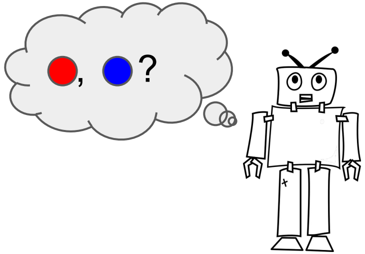
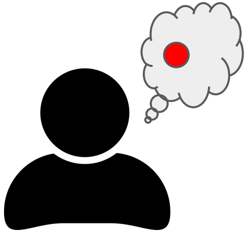
Approximately-Optimal Queries for Planning in Reward-Uncertain Markov Decision Processes

Shun Zhang, Edmund Durfee, and Satinder Singh
University of Michigan

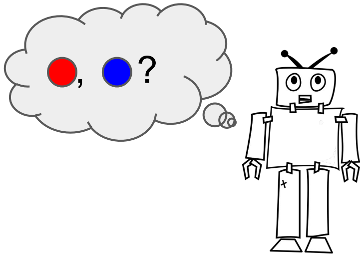




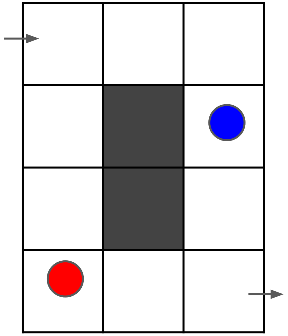
Human-Robot Interaction (HRI)

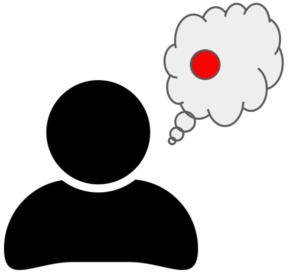
The robot takes actions on behalf of a human and **wants to maximize the human's utility**.

The robot is **uncertain** about the reward function in the human's mind.



Robot



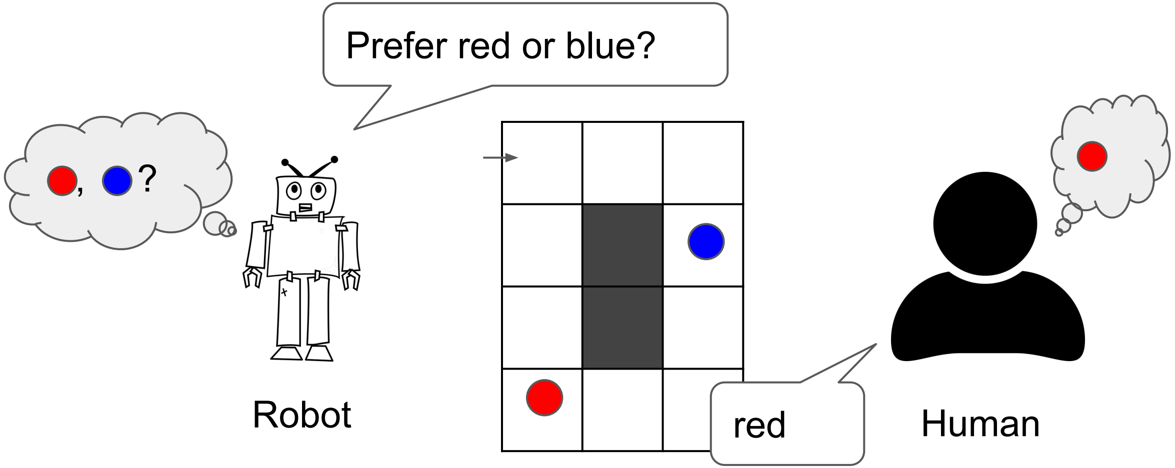


Human

Our Setting

We don't want the human to exert much effort in informing the robot.

We consider the setting that the robot asks a ***k*-response query** and receives the human's response before it plans.



Our Contributions

1. A more efficient algorithm to find an **approximately optimal *k*-response policy query** than Viappiani and Boutilier (2010) in reward-uncertain MDPs.
2. A way to find **trajectory queries** that outperform competing methods in the literature under certain conditions.

