CS 420 (Artificial Intelligence)

Project #4

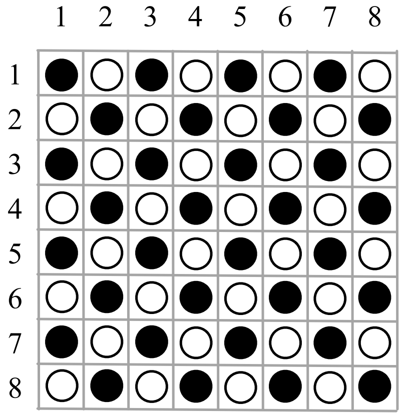
Due by 11:59pm on Sunday, 10/25/2020

Objective

- Implement a program (in your favorite programming language) to play the game Konane.

Introduction

The game Konane is also called Hawaiian Checkers. It is typically played on an 8 by 8 board of light and dark pieces as shown below:



The rules of the game are as follows[i]:

- The game begins with all the pieces on the board. Players decide which colors to play (black or white).
- Black starts first, and must remove one of his/her pieces from one of the four positions: <8, 8>, <1, 1>, <5, 5>, <4, 4>(row-col).
- White then removes one of his/her pieces either horizontally or vertically adjacent to the empty space created by Black. There are now two adjacent empty spaces on the board.
- From here on, players take turns capturing each other's pieces. All moves must be capturing moves. A player captures an opponent piece by jumping one of his/her own pieces over a horizontally or vertically adjacent opponent piece, and landing on a vacant space immediately beyond.
- A player's piece can continue to jump multiple times over enemy pieces, but only in the same horizontal or vertical direction, if the player thinks that this is to his/her advantage. But the player must at least capture one opponent piece in a turn.
- The player unable to make a move is the loser; his opponent is the winner. Note that it is impossible to draw in Konane, because one player eventually cannot perform a move.

Your program should use the MiniMax algorithm with Alpha-Beta pruning. For each game, your program should also provide the following information:

1. The number of times a static evaluation was done.
2. The average branching factor.
3. The number of cutoffs that took place.

You must use the same coordinate system as shown for specifying moves. A different coordinate system will result in grade penalty.

Play the game by varying the depth of search (from 1 to 6) or ply of 1 to 3 (a ply is two levels: Max+Min), and plot the above results. First implement just the plain MiniMax algorithm, record the data and plot it. Next, do the same with Alpha-Beta pruning and record the results.

If you feel inclined to do so, you may use a graphical front end, rather than the text version shown above. However, make sure that all your efforts are initially focused on completing the program as described. If time permits, you may work on the graphics part.

What to produce:

Your submission should include the following:

1. (5pt) Your source code (properly commented) and a readme file with
   a. instructions of running your code,
   b. screen shots of running your program.
2. (15pts) A report that includes:
   a. (6pts) The pseudo code of your game playing algorithm(s). You must note clearly any variations from the standard algorithm.
   b. (3pts) An explanation of your static evaluation function.
   c. (6pts) A comparison of the the data obtained from Minimax against Alpha-Beta and discuss what efficiencies were gained by incorporating pruning.

[i] https://en.wikipedia.org/wiki/Konane