2/28/2021

Done by Ahmadjon Rajabov

Student id:

Assignment 3

Reversi Game

**Introduction**

This work includes development of Reversi game with alpha-beta cut approach in Artificial Intelligence. Work done in Python programming language using several libraries such as collections, sys, etc. In this report given description of program and instruction of using.

**Reversi Game**

Reversi is a strategy board game for two players, played on an 8×8 uncheckered board. It was invented in 1883. Othello, a variant with a change to the board's initial setup, was patented in 1971.

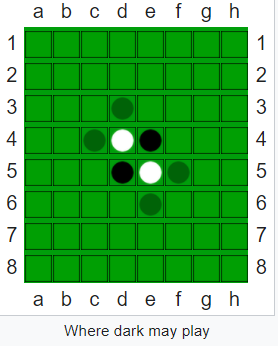
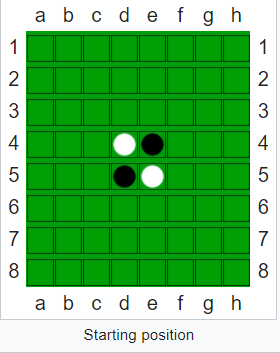
There are sixty-four identical game pieces called disks (often spelled "discs"), which are light on one side and dark on the other. Players take turns placing disks on the board with their assigned colour facing up. During a play, any disks of the opponent's colour that are in a straight line and bounded by the disk just placed and another disk of the current player's colour are turned over to the current player's colour. The objective of the game is to have the majority of disks turned to display your colour when the last playable empty square is filled.

*Rules*

Each of the disks' two sides corresponds to one player; they are referred to here as light and dark after the sides of Othello pieces, but any counters with distinctive faces are suitable. The game may for example be played with a chessboard and Scrabble pieces, with one player letters and the other backs.

The historical version of Reversi starts with an empty board, and the first two moves made by each player are in the four central squares of the board. The players place their disks alternately with their colours facing up and no captures are made. A player may choose to not play both pieces on the same diagonal, different from the standard Othello opening. It is also possible to play variants of Reversi and Othello where the second player's second move may or must flip one of the opposite-coloured disks (as variants closest to the normal games).

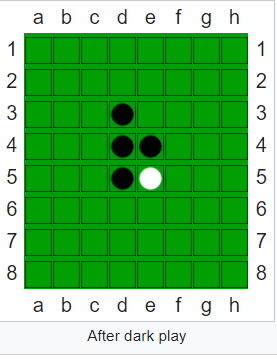
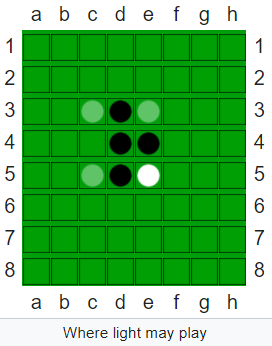
For the specific game of Othello, the game begins with four disks placed in a square in the middle of the grid, two facing white-side-up, two dark-side-up, so that the same-coloured disks are on a diagonal. Convention has this such that the dark-side-up disks are to the north-east and south-west (from both players' perspectives), though this is only marginally consequential: where sequential openings' memorization is preferred, such players benefit from this. The dark player moves first.



Dark must place a piece (dark-side-up) on the board and so that there exists at least one straight (horizontal, vertical, or diagonal) occupied line between the new piece and another dark piece, with one or more contiguous light pieces between them. For move one, dark has four options shown by translucently drawn pieces.

Play always alternates. After placing a dark disk, dark turns over (flips to dark, captures) the single disk (or chain of light disks) on the line between the new piece and an anchoring dark piece. No player can look back to the previous status of disks when playing moves. A valid move is one where at least one piece is reversed (flipped over).

If dark decided to put a piece in the topmost location (all choices are strategically equivalent at this time), one piece gets turned over, so that the board appears.