Reward-Uncertain MDPs

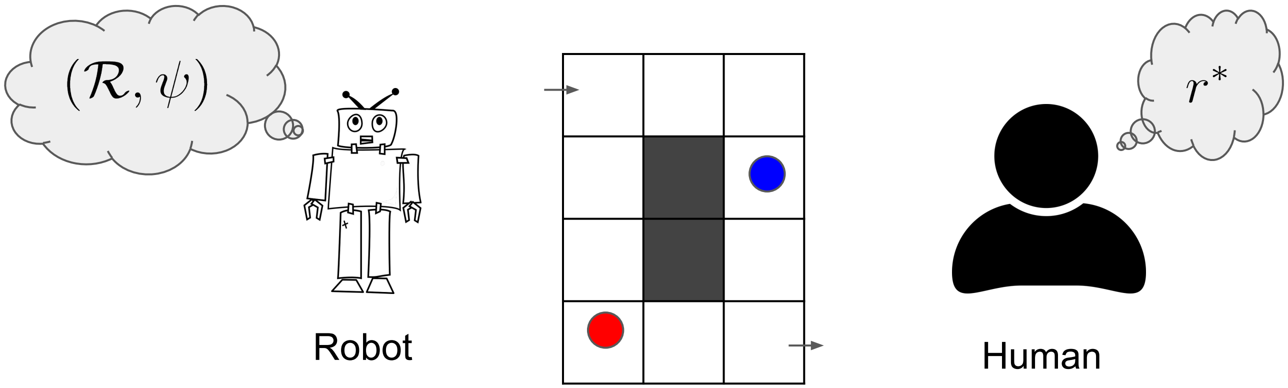
Reward candidates

Prior belief about which is the true reward function

Reward-uncertain MDP: $\langle S, A, T, \boxed{(\mathcal{R}, \psi)}, s_0 \rangle$

$$\mathcal{R} = \{r_1, r_2, \ldots, r_n\}, \psi \in \Delta_{\mathcal{R}}$$

The true reward function $r^* \in \mathcal{R}$



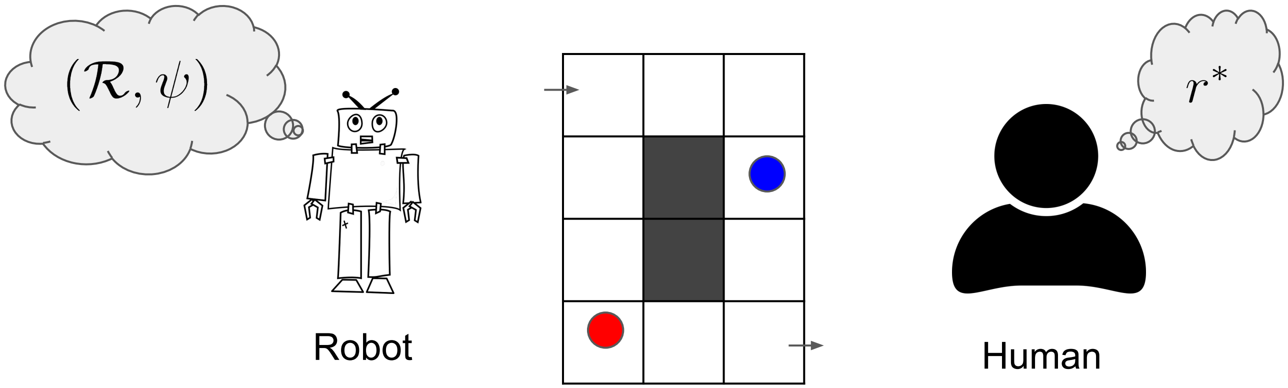
k-Response Queries

k-response query: $q = \{j_1, j_2, \ldots, j_k\}$

The human responds with *j* in *q*

Response model: $\mathbb{P}[q \rightsquigarrow j | r]$

Posterior update: $\mathbb{P}[r | q \rightsquigarrow j; \psi] \sim \mathbb{P}[q \rightsquigarrow j | r]\mathbb{P}[r; \psi]$



Our First Contribution (Detailed)

A more efficient algorithm to find an **approximately optimal *k*-response policy query** than Viappiani and Boutilier (2010) in reward-uncertain MDPs.

Expected Value of Information — Viappiani and Boutilier (2010)

↓

No loss if we only consider policy queries — Cohn et al. (2014)

↓

A greedy construction algorithm (for one-shot decision making) — Viappiani and Boutilier (2010)

↓

A greedy construction algorithm (for reward-uncertain MDPs) — Our work!

