Short Term Opponent Exploitation in Poker

Wesley Tansey
University of Texas at Austin
1 Inner Campus Drive
Austin, TX 78712 USA
tansey@cs.utexas.edu

ABSTRACT

Effectively exploiting opponents in incomplete information, extensive-form games is a difficult task for an online learning agent. Previous work has focused on either maintaining an explicit model that is updated directly based on observed opponent actions or implicit modeling via a portfolio of methods.

This paper introduces four new approaches to playing exploitive poker. First, a one-step temporal differencing version of counterfactual regret minimization called TD(0)-CFR is presented. Second, an alternative implicit modeling approach based on the notion of subpolicies is explored. Third, a hybrid implicit and explicit algorithm is described that leverages a portfolio of models to bootstrap an explicit model. Finally, a combination of the second and third models is discussed along with an approach automtically deriving subpolicies from a portfolio of complete policies.

The game of Leduc poker is used as a benchmark domain to compare the two standard and four novel exploitive approaches. Results from these experiments indicate that everything failed and I should just give up on life.

General Terms

Algorithms, Experimentation

Keywords

poker, opponent modeling, online learning

1. INTRODUCTION

2. BACKGROUND

2.1 Counterfactual Regret Minimization

2.2 OS-MCCFR

2.3 Explicit opponent modeling

2.4 Implicit modeling

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013), Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.*

3. ALGORITHMS

3.1 TD(0)-CFR

3.2 Implicit Subpolicy Modeling

3.3 Policy Bootstrapping

3.4 Subpolicy Bootstrapping

4. EXPERIMENTS

Experiments were performed on all four of our proposed algorithms and the standard approaches from the literature. We next detail our experimental setup and the results for each model in each experiment.

4.1 Setup

To analyze the performance of each model, we conducted a series of experiments in the game of Leduc poker. Leduc is a two-player poker variant where the deck contains six cards (two jacks, two queens, and two kings), each player is dealt a single hole card, there two rounds of betting, and a single community card is dealt after the first round. In total, Leduc contains 144 information sets for the second player, making this game non-trivial for agent modeling while still being tractable to analye in-depth. For more information on Leduc poker, see [1].

Experiments were conducted against three types of stationary opponents. First, a population of 100 "simple" agents was generated by generating two random percentage triplets for each agent; the first triplet was then used to skew the preflop actions of a Nash equilibrium strategy and the second was used to skew the flop actions. A population of 100 "complex" agents was then generated in a similar manner, where with equal probability the agent's preflop or flop actions were skewed, with a similar approach used to skew actions based on the holecard of the opponent (J, Q, or K). These two populations of agents aim to capture the systematic bias presumably found in human players that often over- or under- play a given hand or play looser or tighter at different points in the game. Finally, a Nash equilibrium strategy was used as a baseline to examine how the agents perform when trying to model an optimal player.

In each experiment, the algorithms were tested for 200 matches of 200 hands each, with each algorithm playing against each simple and complex opponent twice. The TD-CFR agent used $\epsilon = 0.1$ with an exponential ϵ-decay factor of 0.99 and a learning rate of $\alpha = 0.05$. For portfolio-

based methods, the portfolio consisted of the Nash equilibrium strategy and four skewed strategies, chosen at random from the population at the start of each match. For explicit models, two times the Nash strategy was used as an initial prior. The subpolicy discovery algorithm used $\delta_{min} = 0.05$, $\mathcal{D} = 0.1$, and a minimum subpolicy size of 3.

4.2 Results

5. DISCUSSION AND FUTURE WORK

6. CONCLUSIONS

7. REFERENCES

[1] F. Southey, M. Bowling, B. Larson, C. Piccione, N. Burch, D. Billings, and D. C. Rayner. Bayes' bluff: Opponent modelling in poker. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI'05)*, pages 550–558, 2005.