ECE 457A - Question 2

1)

The program implements a simulation of two players playing the conga board game. There are no parameters for the user to input when the user runs the program. The black player is implemented with MinMax algorithm, and optimized with Beta-Alpha Pruning technique. The white player is implemented with a random agent. The game starts with the black player moving, and then following by the white player. The game will only end when one player cannot make any more moves.

For each move the players make, the program will display the details about the move, and what the current board look it. At the end of the program, the total number of moves bade by both player will be displayed.

2)

The search agent is implemented with MinMax strategy and Beta-Alpha pruning. The search agent will evaluate its possible moves by expanding the tree of the next moves. If the moves are in leaf nodes or when the max depth of the tree is reached, the search agent will evaluate the position of the board using the evaluation function.

3)

The finalized evaluation function looks at the list of possible blocks adjacent to the black player pieces and the white player pieces that are not blocked by the opponent. Then the value returned by the evaluation function equals the black player’s available blocks subtract by the white player’s available blocks.

|  |  |  |  |
| --- | --- | --- | --- |
| **Depth** | **Move Count** | **Average** | **Speed** |
| 3 | 24 | 24 | Instant |
| 3 | 28 | 26 | Instant |
| 3 | 158 | 70 | Instant |
| 3 | INF | NA | NA |
| 3 | 26 | 59 | Instant |
| 4 | 18 | 18 | Very Fast |
| 4 | 24 | 21 | Very Fast |
| 4 | 62 | 34.67 | Very Fast |
| 4 | INF | NA | NA |
| 4 | 14 | 29.5 | Very Fast |
| 5 | INF | NA | NA |
| 5 | 48 | 48 | Slow |
| 5 | INF | NA | NA |
| 5 | INF | NA | NA |
| 5 | INF | NA | NA |

With a tree depth of 4, the finalized evaluation function can win with an average of 29.5 moves, and tie (goes to infinity) with a chance of 20%. The speed of the calculation is also relatively fast compare to other depths. The time complexity is O(8 ^4), since the depth is 4, and max possible moves for each node is 8. The space complexity is O(8\*4).

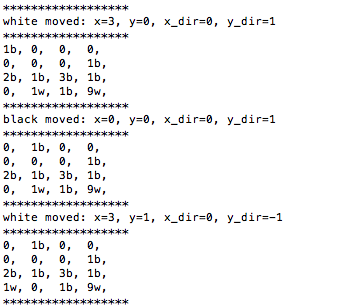
The other evaluation functions I tried usually ended up being much worse than the finalized evaluation function.

*Second Method:*

First of all, I tried using a more forceful approach based on the idea that offensive is better than defensive. Thus, blocking the white piece from moving anywhere had a higher priority. However, this resulted in the game ending with an average of 2000 moves, with a depth of 3.

*Third Method:*

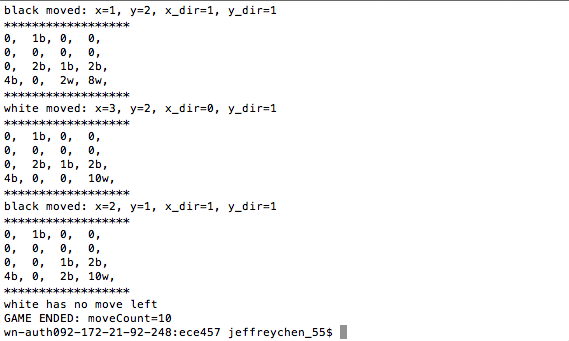
Secondly, I tried improving the finalized evaluation function. The reason the finalized evaluation function sometimes goes into at infinite loop is because of the black player trying to minimize being block by the opponent. Sometimes, toward the end of the game, the black player will keep on moving one free on back and forth. The white piece will only have on place to go. Thus, the game never ends.



To improve this issue, I tried adding a distance algorithm to minimize the distance from a empty white adjacent block to the black pieces, so the black player will move its piece closer to the white. However, it ended with the game not ending most of the time.

4)

The image blow shows a sample of what it looks like when the game ends.



The black player will always try to minimize the possibility of becoming surrounded by the opponent, and will try to make sure the white player cannot move anywhere by moving black pieces to the side of white pieces. For example, one of the key move that the black player made forced the white piece to a corner, which resulted in a quick game.

