MINISTRY OF EDUCATION AND TRAINING

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

------- * -------**

COURSE PROJECT REPORT

INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Topic: Tic-tac-toe game

Teacher: **Dr. Nguyen Nhat Quang**

Student: **Group 15 - class 136462**

Nguyen Viet Trung	20214934
Do Hoang Tuan	20214939
Dau Van Can	20214879
Doan Minh Viet	20210933

Ha Noi, 2nd January 2023

Index

A. description of the practical problem
B. details of the solution used to solve the problem
C. The existing (i.e., third-party) solutions, software packages, datasets, etc. that are used/exploited
D. The problems/issues/difficulties occur during the execution of the course project, and how you handled them
E. The discussion, findings, conclusion and proposal for improvement of the solution and the system in future

A. Description of the particle problem

The game in which two players alternately put Xs and Os in compartments of a figure formed by two vertical lines crossing two horizontal lines and each tries to get 5 consecutive Xs or 5 consecutive Os in a horizontal, vertical, or diagonal row before the opponent does.

B. Details of the solution used to solve the problem

I.Board: A Board Status class

1. Board Status class have 4 components:
- size: board size
- board: a list of lines which each line is a list contain element (‘x’, ‘o’ , ‘ ‘)
- move_history : a list of positions that have been filled
- possible_moves: a list of positions that is reasonable to choose
2. Create score_of_row function to return the score of blocks of 5 consecutive cells in a sequence.
3. Create score_of_col function that returns a dict of the form {-1:a[-1],0:a[0]….,5:a[5]} (a[i] is the number of blocks of 5 cells consecutive having scores equal to i)
4. Create a score_of_col_contain_cell function that returns the score of the column containing (y,x) in 4 directions.
5. Create a limit function to find the furthest position of dy,dx in length on board.
6. Create function sum_direction_value to merge points in each direction of the score of col one

7. Create is_win function to return the game status(LOST, WON , DRAW, Continue playing)
8. Create add_his function and update poss_move function to update the playing history and the moves can go into 2 list move_history and possible_moves
9. Create next move function to create temporary Board_Status() that has appended the move and updated it own move_history and possible_moves
10. Create topmove function to return a sorted decreasing list of moves that the best move (maybe) at the first of the list based on Heuristic

II.Heuristic

1.Create a winning_situation function that return a score of {3,4,5} for winning states and returns 0 if it is in a normal state

1.Create a heuristic function that returns the optimum of possible moves by returning the score res = adv + dis with adv being the player's advantage and dis being the opponent's disadvantage via a heuristic evaluation function

III.AI

1.Minimax algorithms

Considering the board.topmove() , use the minimax algorithm with a depth (in the source code, we give that value = 2) to choose the next move by calculating the values for the Nodes in the game tree then find the Node with the appropriate value to take the next step.

At each node there is also a corresponding score determined in some way. Based on this game tree, we can find good moves to win.

The move search strategy is to choose a node in the tree such that the move is good.

Node of class MAX: assign it the maximum value of that Node.

Node of class MIN: assign it the smallest value of that Node. From these values, the player will choose for himself the most reasonable next move.

If the search limit is reached specifically for the leaf notes on the tree: Calculate the value of the current position relative to the player there. Memorize the results.

If the level under consideration is that of the minimum player (the MIN node), apply this Minimax procedure to its children. Remember the smallest result.

If the level under consideration is that of the maximum player (the MAX node), apply this Minimax procedure to its children. Remember the biggest result.

2.Alpha-beta pruning algorithm

- If a certain search branch can't improve with the value we already have, then we don't need to consider that function anymore. Use alpha and beta to compare and eliminate cases that will not need to be considered in the minimax algorithm.

- Alpha saves the machine's best moves, beta saves the best value of the player. If any time alpha >= beta, then the player will inevitably choose the best move for them and force a worse move than the alpha for the machine, so no further steps need to be considered.

IV.Graphic

Use the Turtle module to draw the output state and execute the game.

C. The existing solutions, software packages, datasets, etc. that are used/exploited:

- Our source code is based on Github by Link:
GitHub - BuiNgocHai/Engine-Caro: Engine Caro
- Our source code uses libraries: turtle, random, math

D. The problems/issues/difficulties occur during the execution of the course project, and how you handled them

In practice time, we have many troubles with the heuristic function and for mini bugs so we try many times to test to choose the optimal function and we read the code again and again to find the bugs and debug.

E. The discussion, findings, conclusion and proposal for improvement of the solution and the system in future

After testing many times, we have concluded that the intelligence of the AI system mostly depends on a heuristic function (how to calculate the score of a move) and the depth of minimax algorithms. We had tried the higher depth value and it became overloaded at the depth of 4, to solve this issue, we tried to cut off as much as possible in finding the best move(just take 5 most efficient moves for example) to reduce the complexity. This may not give the most effective move but it reduces the time to find a move and can try a higher depth value. In the future, we will try as many as possible to find out the most optimal heuristic function and try with the higher value of the depth of minimax algorithms.Besides,we can use numpy to minimize running time.