Introduction to Artificial Intelligence
Final Project Interim Progress Report

鄭伯俞 ID: 0610021

蔡柏毅 ID: 0610018

黃秉茂 ID: 0616098

1 June 2021

1 Introduction

In this project, we aim to create AI agents that play a chosen board game and compare their performances. At the moment, we are considering Ultimate tic-tac-toe [1] and Othello [2], amongst others.

2 Methodology

We plan to implement some methods commonly used for game playing and compare them on the game we ultimately choose. The following methods are currently being considered.

- Minimax (negamax with alpha-beta pruning) with hand-crafted heuristic evaluation function.
- Pure MCTS.
- Neural network-based, where each move is determined by a classifier based on a neural network that learns from the actions of the aforementioned MCTS agent that is allowed to run significantly longer (thus producing higher-quality moves). Note that this is a supervised learning task.
- Method akin to AlphaGo.
- Reinforcement learning methods.

3 Progress

At the moment, we have implemented the first two methods on the game Ultimate tic-tac-toe. We have confirmed that both agents are working by testing them on the CodinGame arena[1]. Currently the MCTS agent is online[2]. At the time of this writing, the agent ranks 321st (out of 3940) on the leaderboard (account name: abt8601). Note that we do not plan to compare the performances of the agents on the CodinGame arena; that would simply require too much

[1] `https://www.codingame.com/multiplayer/bot-programming/tic-tac-toe`

[2] Per account, only one agent can be online at the same time.

accounts. We will just pit them against one another and compare the win rates. Currently, we have found that an earlier iteration of the MCTS agent was able to win all of 10 games against the minimax agent. The current MCTS agent is even stronger. However, to really compare the two methods, we will need to implement some of the optimisations introduced since the adoption of MCTS (like bitboard and some crazy memoisation) on the minimax agent.

4 Remaining milestones

We will proceed in two directions. The first one is to further optimise the current two agents to increase their performances. The second one is to implement (some of) the aforementioned methods. However, there are doubts that the AlphaGo method, amongst other proposed ones, may be an overkill for Ultimate tic-tac-toe. After all, AlphaGo was created to play Go on a 19×19 board.

If the performance improvement brought by using these sophisticated methods turn out insignificant and we still have time, then we may climb the complexity ladder to try more sophisticated games.

References

[1] Wikipedia. Ultimate tic-tac-toe. Retrieved 1 June 2021 from `https://en.wikipedia.org/wiki/Ultimate_tic-tac-toe`.

[2] Wikipedia. Reversi. Retrieved 1 June 2021 from `https://en.wikipedia.org/wiki/Reversi`.