Now light plays. This player operates under the same rules, with the roles reversed: light lays down a light piece, causing a dark piece to flip. Possibilities at this time appear thus (indicated by transparent pieces).

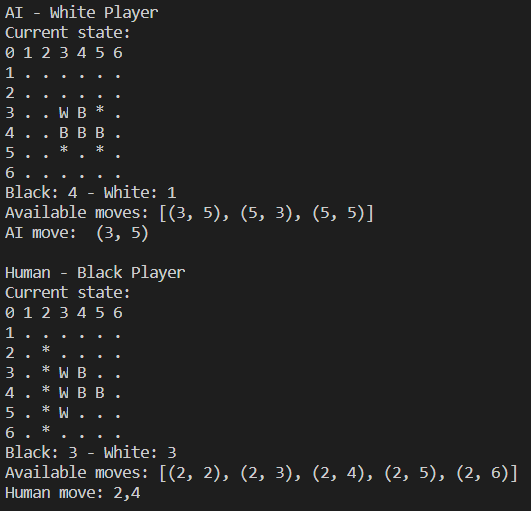
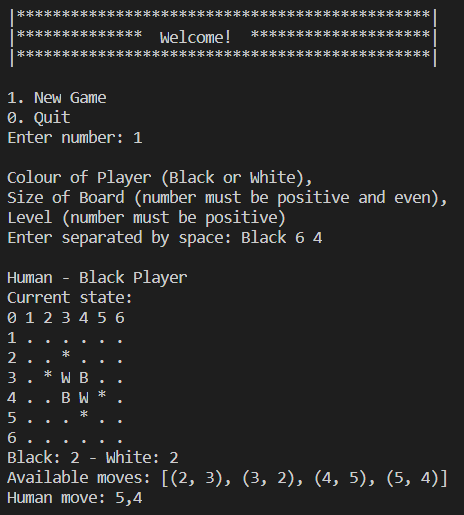
Players take alternate turns. If one player cannot make a valid move, play passes back to the other player. When neither player can move, the game ends. This occurs when the grid has filled up or when neither player can legally place a piece in any of the remaining squares. This means the game may end before the grid is completely filled. This possibility may occur because one player has no pieces remaining on the board in that player's colour. In over-the-board play this is generally scored as if the board were full (64–0).

The player with the most pieces on the board at the end of the game wins. An exception to this is that if a clock is employed then if one player defaults on time that player's opponent wins regardless of the board configuration, with varying methods to determine the official score where one is required.

**My Reversi Game**

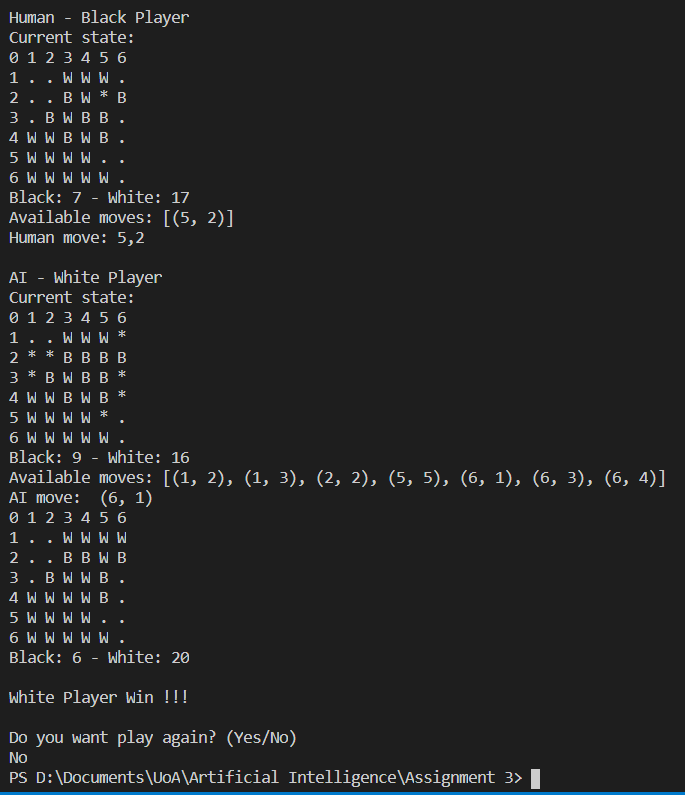
Each reversi piece has a black side and a white side. On your turn, you place one piece on the board with your colour facing up. You must place the piece so that an opponent's piece, or a row of opponent's pieces, is flanked by your pieces. All of the opponent's pieces between your pieces are then turned over to become your colour.

