DeepMind

Multiagent Evaluation under Incomplete Information

Mark Rowland*, Shayegan Omidshafiei*, Karl Tuyls, Julien Pérolat, Michal Valko, Georgios Piliouras†, Rémi Munos

*Equal contributors †Singapore University of Technology and Design

Summary

Problem of interest: multiagent evaluation in K-player, general sum games, where exact payoffs are unknown, using the recently introduced evaluation method α-Rank.

Key application domain: multiagent training scenarios, where payoffs are estimated via repeated simulations.

Main contributions:
- Algorithm for adaptively sampling strategy profiles to be simulated to guarantee correct ranking with high probability.
- A method for propagating uncertainty in payoffs through to uncertainty in the α-Rank outputs.

Background

Game between K players, each with finite strategy sets $S^1, \dots, S^K$

Payoff to player k against the profile $s \in S^1 \times \cdots \times S^K$ is $\mathbf{M}^k(s)$

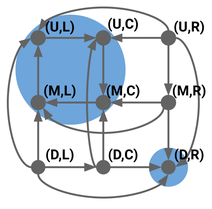
When $\mathbf{M}$ is known, many methods apply for ranking agents, including Elo ratings, Nash equilibria, and α-Rank. These rankings can be used for pure evaluation, or as a component in a training pipeline.

α-Rank computes a distribution over strategy profiles, motivated by evolutionary game theory.

Payoff table $\mathbf{M}$

	L	C	R
U	2, 1	1, 2	0, 0
M	1, 2	2, 1	1, 0
D	0, 0	0, 1	2, 2

Response graph



α-Rank defines an irreducible Markov chain $(X_t)_{t\geq 0}$ over the response graph. If s' is obtained from s by a deviation of player k, then:

$$\mathbb{P}(X_{t+1} = s' | X_t = s) \propto \frac{1 - \exp(-\alpha(\mathbf{M}^k(s') - \mathbf{M}^k(s)))}{1 - \exp(-\alpha m(\mathbf{M}^k(s') - \mathbf{M}^k(s)))}$$

[Exact form motivated by evolutionary game theory]

α-Rank outputs the stationary distribution for this Markov chain.

Motivation: In many practical scenarios, we can only *estimate* $\mathbf{M}$ by simulating games between particular strategy profiles.

Questions:
- How can we efficiently select strategy profiles to simulate that will be most informative in determining the α-Rank output?
- If there is remaining uncertainty about the elements of $\mathbf{M}$, can we efficiently propagate this uncertainty through into the α-Rank output itself?

Adaptive sampling

Key question: If we want to obtain exact α-Rank output with high confidence, which profiles should we simulate, and how many times?

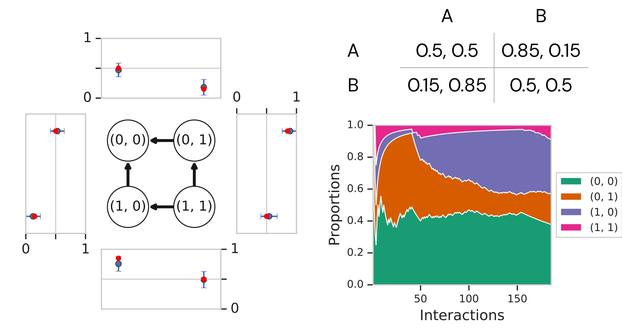
ResponseGraphUCB is an algorithmic framework for solving this problem. It requires a method $\mathcal{S}$ for iteratively sampling profiles to be simulated, and a method $\mathcal{C}$ for computing confidence bounds on empirical payoffs.

See paper for sample complexity bounds, and proofs of correctness.