Expected Value of Information

(e.g. Viappiani and Boutilier, 2010).

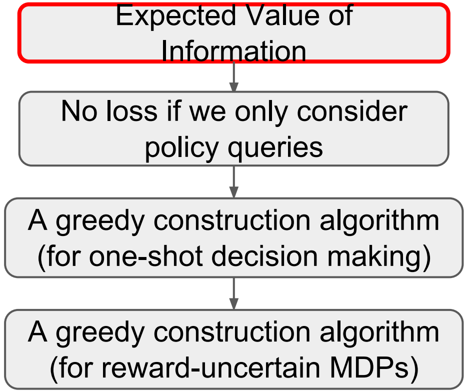
$$EVOI(q, \psi) = \sum_{j \in q} P[q \rightsquigarrow j; \psi] V^*_{\psi | q \rightsquigarrow j} - V^*_{\psi}$$

The prior optimal value

The probability that the human responds with j

The posterior optimal value when the human responds with j

Find the optimal *k*-response query?



Expected Value of Information

No loss if we only consider policy queries

A greedy construction algorithm (for one-shot decision making)

A greedy construction algorithm (for reward-uncertain MDPs)

Find the optimal ~~*k*-response query~~?

k-response **policy** query

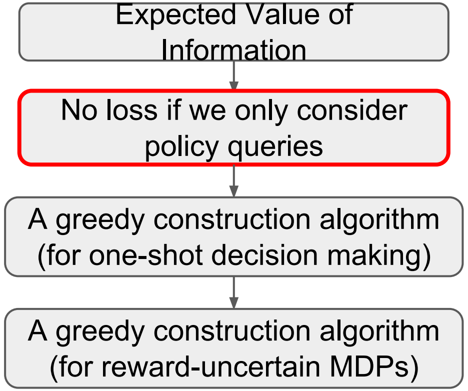
A *k*-response policy query: $q = \{\pi_1, \pi_2, \ldots, \pi_k\}$

Response model: $q \rightsquigarrow \pi \implies V^{\pi}_{r^*} \geq V^{\pi'}_{r^*}, \forall \pi' \in q$

Observation: An optimal *k*-response **policy query** is an optimal *k*-response **query**. (derived from Cohn et al. (2014))

$$q^* = \{j_1, j_2, \ldots, j_k\}$$

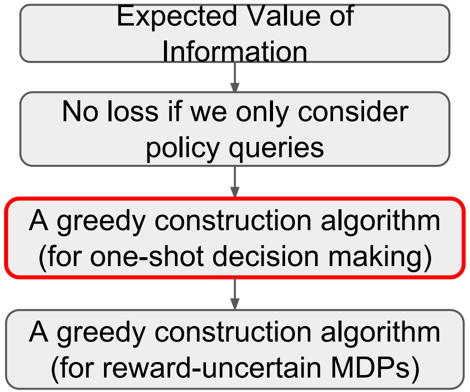
$$\{\pi_1, \pi_2, \ldots, \pi_k\}$$



Find the optimal *k*-response policy query

Enumerate all policy queries to find the optimal one?

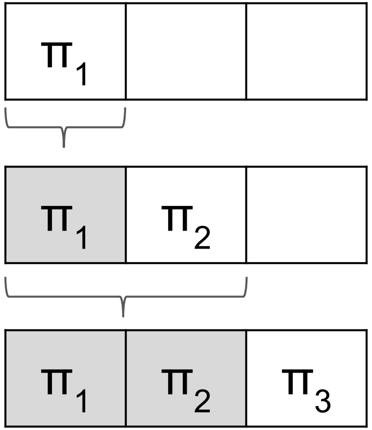
- Computationally intractable.
  The number of deterministic policies: $|A|^{|S|}$



Find the optimal *k*-response policy query

(or an approximately optimal one?)

Greedy construction (Viappiani and Boutilier, 2010).



- Add one policy at a time.
- Approximate optimality.
- Still computationally intractable. Need to enumerate all policies!

Contribution: A more efficient way to find the next policy to add in the greedy construction method.

