CS222: Data Structures & Algorithms

Project Outline: Milestones 2 & 3

Joel Osei-Asamoah

Yamoah Frimpong Attafuah

Lecturer: Mr. Attlee Gamundani

Faculty Intern: Osei Owusu-Banahene

Cohort A

20th April 2021.

INTRODUCTION TO CHECKERS

It is a game that is played on a 8 x 8 chessboard, where only the dark squares are used. The board is positioned such that each player has a light-side square on the right-side corner. View *fig.1* below.

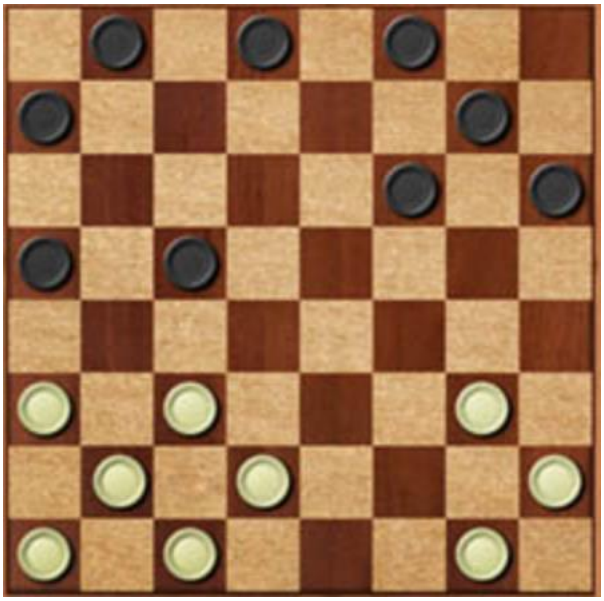

Fig.1: A checker gameboard setup

On the board are 24 discs, each player has 12. At the beginning of the game, Each player has 12 discs and places them on the 12 dark squares closest to them. Black opens the game and the players alternate turns. Which player gets black is random.

Game Rules

- The pieces move diagonally forward, unless that piece is a king
- A king is a piece that has reached the furthest row, when this happens, that piece can move diagonally (forward and backward)

- A regular piece making a non-capturing move can only move one square
- A capturing move is a move, where your piece leaps over the opponent's piece and lands in a straight diagonal line on the other side.
- Only one piece may be captured in a single jump, but multiple jumps are allowed in a single turn.
- If a player can capture, then the jump must be made.
- If more than one jump is available, then the player decides if he prefers it or not.
- Winner is declared when the number of the player's pieces is 0 or the number of the AI's pieces is 0.

SOLUTION ARCHITECTURE

Board Creation

A graphical user interface of an 8x8 checkers board patterned cream and brown will be painted in the window using Java window and graphics libraries.

The Java Mouse Listener and Action Listener classes will be used to receive click input on the checkers board from the user.

Game State Representation

The location of every piece in the rows and columns of the board will be represented by a two-dimensional integer array, where the index of each value represents the row and column of that particular tile, and the value of each integer in the array represents whether that tile is vacant or

the kind of piece within that tile. For example, the integer 1 may refer to a red piece, 2 may refer to a black piece, and 0 may refer to an empty tile.

Artificial Intelligence (AI)

Using the algorithm, decide the best move, and pass the current position of the piece and where it must go.

Algorithm (Minimax):

This algorithm would seek to produce the best move for the computer by, predicting the moves of its opponent. Below is how it would work.

- In each given state, that is the current set up of the board, asses all possible moves that the AI can make.
- Based on the possible moves, assess every possible move opponent (player) can make against the computer.
- Do a cost-benefit analysis to decide which move is then the best play
- Cost-benefit analysis is decided based on the score of the board if those moves are made.
- A score is a way of checking if the computer is winning or not. Which in the game of checkers could be total number of computer pieces – total number of opponent pieces.

The data structure than could be used to support this algorithm the tree, where the nodes represent the possible states of the board. The first node the root would represent the current state, the next level in the tree would represent the potential state that could be gotten from the

root node, based on the AI's moves. The level after would be the potential moves of the opponent after the AI has made their move.

