**CMPT 310**

PROJECT: Reversi with Monte-Carlo Tree Search

Contents:

* Introduction
* Functions
* Gameplay
* MCTS Comparisons
* Resources

**Introduction –**

The project implements an AI controlled game of Reversi using 2 Monte-Carlo Tree Search algorithms i.e., the Pure MCTS and MCTS using heuristics.

**Functions –**

Defined Macros:

* SIZE – Sets the size of the board for the game of Reversi.
* MAX\_RAND\_PLAYOUTS – Total number of random playouts performed by the MCTS functions

void display (char board[][SIZE]);

The function displays a 2D array that is passed as its parameter in the form of a Reversie game board.

int validMoves (char board[][SIZE], int moves[][SIZE], char player);

The validMoves function checks the board 2D array for every possible move the player char can make and sets that respective index of the board array as 1 (initially 0) in the moves 2D array and returns the count of the total number of moves for the player.

void moveStone (char board[][SIZE], int row, int col, char player);

moveStones function places the player char in the board[row][col] index and flips the opponent’s characters in every possible direction around it.

int\* make\_mcts\_move (char board[][SIZE], int moves[][SIZE], char player);

Function returns an array of the index of a move picked by the MCTS functions to start their random playouts.

int checkWinState (char board[][SIZE], int stonesOnBoard, char player, char computer);

Function returns an integer to determine if the game has ended (by returning 0, 1 or 2) and -1 if it has not ended.

* -1 – Match has not ended
* 0 – Match draw
* 1 – player won
* 2 – computer won

void pure\_mcts\_reversi (char board[][SIZE], int moves[][SIZE], char computer, int stonesOnBoard);

Function performs the Monte-Carlo Tree Search method for the AI by playing random playouts until the end of the game by making various random moves to determine which move has the best winning chance for the AI based on the number of wins, loses or draws during the random playouts.

void heuristic\_mcts\_reversi (char board[][SIZE], int moves[][SIZE], char computer, int stonesOnBoard);

Function performs like pure\_mcts\_reversi function with the added advantage that it contains a heuristic algorithm that helps it make moves that are more advantageous over the other in the case that they are available. This function uses the heuristics that it must make a move to the edge positions of the board over the others if those positions are valid moves giving it a higher winning chance over the pure mcts function.

**Gameplay –**

The game starts with the user having to choose the game mode that they want to play by entering 1, 2 or 3.

* 1 – Player vs Pure MCTS AI
* 2 – Player vs Heuristics MCTS AI
* 3 – Pure MCTS AI vs Heuristics MCTS AI

The user is then asked to choose the character (X or O) for the first player in the game mode. X represents black and O represents white. Black always plays the first round. Objective of the game is to take over the opponent’s pieces as much as possible by the end of the game.

End game states are when:

* The board is full of the players’ stones.
* A player’s stone no longer exist on the board
* No more moves are possible by either of the players

**MCTS Comparisons –**

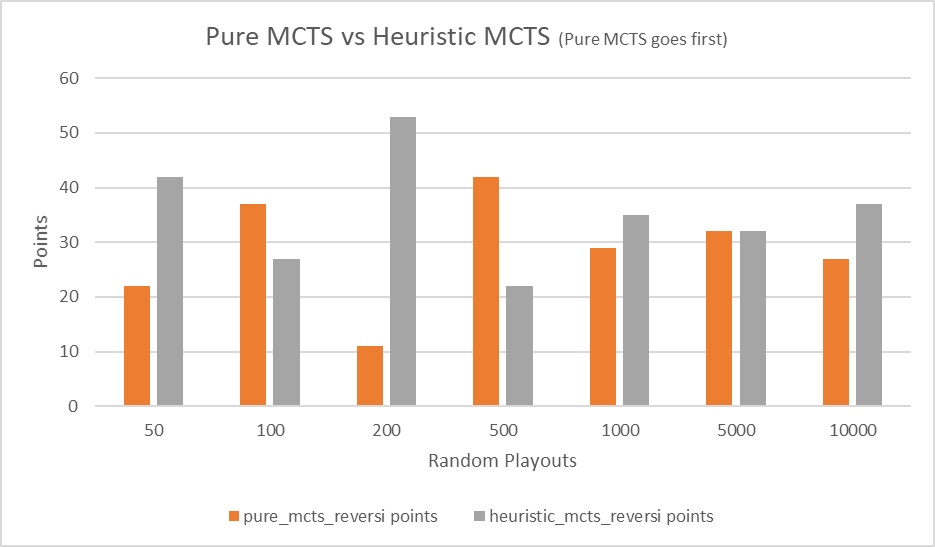
Playout results when pure\_mcts\_reversie plays first -