Expected Value of Information

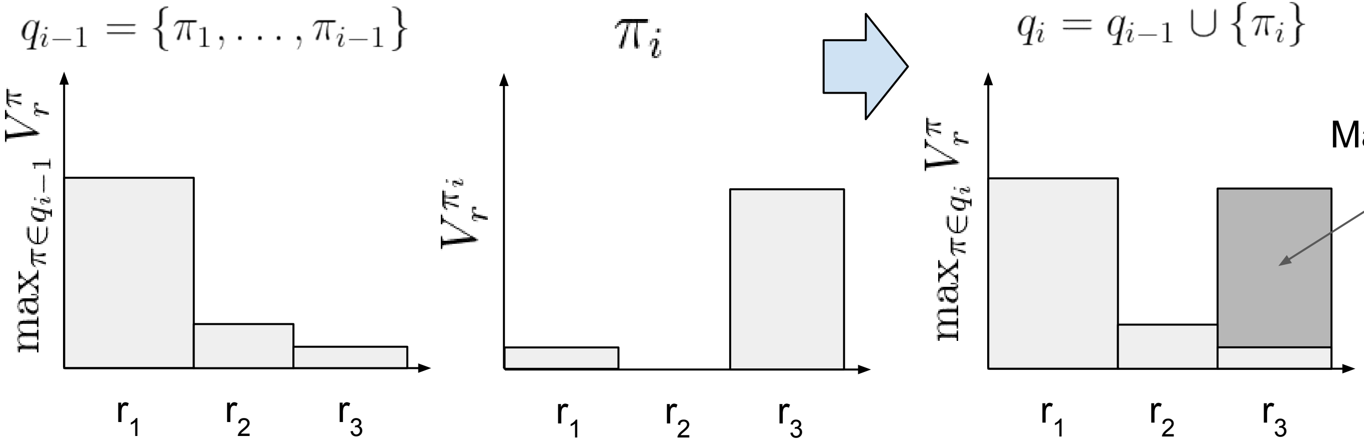
No loss if we only consider policy queries

A greedy construction algorithm (for one-shot decision making)

A greedy construction algorithm (for reward-uncertain MDPs)

Greedy Construction of Policy Queries

Intuition of the problem: How to find the policy that best *complements* the added policies?



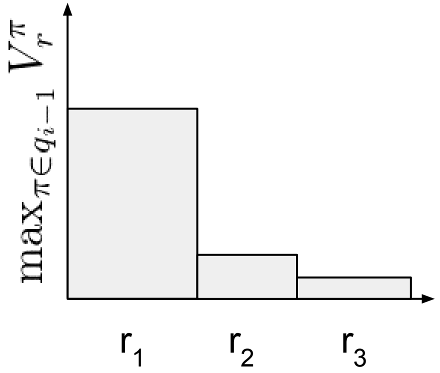
Finding the next policy can be formulated as a mixed integer linear programming (MILP) problem.

Greedy Construction: Algorithm

Equivalent to maximizing the **expected utility of selection** function (*EUS*).

$$EUS(q, \psi) = \sum_{r \in \mathcal{R}} \mathbb{P}[r; \psi] \max_{\pi \in q} V_r^\pi$$

$$q_{i-1} = \{\pi_1, \ldots, \pi_{i-1}\}$$



Algorithm 1 Greedy construction of policy queries

$q_0 = \emptyset$
for $i = 1, 2, \ldots, k$ **do**
    $\pi_i \leftarrow \arg\max_\pi EUS(q_{i-1} \cup \{\pi\}, \psi)$
    $q_i \leftarrow q_{i-1} \cup \{\pi_i\}$
end for
return q_k

We cannot afford to enumerate all the policies.
We find π_i by solving an MILP problem (next slide).

Greedy Construction: The MILP Problem

Occupancy measure

$$\max_{x,\{y_r\},\{z_r\}} \sum_{r \in \mathcal{R}} \mathbb{P}[r;\psi] y_r$$

An arbitrary large number.