Algorithm 1 ResponseGraphUCB$(\delta, \mathcal{S}, \mathcal{C}(\delta))$

1: Construct list L of pairs of strategy profiles to compare
2: Initialize tables $\hat{\mathbf{M}}, \mathbf{N}$ to store empirical means and interaction counts
3: **while** L is not empty **do**
4: Select profile s appearing in an edge in L using sampling scheme $\mathcal{S}$
5: Simulate one interaction for s, update $\hat{\mathbf{M}}(s), \mathbf{N}(s)$
6: Check for resolved edges with $\mathcal{C}(\delta)$, remove them from L if so
7: **return** empirical table $\hat{\mathbf{M}}$

Example: ResponseGraphUCB applied to a 2x2 normal-form game.

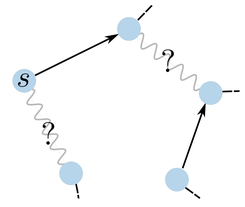


	A	B
A	0.5, 0.5	0.85, 0.15
B	0.15, 0.85	0.5, 0.5

Uncertainty propagation

Key question: Given estimated payoff table $\hat{\mathbf{M}}$ and lower/upper confidence bounds $\mathbf{L}$ and $\mathbf{U}$, what is the minimum/maximum plausible α-Rank weight for a particular strategy profile?

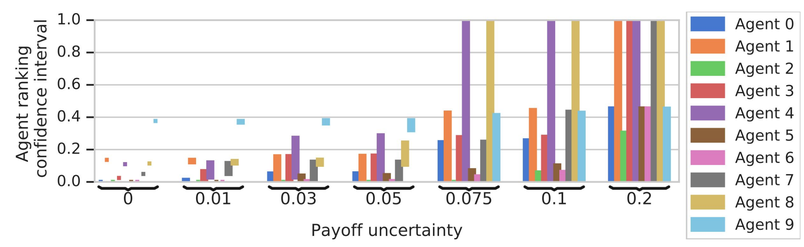
Step 1: Translate into an edge direction selection problem.



Step 2: Translate this into a constrained stochastic shortest path problem, as in the PageRank literature.

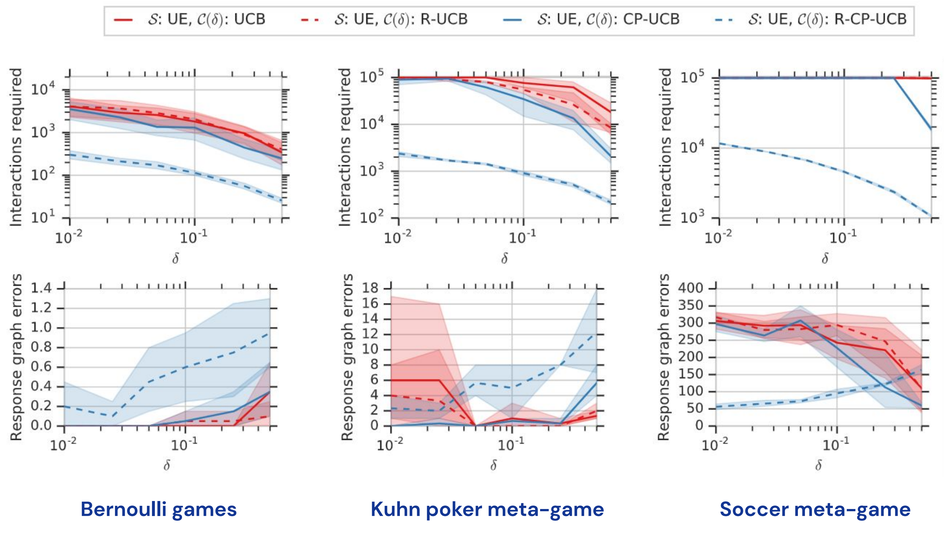
Step 3: We show that it is valid to solve the *unconstrained* SSP problem, which can then be solved using standard techniques such linear programming or value iteration.

Example: α-Rank weight uncertainties for agents in a soccer meta-game.



Experiments

Evaluation of ResponseGraphUCB variants demonstrates that, intuitively, lower error tolerances δ imply more interactions required and fewer response graph errors.



Agent ranking error similarly decreases with the # of ResponseGraphUCB samples (i.e., simulations).