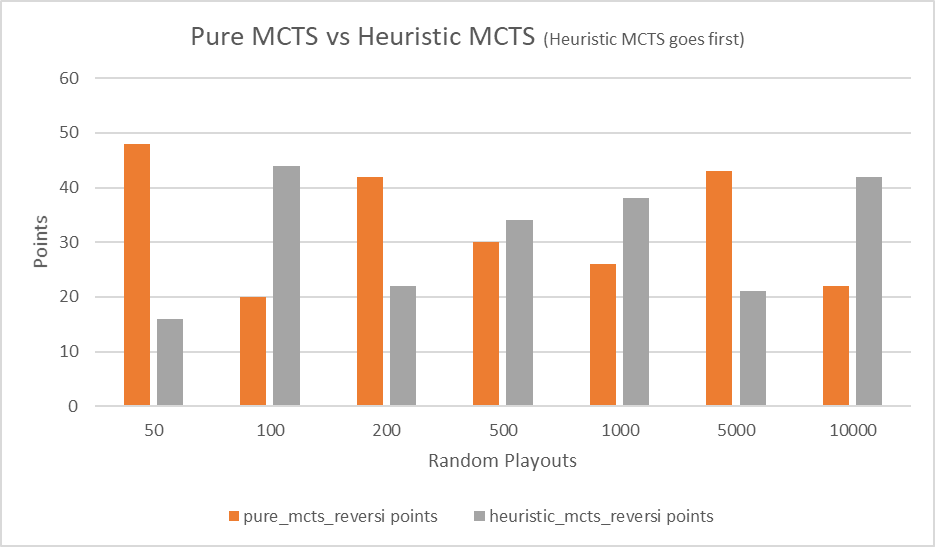
|  |  |  |
| --- | --- | --- |
| MAX\_RAND\_PLAYOUTS | pure\_mcts\_reversi points | heuristic\_mcts\_reversi points |
| 50 | 22 | 42 |
| 100 | 37 | 27 |
| 200 | 11 | 53 |
| 500 | 42 | 22 |
| 1000 | 29 | 35 |
| 5000 | 32 | 32 |
| 10000 | 27 | 37 |

Playout results when heuristic\_mcts\_reversie plays first -

|  |  |  |
| --- | --- | --- |
| MAX\_RAND\_PLAYOUTS | pure\_mcts\_reversi points | heuristic\_mcts\_reversi points |
| 50 | 48 | 16 |
| 100 | 20 | 44 |
| 200 | 42 | 22 |
| 500 | 30 | 34 |
| 1000 | 26 | 38 |
| 5000 | 43 | 21 |
| 10000 | 22 | 42 |

The heuristics MCTS algorithm seems to be winning 61.5% of the time over the pure MCTS algorithm





**Resources –**

<http://www.java2s.com/Code/C/String/REVERSIAnOthellotypegame.htm>

<https://cardgames.io/reversi/>

<https://github.com/kartikkukreja/blog-codes/blob/master/src/Heuristic%20Function%20for%20Reversi%20(Othello).cpp>

Discussed about bugs and other issues of the project with my peer Ritvik Shah (ritviks).