ASSIGNMENT 2
Connect4 Using Minmax

Artificial Intelligence

Alia Medhat | Artificial Intellignece | 20221440735

Malak Mahmoud | Artificial Intellignece | 20221445867

Nureen Ehab | Artificial Intellignece | 20221465124

Zeinab Mohamed | Artificial Intellignece | 20221310251

Table Of Contents Contents

Chapter 1: Introduction

1.1 Introduction

Connect four is a two player game in which the players take turns placing discs in a 6x7 grid. Each player has discs in their own specific color. Their aim is to get four of their own discs in a horizontal, vertical, or diagonal alignment. While doing so they have to prevent their opponent from getting four of their discs in a row. In this project, the purpose is to design and build a connect four playing robot, that can play connect four opposite a human. To do so, the robot has to be able to physically move and drop discs in the grid of the game, it also has to be able to figure out in which column to drop the discs, and detect where the player puts their discs. To achieve this, a demonstrator was built. It consists of a frame around the game, onto which motors are placed to move and drop discs into the game.

1.2 Backgroud

Connect 4 is a two-player game in which the players first choose a color and then take turns dropping their colored discs from the top into a grid. The pieces fall straight down, occupying the next available space within the column. The objective of the game is to connect-four ofone's own discs of the same color next to each other vertically, horizontally, or diagonally. Thetwo players keep playinguntil the board is full. The winner is the player having greater number of connected-fours.

1.3 Scope

The main objective is to create a connect 4 game which can figure out where to put discs in a connect four game, and also physically do it. The robot does not have to be able to sort the discs on its own and put them back in the depository. This can be done manually. Given the casual nature of this game, the calculation time is limited to 20 seconds, and the human player will always play first. This gives the player a chance to win, it also means that the AI algorithm doesn't have to be perfect given the time constraints, but good enough to beat the average human player.

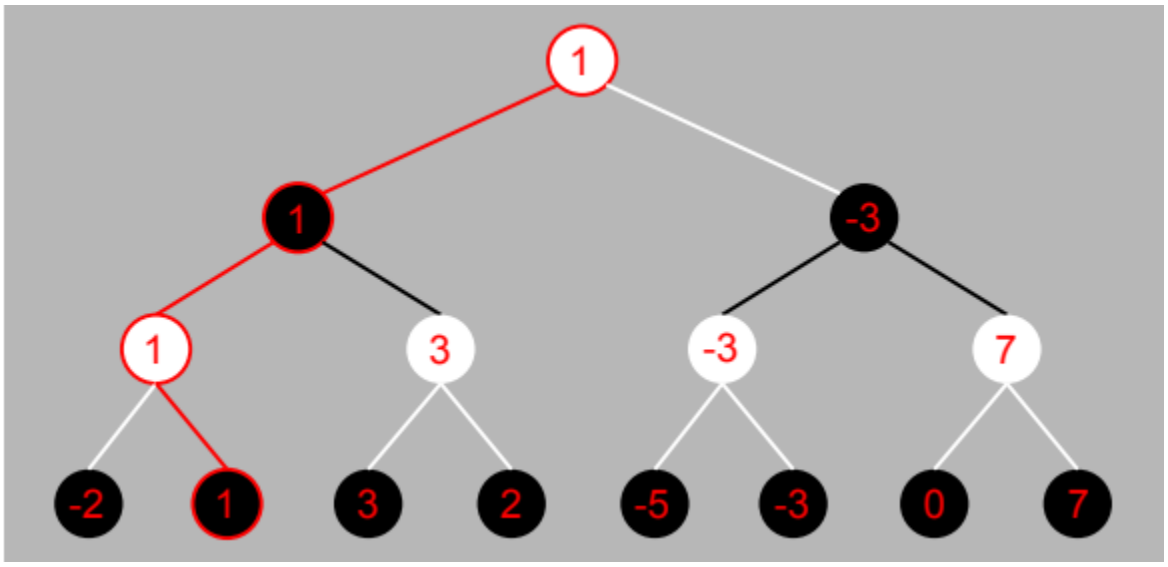
Chapter 2: Connect 4 Algorithm

2.1 Minimax

Solving Connect Four can be seen as finding the best path in a decision tree. Each node in the search tree is a certain position in the game, and each position has a score, it tells you how likely you are to win the game at that position. Given a position, a player has to choose a move which leads to, in the case of Connect Four, 7 other possible moves for the opponent. When it's your turn, you would like to choose a move that maximizes your score. On the other hand, when it's your opponent's turn, he would like to do the same for himself, thus minimizing your score on his turn. This process repeats itself until a winning move is found, or all possible moves have been exhausted in the case of a draw

Figure shows a simple example of how the Minimax algorithm works. The white nodes are maximizing nodes and the black ones are minimizing nodes. A white node will choose the node with the highest

score directly below it, and a black node will do the opposite, choose the node with the lowest score. The result in this case is the highlighted red line.



