AlphaOne: Self-play Reinforcement Learning on Imperfect Information Games

IN2349 ADL4R - Final Report

Tobias Kirschstein
tobias.kirschstein@tum.de

Ananta Bhattarai
ananta.bhattarai@tum.de

I. Motivation

Games appear everywhere in politics, diplomacy or business decision making. However, in the vast majority of cases, the agents do not have access to all the information of the game. Instead, they have to decide on actions based on partial knowledge. As such, the power of *AlphaZero*, which has achieved super-human performance in single agent settings and perfect information games, cannot readily be applied to most real-world scenarios. Furthermore, solving such imperfect information games constitutes a much bigger challenge due to the high uncertainty of the true game state. In this work, we propose and evaluate a novel approach *AlphaOne* that extends *AlphaZero* to imperfect information games using an adapted version of Monte-Carlo Tree Search.

II. Methods

A. Imperfect Information MCTS (MCTS-II)

Monte-Carlo Tree Search (MCTS) is the key ingredient behind the success of AlphaZero [2]. To extend MCTS to imperfect information games we realize that guessing a state in the information set, which contains all currently possible game states, can be treated the same as choosing an action at a specific game state in traditional MCTS. From this, we can construct a tree with alternating levels, where one can follow an edge from an observation node to a game node (by guessing the state in the information set), and subsequently follow an edge from the game node to an observation node (by choosing an action). By applying MCTS to this larger tree, we obtain a belief over the game states in the information set as well as a policy over the possible actions. To account for the knowledge difference of the players, MCTS-II furthermore traverses this tree for each player in lock-step by only relying on their respective information set. By doing this, MCTS-II explicitly explores scenarios where the opponent wrongly guesses a game state that the root player knows cannot arise anymore. This kind of reasoning is important to detect the benefit of bluffing in poker for example.

B. AlphaOne

To extend *AlphaZero* to imperfect information games, we introduce an AlphaOne agent that uses two neural networks to guide the MCTS-II search tree. At the observation node, we use an observation model $g_\phi(o_1^i, ..., o_t^i)$ that takes the history of a player's observations (past three observations in our case) to select/guess one of the possible game states $\hat{s}_t = \arg\max_{s_t^* \in \mathcal{S}^i(o_1^i,...,o_t^i)} P'(o_t^i, s_t^*)$. At the game node, we use a game model $f_\theta(\hat{s}_t) = (P(\hat{s}_t, \cdot), V(\hat{s}_t))$ that uses the perfect information of the guessed state $\hat{s}_t$ to select one of the possible actions. Figure 1 shows how the two neural networks and MCTS-II work under the hood in AlphaOne.

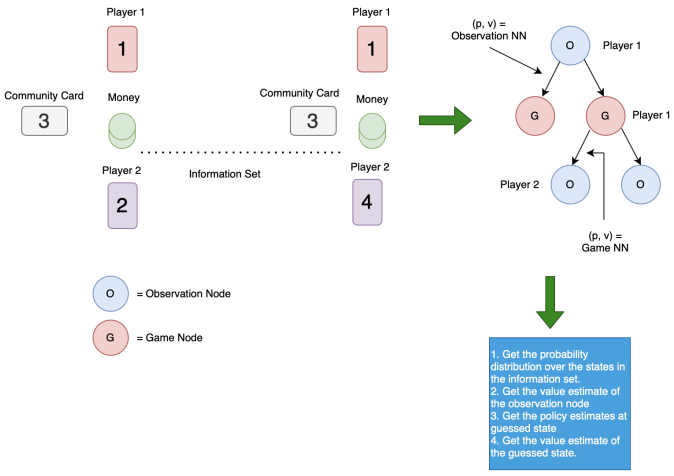


Fig. 1: MCTS-II in AlphaOne. Information set contains two states that the agent might be in for a very simplified game of poker.