$$\text{s.t.} y_r \leq \sum_{s,a} [x(s,a) r(s,a)] - \max_{\pi \in q_{t-1}} V_r^\pi + M(1 - z_r), \forall r \in \mathcal{R}$$

$$y_r \leq 0 + M z_r, \forall r \in \mathcal{R}$$

One of them needs to be effective, toggled by z_r

$$\sum_{a'} x(s', a') = \sum_{s,a} x(s,a) T(s,a,s') + \delta(s', s_0), \forall s'$$

x is consistent with the transition function

$$x(s,a) \geq 0, \forall s, a$$

$$z_r \in \{0,1\}, \forall r \in \mathcal{R}$$

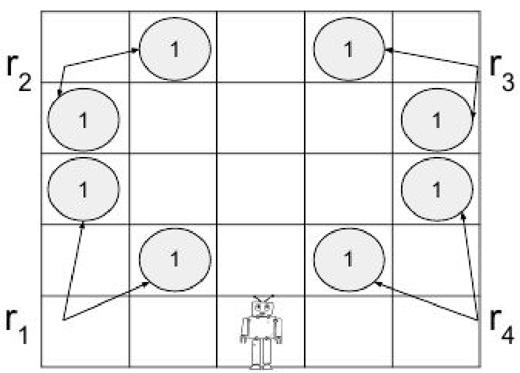
A binary variable for each reward candidate.

Empirical Evaluation

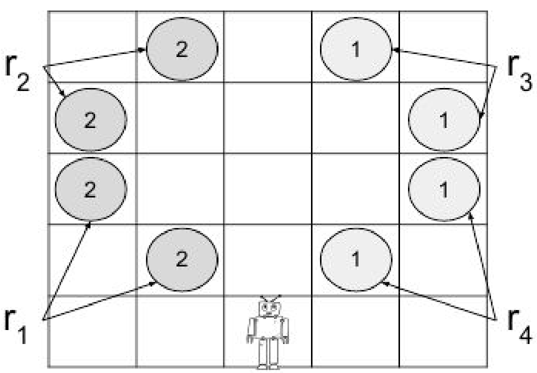
- The **EVOI** of greedily constructed policy query is close to that of the optimal policy query.
- The **computation time** of greedily constructed policy query is much shorter than that of the optimal policy query.

Rock Collection Domain

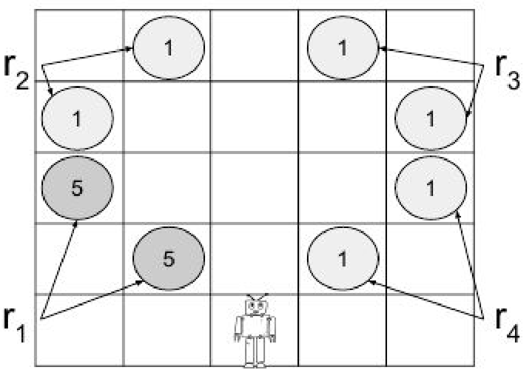
Modified from RockSample in (Smith and Simmons, 2004).