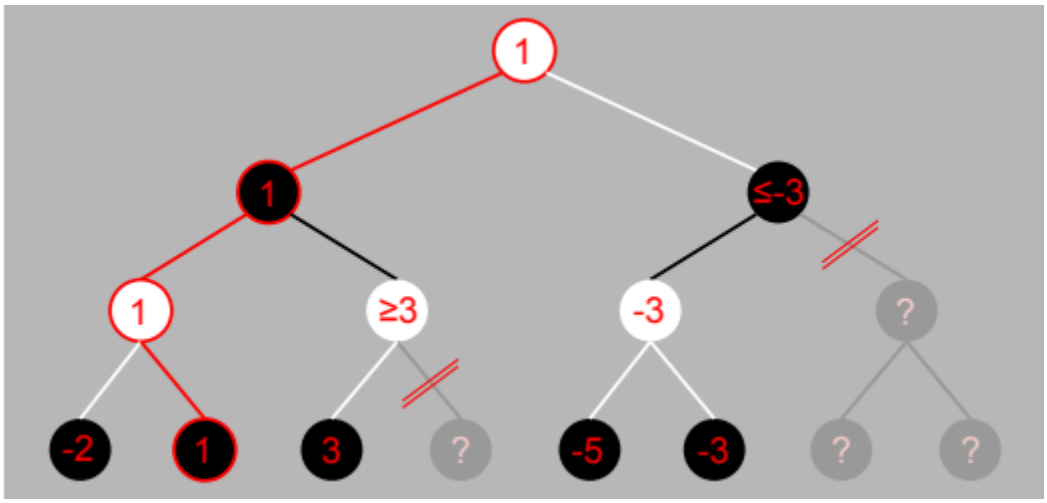
The Minimax algorithm searches through all the possible moves to find the best path, it is also known as a depth first search algorithm. This is however time consuming a standard 6x7 Connect Four game has $4.2*10^{12}$ possible positions , this is clearly not practical in a playable game setting. Other methods will have to be incorporated to trim the search tree and shorten the computing time.

2.2 Alpha beta pruning

Alpha beta pruning is a method to eliminate large parts of the search tree from consideration, this saves a lot of time and makes the search algorithm more efficient. Alpha is the worst score that the maximizing player is guaranteed to have, and beta is the worst score that the minimizing player is guaranteed to have. Before exploring the next node, it checks if alpha is greater than or equal to beta, if that's the case, then this node and all its descendants don't have to be explored because they won't affect the outcome of the game

Alpha beta pruning is a method to eliminate large parts of the search tree from consideration, this saves a lot of time and makes the search algorithm more efficient. Alpha is the worst score that the maximizing player is guaranteed to have, and beta is the worst score that the minimizing player is guaranteed to have. Before exploring the next node, it checks if alpha is greater than or equal to beta, if that's the case, then this node and all its descendants don't have to be explored because they won't affect with outcome of the game

The optimal path stays the same as before even though less nodes have been explored



2.3 Heuristics

Heuristic function is used in Minimax for evaluation of the current situation of the game. The final decision made by Minimax largely depends on how well the heuristic function is. Therefore, designing a reasonable heuristic function is pa-ramount. In our research, we designed a heuristic function for Connect-4. To evaluate the current situation of the game, the heuristic function firstly looks for different features on the board and then gives them proper values. Finally, the heuristic function returns a summation of all the values of features on the chess board. We also introduced another heuristic function in . It evaluates the board in a different way. It doesn't look for features on the board. Instead, it evaluates each square on the board and gives them a proper value. We want the two heuristic functions to fight against each other so that we can assess them.

Chapter 3: The Project Classes

3.1 The Classes

The game is consists of 3 packages containing code files which are:

1. **Connect4:**
   - **Connect4.java:** contains the main class and check winner functions and the main function.
2. **Connect4Game:**
   - **Connect4AIPlayer:** which extends from the Connect4Player class, contains the minimax function and alfa beta pruning method which is the core of our project.
   - **Connect4Board:** which contain the function which deals mainly with the board and contains of important functions responsible for printing, modifying, and checking the board
   - **Connect4Game:** which contains all the main functions of the connect 4 game rules.
   - **Connect4HumanPlayer:** a subclass which extends from the Connect4Player
   - **Connect4Player:** a superclass for the other subclasses
3. **Connect4GUI:**
   - **Connect4Frame:** which contains the GUI of our connect 4 game implemented by using the JavaFX .

