Noah Miles

CS4200-01

Project 4 Report

05/03/20

Approach:

The evaluation function I decided to use was one that took the computer’s position on the board and counted the number of legal moves from that one position. The alpha-beta pruning algorithm then applies the evaluation function on every potential move that can be generated from the current state to determine which move is the best move. The higher number of legal moves, the more desirable the move is.

Problems:

A problem I encountered was using my evaluation function to help determine if a move is valid. I solved this problem by adding a check for the direction of the move and passing that and the distance between the source and destination to the evaluation function. Inside my evaluation function, I added a flag that was toggled if the input contained more values than just the source and destination coordinates. If there was, it would check up to the total distance away from the source in the direction of the direction and return -1 if it was an invalid move. Another problem I encountered was generating the successor states of the current position and handling setting and unsetting the board from the successor states. I ended up creating a setMove method that took the source, the destination, and a boolean flag that would determine if it was to set or unset the action. The last major problem I encountered was keeping track of the action that returned the value in the minimax algorithm. I solved this problem by implementing my own tuple class since java does not have a built-in tuple.