We train the AlphaOne agent in a self-play reinforcement learning setting. First, the weights of the neural networks are initialized randomly. In each iteration of our training algorithm, we play a number of games of self-play. At each decision point of the game, we execute an MCTS-II search at the information set. The training data generated using the MCTS-II algorithm is used to update the parameters of both models. At the end of the iteration, the newly trained models and the current best models compete against each other. If the win rate or average reward of the newly trained models are greater than a set threshold (55% win rate or 0.2 average reward in our case), we set them as the current best models.

C. Determinized MCTS (D-MCTS)

An alternate way to tackle imperfect information games is *Determinization* [3]. A *Determinization* is an instance of the equivalent deterministic game of perfect information in

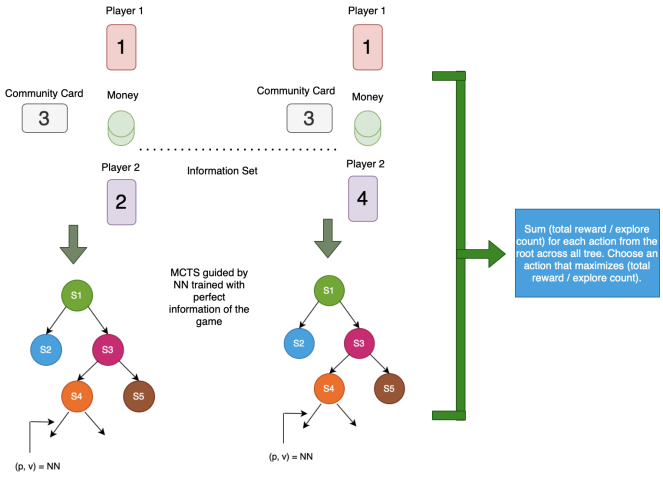


Fig. 2: D-MCTS algorithm. Information set contains two states that the agent might be in for a very simplified game of poker.

which the agent chooses the current state from the information set, and the outcomes of all future chance events are fixed and known. Our D-MCTS agent runs a regular MCTS search guided by the neural network independently for every state in the information set. After that, we sum up the total reward over explore count for each action from the root across all trees. In the end, we choose an action that maximizes the total reward over explore count. Figure 2 shows the process of the D-MCTS algorithm. During training, we train the network in a regular AlphaZero manner with perfect information about the game.

D. Omniscient and Blind Agent

As an alternative to AlphaOne and D-MCTS, we also designed a Blind Agent that uses a neural network to output policy and value estimates directly, given the player's observations. We first train an omniscient (cheating) agent in a regular AlphaZero manner with perfect information about the game. We then fit another model, the blind agent, to directly imitate the predicted policies of the Omniscient MCTS, but given only the observations of the respective player.

E. Counterfactual Regret minimization (CFR)

CFR is a game-theoretic approach widely used to solve imperfect information games [4]. It works by repeatedly playing against itself while minimizing regret. Regret is a numeric value that can be described as "How much better would I have done over all the games so far if I had just always played this one action at this decision, instead of choosing whatever mixture over actions that my strategy said I should use?". After every game, CFR improves its strategy by summing the total amount of regret for each action at each decision point. By repeating this process for a number of games of self-play, the average policies converge towards the Nash Equilibrium. The Nash Equilibrium strategies always play a perfect defense which means it can not do worse than a tie. However, CFR becomes very hard to deal with for larger games as it fills an information set - action table, e.g., in the game of Hold'em poker, naively storing this table would take 262TB of storage [1].

F. Hybrids

We furthermore investigate the following hybrid variants of D-MCTS, AlphaOne and Omniscient models:

Base	Name	Description
D-MCTS	D-MCTS + Omniscient	Use Omniscient Model to guide MCTS
	D-MCTS + AlphaOne	Use AlphaOne's observation model to weight states in the information set
	Super Hybrid	Combine the above two
	D-MCTS rollout	Guide MCTS by using rollouts instead of NN predictions
AlphaOne	AlphaOne + Omniscient	Replace Game Model with Omniscient Model