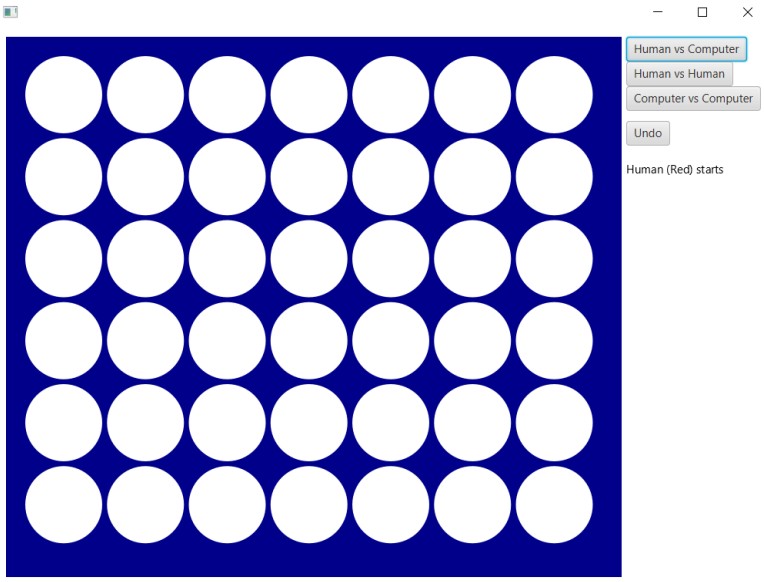
3.2 The Bonus Features

- The user can choose the game options (“Human Vs Computer”) or (“Human Vs Human”) or (“Computer Vs Computer”) from the GUI frame .
- The user can undo the previous move that the player by pressing the “undo” button from the GUI Frame or by pressing any game option and it will restart the game .

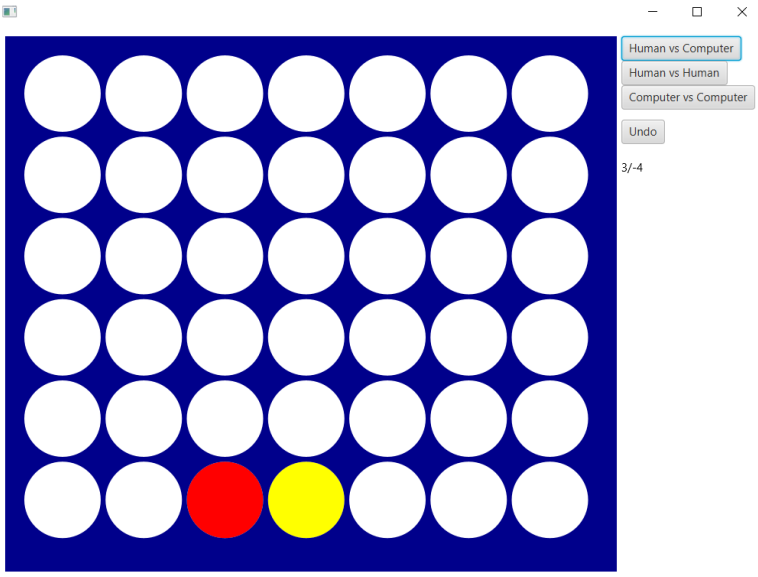
Chapter 4: Tests & Results

4.1 Sample Runs (GUI)

- We can start the game by pressing any game option we want to play , and it will start the game



- The human player chooses a column to drop his coin, then the bot player continues the game.



- The game keep going until the board is full , and the winner is the player who got the most wins by connect 4 coins.

4.2 Sample Run(Output)

- In the console, we will find the depth, lines and the number of pieces and the of each play the player has done

```
Output - Conect4 project (run) #2
run:
Connect 4
RED:2
Computer (Yellow) is thinking (depth=10,lines=69,pieces=1) ...
YELLOW:3
RED:1
Computer (Yellow) is thinking (depth=10,lines=66,pieces=3) ...
YELLOW:3
RED:2
Computer (Yellow) is thinking (depth=10,lines=61,pieces=5) ...
YELLOW:3
RED:4
Computer (Yellow) is thinking (depth=10,lines=57,pieces=7) ...
YELLOW:3
```