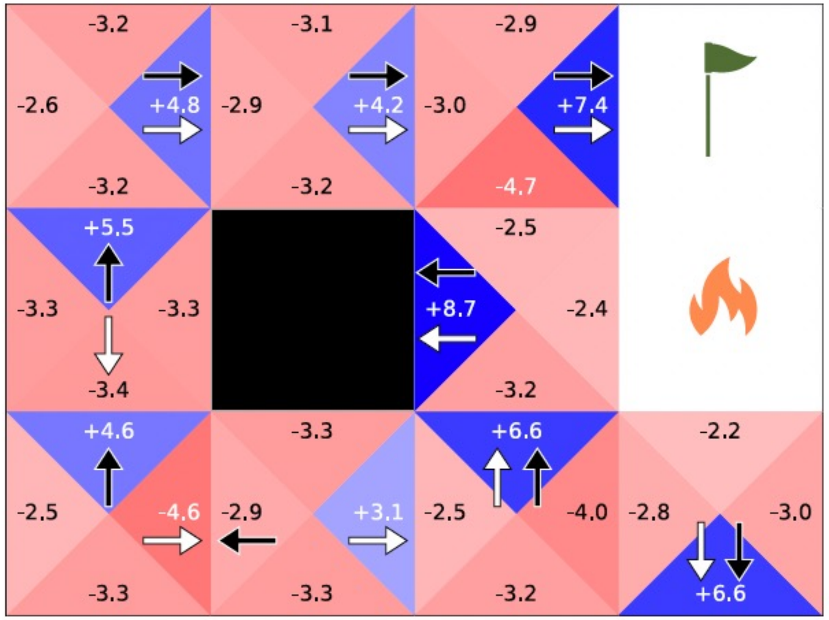


Figure 3: The approximate fraction of SOAPs that are expressible by reward in CMPs with a handful of states and actions, with 95% confidence intervals. In each plot, we vary a different parameter of the environment or task to illustrate how this change impacts the expressivity of reward, showing both equal (color) and range (grey) realization of SOAP.

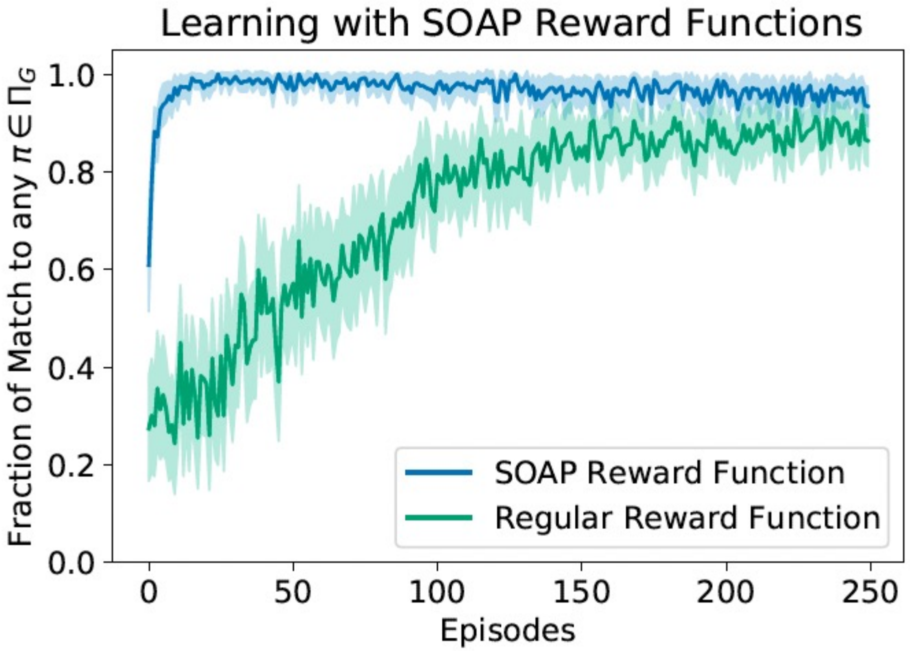
*SOAP size refers to the number of good policies in Π_g

Experiments





(a) Grid World SOAP Reward



(b) Grid World Learning

- Faster convergence
- The ϵ-greedy policy when using “regular reward function” does not converge exactly to an optimal policy

Thoughts, Questions, Future Directions

- Comparison with IRL
  - Maximize ϵ is similar to the maximize the margin in traditional IRL (feature matching); non-parametric
  - LP problem formulation is simple and the algorithm is not iterative (we can have iterative versions that enables interaction between reward designing and policy optimization)
  - PO has stronger prior knowledge because of more fine-grained partition, but also more computation in fitting $\rho_i(s, a)$.
- Feasibility:
  - Toy-example MDPs in the experiments; continuous state/action space $\Rightarrow$ deep neural nets
  - If $|\Pi_g|$ is large, we need to 1) specify all of them (SOAP equal); 2) fit $\rho(s, a)$ for *each* one
  - For PO/TO, specify a lot of orders
  - Assumed perfect task knowledge, but difficult to transfer task knowledge/finetune a reward function when tasks are defined by “policy/trajectory inequalities”
- For TO, what if the good trajectories have different length?

Thoughts, Questions, Future Directions

- Sufficient conditions for realizability?
- Other formulation of tasks
  - Functional Task descriptions: SOAP, PO and TO are very general descriptions of task definitions, but task descriptions from a functional standpoint represent completion of goals or description of path constraints. While the definitions are general enough to capture all possible tasks, the set of 'practical' tasks might be a smaller subset of these, and Markov rewards might represent a larger fraction of these tasks.
  - Regarding functional task descriptions; This point resonates with us as well; it is likely useful to carefully isolate the conditions on SOAP/PO/TO that ensure Markov rewards are sufficient (as suggested by another reviewer, too). These particular subsets of SOAPs/POs/TOs might be of interest on their own. We believe this is another useful direction for future work
- Convex rather than linear
  - Another interesting direction for future works may study the expressivity of the convex MDP model (see, Zhang et al., Variational policy gradient method for reinforcement learning with general utilities, 2020; Zahavy et al., Reward is enough for convex MDPs, 2021), which allows to specify the objective as any convex function of the expected state visitations rather than a linear combination.
- For tasks that are not realizable, what can we do?

Summary

The authors

- Proposed clear definitions of tasks
- Examined the expressivity of Markov rewards
- Explored variants of the reward design problem (finite output, shared state & action space)
- Framed the problem into LP and solved it in polynomial time

Thank you!