III. Results

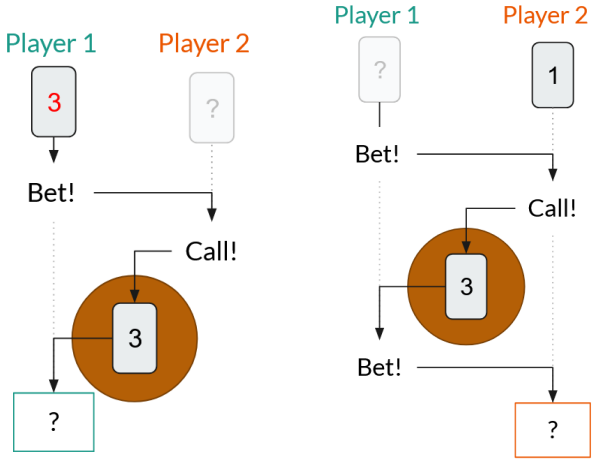
A. Tournament

We compare all the different agents on the game of Leduc Poker which is a simplified Poker variant that only contains 3 pairs of cards and players can choose among 3 actions: (i) Fold, (ii) Check/Call, or (iii) Bet/Raise. Every player can Bet/Raise twice in the game increasing the pot size by 2 (pre-flop) or 4 (post-flop). The maximum pot size, i.e., the maximum reward, is 13. In a big tournament, we let every pair of agents compete against each other for 200 matches, equally distributing who plays first. Every MCTS-based agent is granted 100 simulations to compute its policy. The results of the tournament are depicted in figure 6. The cheating Omniscient Model and a randomly acting agent mark realistic upper and lower bounds for the average reward per game. It can be seen that simply training a blind model to imitate MCTS by directly predicting the policy does not work very well. On the other hand, Counterfactual Regret Minimization performs very well, which is also expected as it computes a near optimal policy table. However, CFR is hard to scale to larger games. Furthermore, all the D-MCTS variants perform almost equally well, but none of the variants and hybrids could improve on raw D-MCTS. Finally, untrained AlphaOne, i.e., raw MCTS-II, performs quite poor which is not surprising as the introduction of alternating observation and game nodes essentially doubles the depth of the tree that has to be traversed via MCTS-II. This means that MCTS-II explores fewer game states within its 100 simulations than D-MCTS. However, when trained, AlphaOne shows a big leap in performance which can be enhanced even more by using the Omniscient Model to evaluate game nodes. However, as

it stands, AlphaOne falls a bit behind CFR and D-MCTS in terms of performance.

B. Specific Game Scenarios

We investigate two specific game scenarios to get a better understanding of the differences between the approaches.
In the first scenario, the flop was just revealed with player 1 hitting top pair. As it can be seen in figure 4a, the value prediction that AlphaOne assigns to this scenario reflects the good position of player 1 quite well. From the predicted policies in figure 5a we can observe that AlphaOne, D-MCTS and CFR identify that it is a good time to bet. However, the omniscient and blind model fail to do so, which is due to the fact that the cheating model does not deal with information sets, but instead directly works on the true game state. In the second scenario, it is player 2's turn to react to player 1's bet, while player 2 has the lowest card. Again, the value prediction of AlphaOne as depicted in figure 4b is quite realistic. However, when looking at the predicted policies in figure 5b it can be observed that all models except for AlphaOne favor folding in this scenario. This shows that AlphaOne computes very optimistic policies which can be attributed to MCTS-II being biased towards guessing states that are beneficial for the player.



(a) Scenario 1. (b) Scenario 2.

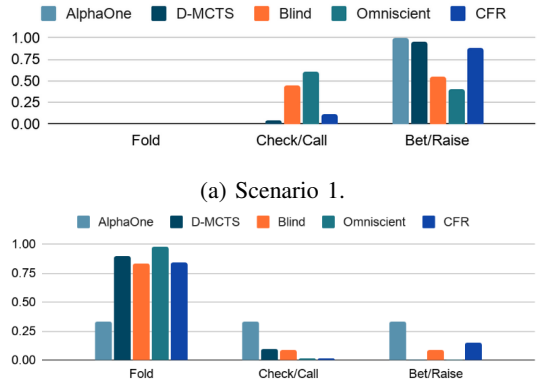
Fig. 3: Specific Game Scenarios in Leduc Poker.

