Multiagent Reinforcement Learning
Michael Krumdick, Geoffrey Liu

1 Main Findings So Far

1. AlphaRank is ***slow.***
2. Noisy evaluation matrices degrade performance, but still work.
3. Good meta-solvers induce distributions that are very different from the baseline strategies (Uniform, biased Uniform, etc).

2 Background and Related Works

2.1 MDPs

A **Markov decision process (MDP)** [1] is a common formulation of decision making problem. Each MDP can be defined by a four-item tuple. $< S, A, P, R >$

1. S: The space of valid states.
2. A: The space of valid actions.
3. $P_a(s, s')$: A transition function.
4. $R_a(s, s')$: A reward function.

The solution to the MDP will come in the form of a policy π which represents a mapping $S \to \Delta(A_s)$, or from the state space to a distribution over action space. The policy deemed optimal will obtain the highest expected reward, with some discount.

$$\pi^* = \arg\max_{\pi \in \Pi} \sum_{s \in S} d^\pi(s) \mathbb{E}_{a \sim \pi(s)} r(s, a) \tag{1}$$

Reinforcement learning is one method of finding these policies that has seen success. This family of methods has seen increased attention Algorithms such as ...

A **normal form game** can be defined (N, A, u)

1. N, the number of players
2. A, the number of players
3. u, the number of players

Algorithm 1 Double Oracle

```
actions ← RandomAction()
$M$ ← ComputePayoff(actions)
for i = 1:N do
    $\pi_i$ ← MetaSolver($M$)
    newAction ← ResponseOracle($\pi_i$, actions)
    $M_i$ ← UpdatePayoff(actions, newAction)        ▷ Fill in the new rows and columns for the new agent
    actions ← append(actions, newAction)
end for
```

Every time you add a new agent, you need to fill in $2n + 1$ cells.

Kuhn Poker is a simplified version of poker. The deck consists of three cards, the King, Queen and Jack,

Algorithm 2 PSRO

```
agents ← RandomAgents()
$M$ ← ComputePayoff(agents)
for i = 1:N do
    $\pi_i$ ← MetaSolver($M$)
    newAgent ← ResponseOracle($\pi_i$, agents)
    $M_i$ ← UpdatePayoff(agents, newAgent)        ▷ Fill in the new rows and columns for the new agent
    agents ← append(agents, newAgent)
end for
```

Name	Time Complexity	Convergence
Uniform	O(1)	-
Regret Minimization	O(n)	-
Nash	O(n^p)	-
AlphaRank	O(n^p)	-

Table 1: Meta Solvers for Zero Sum Games

References

[1] Martin L. Puterman. Markov decision processes: Discrete stochastic dynamic programming. In *Wiley Series in Probability and Statistics*, 1994.

[2] Eilon Solan and Nicolas Vieille. Stochastic games. *Proceedings of the National Academy of Sciences*, 112(45):13743–13746, 2015.