Reward Setting #1

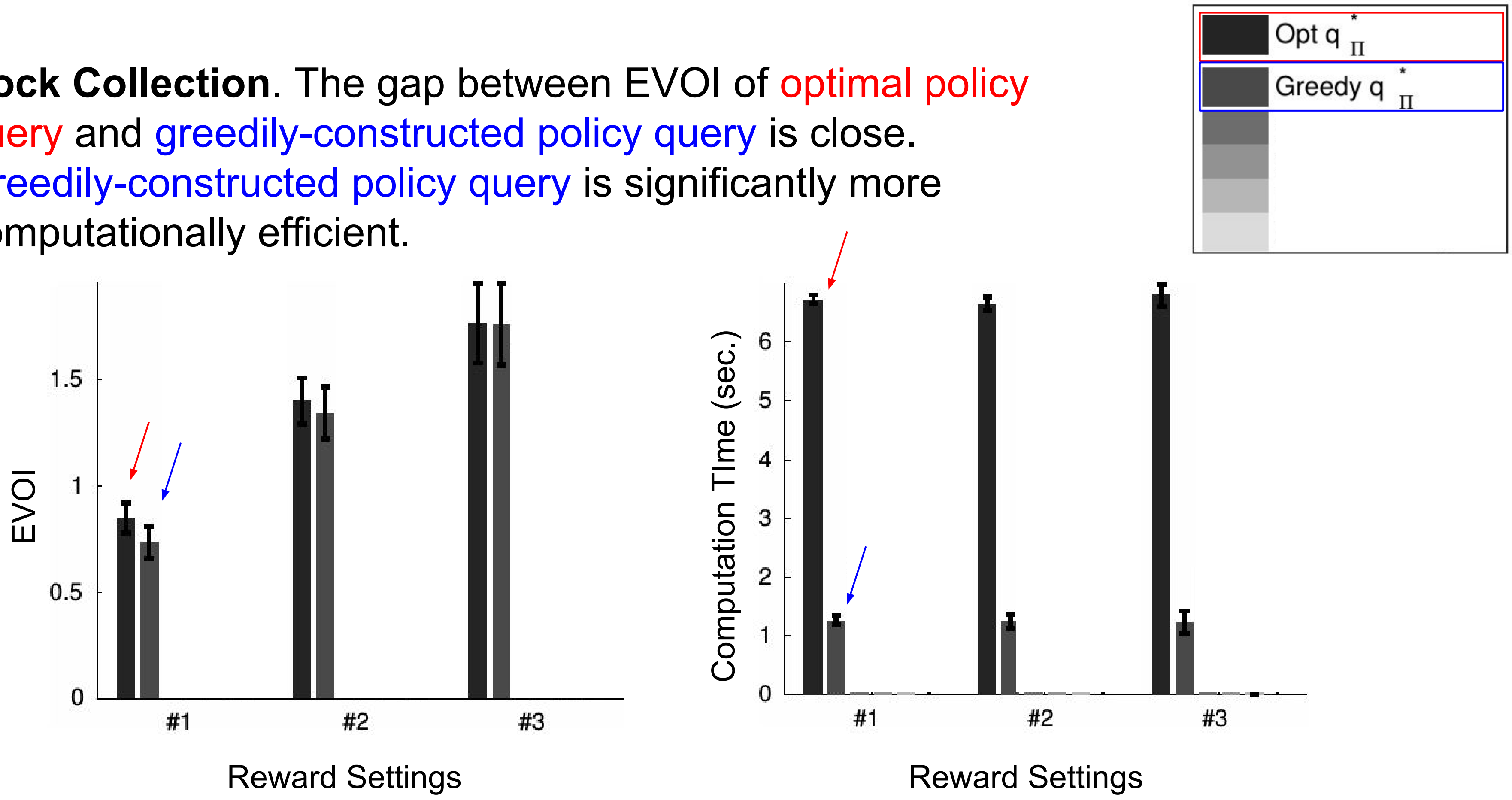


Reward Setting #2



Reward Setting #3

Rock Collection. The gap between EVOI of optimal policy query and greedily-constructed policy query is close. Greedily-constructed policy query is significantly more computationally efficient.



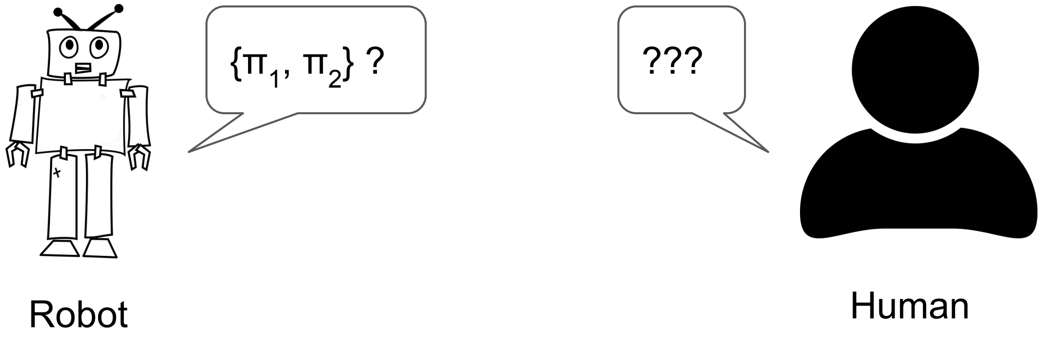
Contributions

1. A more efficient algorithm to find an **approximately optimal *k*-response policy query** than Viappiani and Boutilier (2010) in reward-uncertain MDPs.
2. A way to find **trajectory queries** that outperform competing methods in the literature under certain conditions.

Limitations of Policy Queries

- It can be inefficient to communicate a set of policies.
- Policies are not easy for human to comprehend.