(a) Scenario 1. (b) Scenario 2.

Fig. 4: Value Predictions of AlphaOne for the game scenarios.

IV. Conclusion

We proposed a new MCTS extension called MCTS-II for incomplete information games, showed how it can be trained with neural networks in an AlphaZero manner, and conducted comprehensive comparisons against other methods. The results show that the value predictions of AlphaOne are quite on point, while the predicted policies are too optimistic. In a tournament, AlphaOne falls a bit behind CFR and D-MCTS but shows great improvement through training which indicates the overall potential of the approach. Furthermore, we outlined the shortcomings of an Omniscient/Blind approach which, as opposed to information set based methods such as AlphaOne, D-MCTS and CFR, is unable to detect the benefits of bluffing or low-playing of a card.



(a) Scenario 1.

(b) Scenario 2.

Fig. 5: Computed policies of the different approaches for the game scenarios.

References

[1] Michael Bowling et al. "Heads-up limit hold'em poker is solved". In: *Science* 347.6218 (2015), pp. 145–149.

[2] David Silver et al. "Mastering the game of Go without human knowledge". In: *Nature* 550 (Oct. 2017), pp. 354–359. DOI: 10.1038/nature24270.

[3] D. Whitehouse, E. J. Powley, and P. I. Cowling. "Determinization and information set Monte Carlo Tree Search for the card game Dou Di Zhu". In: *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*. 2011, pp. 87–94. DOI: 10.1109/CIG.2011.6031993.

[4] Martin Zinkevich et al. "Regret minimization in games with incomplete information". In: *Advances in Neural Information Processing Systems* 20 (Jan. 2008), pp. 905–912.

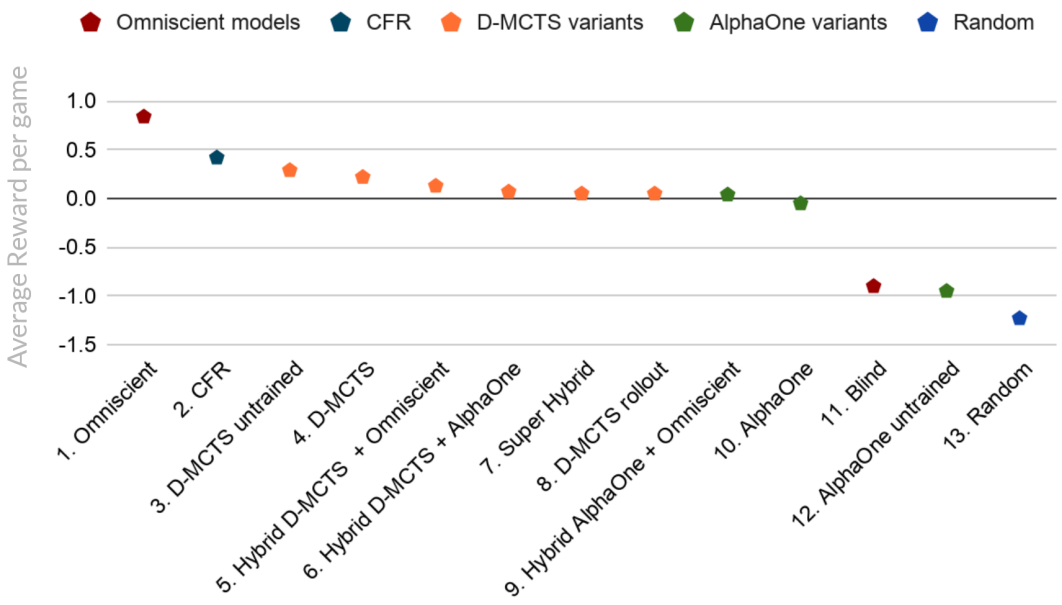


Fig. 6: Tournament Results. Performance is measured in average amount of money won/lost per game.