Exercise 4: Writing a MCTS-based connect four player

February 12, 2024

The goal of this assignment is to write a program that plays connect four well by using the MCTS algorithm learned in class.

In the connect four game, players take turns putting black and white pieces in a rectangular "grid". The first player to get four pieces in a row, either vertically, horizontally, or diagonally, wins. The game grid is placed standing up, and each player chooses a column into which their piece will drop. The piece occupies the lowest available space within the chosen column.

Assignment Steps

1. I am attaching the file ConnectFour.py to get you started. The ConnectFour class represents the Connect Four game and its rules. I have implemented the functions that I think you will need for your MCTS player. Download the file, read the code and make sure that you understand it. I encourage you to run the code and change it as you see fit.

2. Write a class called MCTSNode that represents a single node in the MCTS tree. It should have the data members 'wins' and 'visits', as well as any other attributes that you see fit to add.

3. Write a class called MCTSPlayer. In this class, you should implement the MCTS algorithm to write a function called `choose_move`, that receives a ConnectFour object and the number of iterations (as well as any other parameters that you see fit to add), and uses MCTS to select the best move.

4. As design choices, you should think about the following:
    - How many iterations strikes the best balance between strength and speed?
    - For the selection state, will you use UCT or epsilon-greedy? How should you balance exploration and exploitation?
    - Will your nodes hold cloned game states? Or will you have a single game state, and use move and unmove to go up and down the tree?
    - Will you have completely random simulations? Or will you apply a heuristic? For example, you could say that if there is a move leading to a win then you play that move.
    - After searching the tree will you choose the move with the largest visit count, or the one with the highest win rate?
5. Pay attention to the following tricky issues:
    - Make sure that your search counts wins, losses and draws correctly.
    - If you reach a terminal game state during the selection phase, then you don't expand or simulate - you go directly to the propagation step.
6. Make sure to test the different functions in the MCTS thoroughly to check for bugs. You should also play against it yourself and have it play itself to see if you can identify gaps or weaknesses in its play.