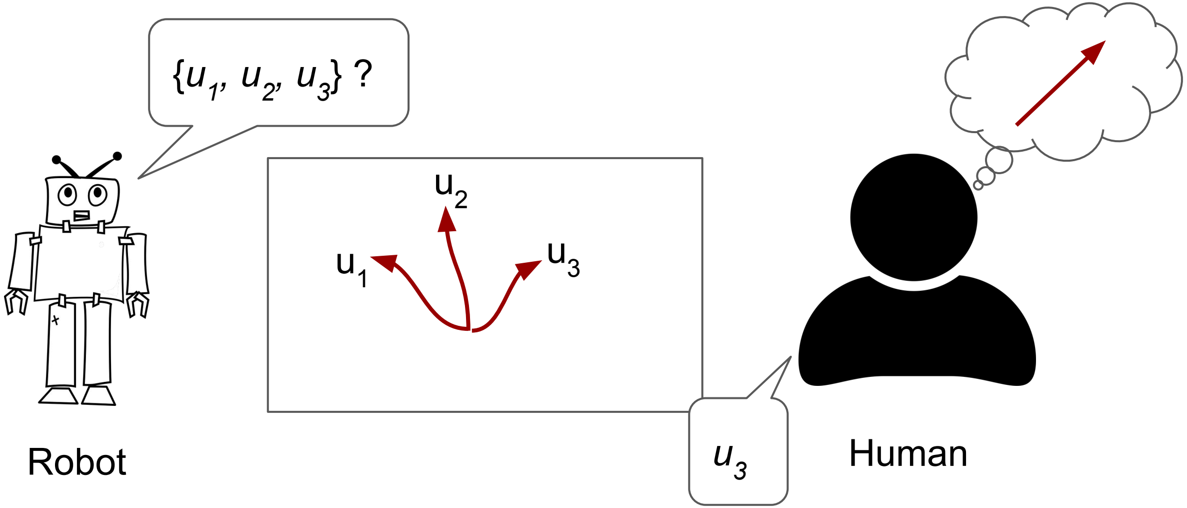
We consider how to generalize our technique to trajectory queries (e.g. Wilson et al., 2012; Akrour et al., 2012)



Trajectory Queries

A set of trajectories with a finite length starting from the same state.

Ask the human which trajectory is most similar to what he would do.

