Oliver Willis & Hasan Polat

CS 730/830, Spring 2023

Preliminary Proposal

Problem:

The problem we intend to investigate revolves around an artificial intelligence (AI) implementation of a 2 player Chinese Checkers game. Within this implementation, we intend for our AI to compete on an equal skill basis to a human during a game. For this project, we want to implement a reinforcement leaning agent in pytorch. We do not have much experience with the pytorch library, and so implementing this model will bring a fun challenge to our solution.

Points of Interest:

The game of Chinese checkers is interesting because it presents a challenging game with very simple rules that require the players to think strategically and make complex decisions in order to win. Compared to the conventional checkers or chess game that many know, Chinese checkers is simple enough for a human to strategize at the least, a few steps ahead, or significantly more. In terms of our solution, this poses an interesting challenge. Chess has been studied for years in AI research to determine what states are considered 'good states.' However, Chinese checkers is not as popular and has not been studied as intensely as chess, thus the static evaluations for this game are not as sophisticated. A common heuristic for Chinese checkers is the single-agent distance to the goal, but it yields a relatively weak AI.

Challenge:

- State space for this problem can be huge depending on player number and board size. For our board, it will be greater than 10^{12} positions.
- We do not have a good heuristic.

Recent advancements in reinforcement learning have shown that AI can learn to play a game at superhuman levels without a static evaluator. AlphaZero uses the Monte Carlo tree search combined with deep learning to evaluate game positions. Importantly, AlphaZero does not require training data and learns through playing simulations. In this way, AlphaZero can be pinned against itself so that it can learn how to play well over time.

How to Solve:

The first step to our solution is to create a game board simulator for Chinese checkers. With this board simulator, we will be able to generate the next possible states for a search algorithm. The board simulator will also act as a trainer for RL. We have two algorithmic approaches to solving the problem. The first is to naively apply the minmax search using the single agent to goal distance heuristic. Given the complexity of this game, we do not expect the AI to have a strong performance. The second approach involves the use of AlphaZero through its self-training in order to produce a strong AI. We will use the min max as our based implementation and if time allows, we will fine tune the RL approach.

References:

http://cs229.stanford.edu/proj2016spr/report/003.pdf

https://webdocs.cs.ualberta.ca/~nathanst/papers/sturtevant2019chinesecheckers.pdf

https://webdocs.cs.ualberta.ca/~nathanst/papers/comparison_algorithms.pdf

https://webdocs.cs.ualberta.ca/~nathanst/papers/sturtevant2019chinesecheckers.pdf

Appendix:

```
     1
    1 1
   1 1 1
  - - - -
 - - - - -
- - - - - -
 - - - - -
  - - - -
   2 2 2
    2 2
     2
```

1s refers to player 1's stones. 2s refers to player 2's stones. '-' refers to vacant spaces that players can move. The State Space of this game board is 3 ^ 36.