The object of the game is to own more pieces than your opponent when the game is over. The game is over when neither player has a move. Usually, this means the board is full.



The game is started in the position shown below on a reversi board, size of board was input at the beginning. Also the choice of colour and difficultic level in order.

This program also provides avaliable moves to human and AI player. A move consists of placing one piece on an empty square.You can capture vertical, horizontal, and diagonal rows of pieces. Also, you can capture more than one row at once. The game ends when one player wins, by making his color dominant on the board.



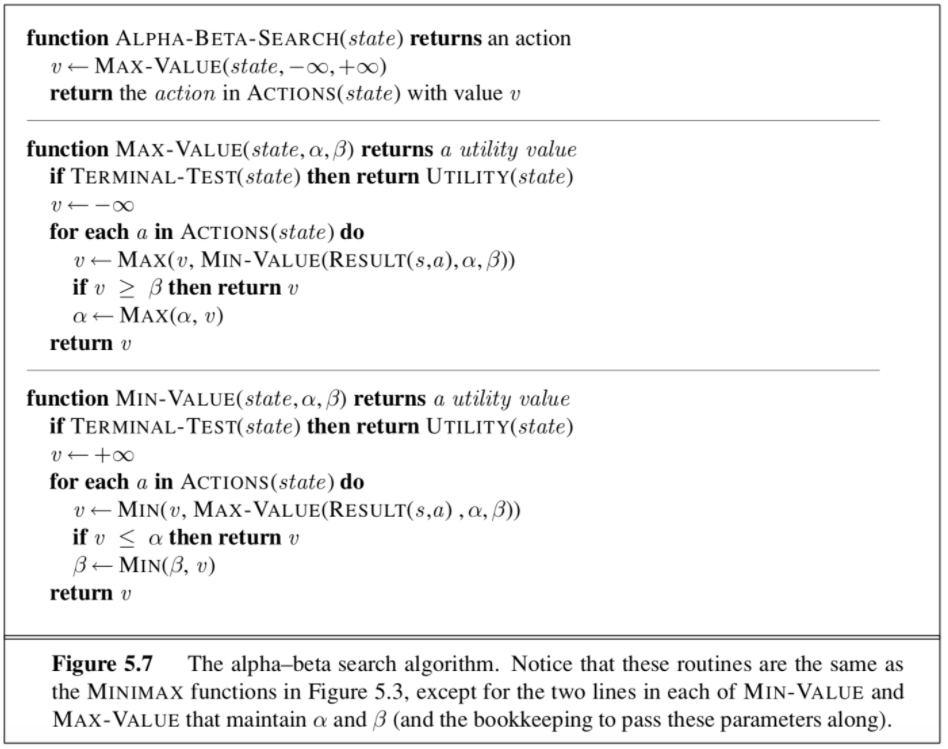
At the end of game appear score and question “Do you want play again?” by answering “No” we quit from game, “Yes” restarts program from the beginning.

**Search Engine**

Minimax procedure with alpha-beta pruning.