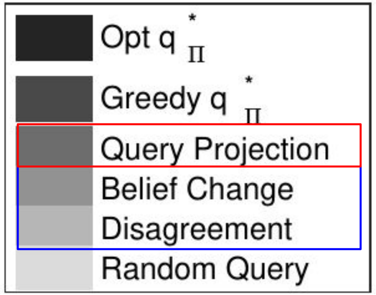


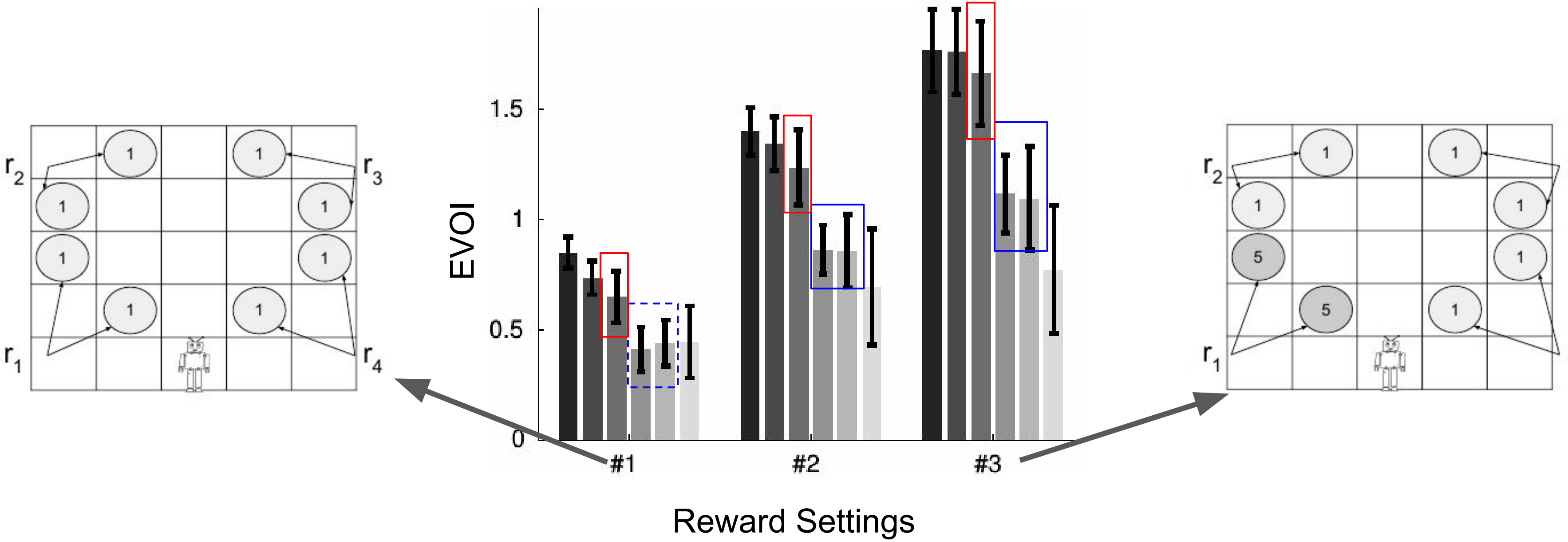
Projecting to Trajectory Queries

- Greedily construct trajectory queries?
  - Unlike policy queries, trajectory queries are not *decision queries*.
  - Greedily constructed trajectory query can be unboundedly suboptimal.
- We find a trajectory query similar to the greedily constructed policy query using a conditional entropy function (Cohn, 2016).

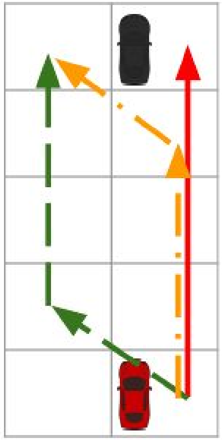
$$H(q'|q) = \sum_{j \in q} P[q \rightsquigarrow j; \psi] H(q'|q \rightsquigarrow j)$$

Rock Collection. The gap between EVOI of trajectory query by projection and competing methods grows when the differences between the optimal values of reward candidates grows (reward setting #3).

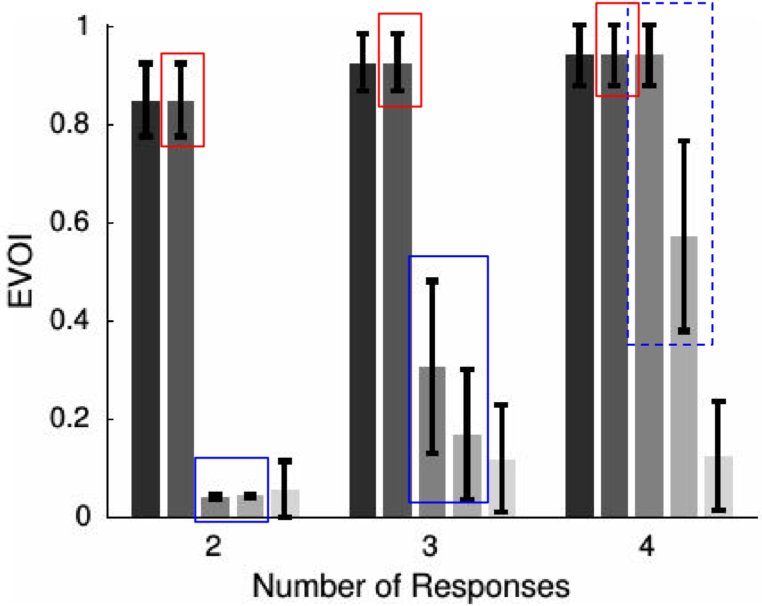


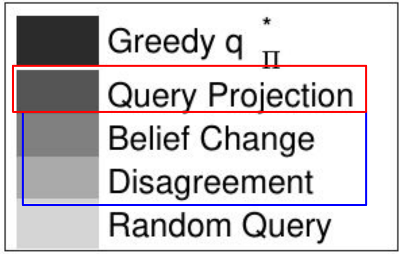


Discrete Driving Domain. The gap between EVOI of trajectory query by projection and competing methods is large when the number of response is small



An example of 3-response trajectory query.





Summary

- Contributions:
  - Greedily construction method to find an approximately optimal policy query.
  - Finding trajectory queries by projecting the greedily-constructed policy query.
- Not covered in this talk but can be found in our paper:
  - A necessary condition of the optimal policy query.
  - An algorithm to find a trajectory query similar to the greedily constructed policy query.
  - Evaluation under different numbers of reward candidates and different numbers of responses.
- Future work:
  - More efficient greedy construction method assuming linearly decomposable reward functions.
  - Querying when the human’s reward function is not fully specified.