Felipe Gutierrez
CS540

HOMEWORK 2 - Checkers Report

1. For my competitive player I implemented a new SBE which I called CompetitiveEvaluator.java. I started this SBE from the fact that having more pieces in the board was the most important feature, so the feature that weights the most is the number of pieces the player has against the number of pieces the opponent has (just like the simple evaluator function that was given to us). On top of this I added 4 more features that have less weight than the number of pieces:

a. Number of defense pieces: These are the number of pieces that are in the first two rows of each player side.

b. Number of safe pieces: These are the pieces that are on the edge of the board. Since they are on the edge of the board the opponent can't take them away.

c. Number of safe attacking pawns: This feature counts the pawns that have an opponent pawn right in front of them and have blank squares on both of the diagonals (if red, the upper diagonals, and if black the lower diagonals). This is a safe attacking pawn because when the opponent moves their pawn to any of the diagonals the player will be able to take that piece. This was the only feature I came up with, I feel that a couple more cases should be taken care of to optimize this feature but that was what I implemented.

d. Number of triangle formation: Counts the number of pieces that are in triangle formation. This means that one piece is in front and has its 2 back diagonals covered.

Most of my features could be improved if I also take into account when we get to the other side of the board. They seem to be effective at the beginning of the game, but as the game goes on the features become less effective (I think).

2. I implemented the killer move heuristic to my alpha beta player. I have a 2D array with a fixed length and height, that contains moves. If a move at a certain depth is good enough to cause pruning I push that move into the killer move 2D array, the row is determined by the current depth. I keep track of 5 killer moves per row, and after it is filled the killer move in the first spot is replaced by a new one and the last killer move is discarted. When I come back to that depth in future deepening I am going to look if the killer moves are in the list of possible moves and put them at the beginning. By doing this we could expect more pruning.

Before trying the kille move heuristic, I was trying to implement the negascout algorithm for my competitive player. First I succesfully implemented the negamax algorithm which performs around the same as my alpha beta player. After this I started building upon it to make the negascout player. I wasn't able to succesfully implement the negascout player and I was running out of time to submit the program, so I change my approach and implemented the killer heuristic. In the files I submitted under the package Felipe where my competitive player is, there are 2 more files called NegaMaxFelipePlayer.java and NegaScoutFelipePlayer.java which contain my effort to implement these algorithms.

3. Something through the whole implementation of the competitive player, was being able to create a good enough SBE that would make it beat the alpha beta player. Unfortunately I wasn't able to get to that point. My competitive player seems to perform a little bit better than the alpha beta player but it still ends up in a tie because towards the end of the game, both players end up repeating the same move over and over again to keep their kings safe.