**Alpha-beta Pruning**

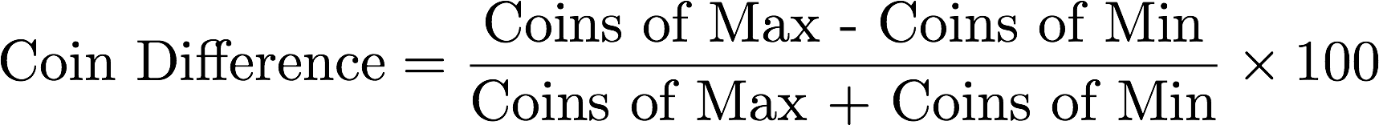
Alpha–beta pruning is a search algorithm that seeks to decrease the number of nodes that are evaluated by the minimax algorithm in its search tree. It is an adversarial search algorithm used commonly for machine playing of two-player games (Tic-tac-toe, Chess, Go, etc.). It stops evaluating a move when at least one possibility has been found that proves the move to be worse than a previously examined move. Such moves need not be evaluated further. When applied to a standard minimax tree, it returns the same move as minimax would, but prunes away branches that cannot possibly influence the final decision.



**Heuristics Applied**

Coin Difference

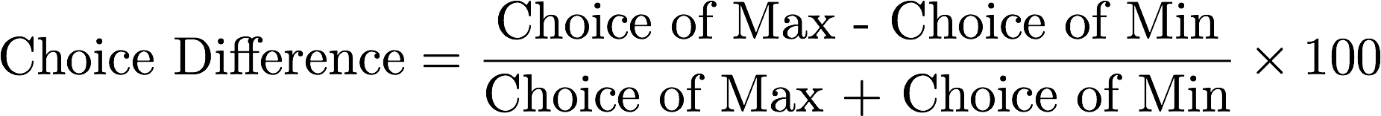
Apparently, for a player, the more coins he has (and the fewer coins his opponent has), the better.



But note that the weight assigned to this heuristic should not be heavy, because the difference in coins fluctuates intensely and frequently, due to the nature of the game, and only in the very end does the difference in coins matters.

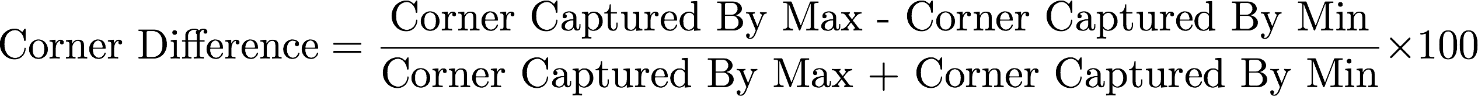
Choice Difference

This heuristic aims to increase one's flexibility in terms of choices and constrain the opponent's choices of valid moves.



Corner Difference

From the experience of playing the game, we learned that corners are crucial because once a corner is captured, it cannot be flipped. Furthermore, a player can build up stable coins around the corner. This is how one usually wins. Thus, this heuristic should receive a heavy weighting.



Evaluation Function



If the game has already ended, then returns +100 if the max player has won, or returns -100 if the AI has won.

**Conclusion**

To sum up was developed a Reversi game with AI based on alpha-beta pruning using Python programming language. As a heuristics was chosen Coin Difference, Choice Difference and Corner Difference. Evaluation function was combined with heuristics with 15%, 15% and 70%, respectively. Game provides functionality such as choosing colour, size of board and difficulties level.

**References**

1. Flyordie.com. (n.d.). Reversi Rules. [online] Available at: <http://www.flyordie.com/games/help/reversi/en/games_rules_reversi.html> [Accessed 3 Dec. 2017].
2. Kukreja, K. (2013). Heuristic/Evaluation Function for Reversi/Othello. [online] Everything Under The Sun. Available at: <https://kartikkukreja.wordpress.com/2013/03/30/heuristic-function-for-reversiothello/> [Accessed 3 Dec. 2017].
3. Sannidhanam, V., & Annamalai, M. (2015). An Analysis of Heuristics in Othello.
4. Stuart, R. and Norvig, P. (2010). Artificial Intelligence: A Modern Approach. 3rd ed. Englewood Cliffs: Prentice-Hall, pp.167-174.
5. <https://en.wikipedia.org/wiki/Reversi>