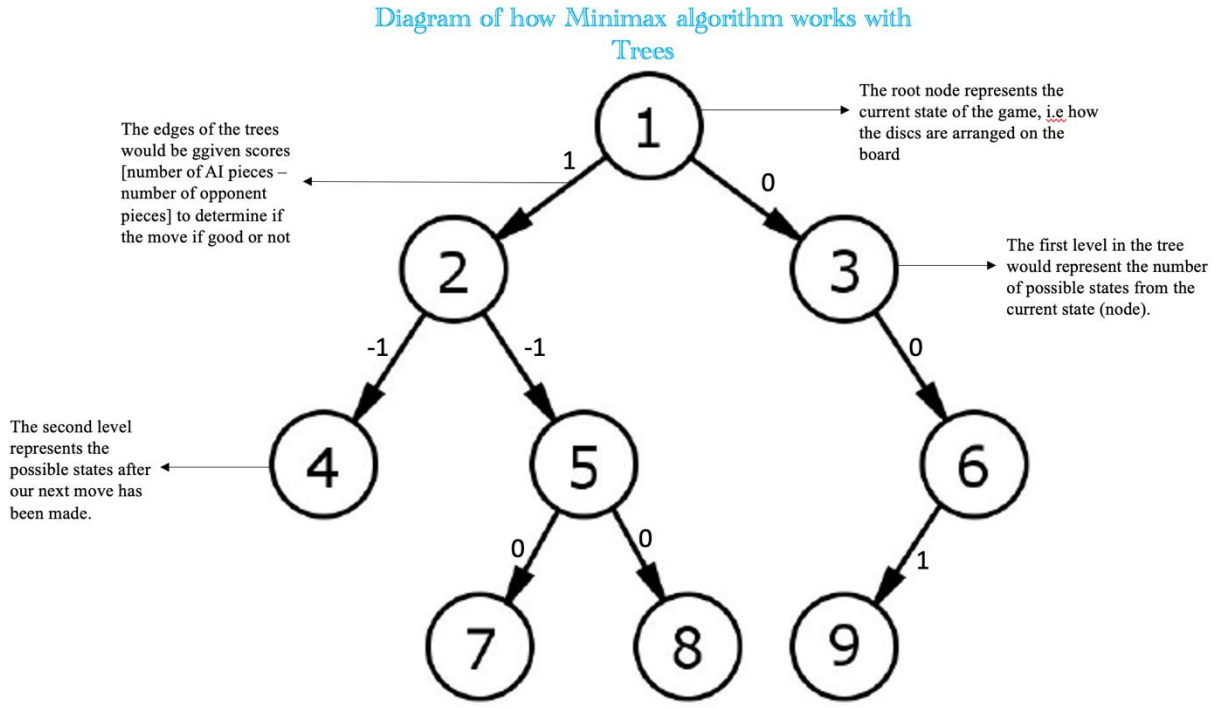

Fig.2: Diagram of tree to support minimax algorithm

From, the tree above, the AI would be able to see that in the long term the best move to make from '1' is '3' assuming the numbers represents state. Because in the long run it would give a total score of 1. But for the other state, though initially it gives a score of 1, it would just be cancelled out by the move of the opponent, resulting to a total score of 0.

Movement of AI's Pieces

- Once the AI decides where to play, move(), a function of the board, would be called with the following coordinate parameters; currentLocation, and newLocation.

- If this move results in a disc being changed to a king. It would be updated by the makeKing() function of the board.
- If this move captures the opponents piece, it would be cleared from the board using the clearDisc() function of the board, which takes currentLocation as a parameter, then the disc for the AI would be moved to the correct spot.
- If the opponent's disc is captured, number of opponent's (player's) pieces should be updated – decreased by the number of discs captured.

Movement of Player's Pieces

- Once the player decides where to play, they would click select the disc, they want to move, and click where they would want it to go.
- Once the spot is clicked the move() function of the board, would be called with the following parameters; currentLocation and newLocation.
- If this move results in a disc being changed to a king. It would be updated by the makeKing() function of the board
- If this move captures the AI's piece, it would be cleared from the board using the clearDisc() function of the board, which takes currentLocation as a parameter, then the disc for the player would be moved to the correct spot.
- When the AI's disc is captured, number of AI's pieces should be updated – decreased by the number of discs captured.