Artificial Intelligence (106266)

Final Project Report

Project Name :- Tic Tac Toe

Group Members :-

Muhammad Faraz 9536

Kashaf Alam Lodhi 9198

Introduction :-

Tic-tac-toe (also known as noughts and crosses or Xs and Os) is a paper and pencil for two players, *X* and *O*, who take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.

In order to solve Tic Tac Toe, we need to go deeper than just to think about it as a game where two players place X's and O's on the board. Formally speaking, Tic Tac Toe is a zero-sum and perfect information game. It means that each participant's gain is equal to the other participants' losses and we know *everything* about the current game state.

Logic Behind Tic Tac Toe :-

One of the player chooses 'O' and the other 'X' to mark their respective cells. The game starts with one of the players and the game ends when one of the players has one whole row/ column/ diagonal filled with his/her respective character ('O' or 'X'). If no one wins, then the game is said to be draw.

How We Can Beat AI In Tic Tac Toe :-

When you're the first one up, there is a simple strategy on how to win tic tac toe: put your 'X' in any corner. This move will pretty much send you to the winner's circle every time, so long as your opponent doesn't put their first 'O' in the center box. This can make it harder to win, but it can happen.

Algorithm We Use In Our Project :-

Minimax Algorithm :-

Minimax Algorithm is a decision rule formulated for 2 player zero-sum games (Tic-Tac-Toe, Chess, Go, etc.). This algorithm sees a few steps ahead and puts itself in the shoes of its opponent.

It is a recursive or backtracking algorithm which is used in decision-making and game theory. It provides an optimal move for the player assuming that opponent is also playing optimally. ... This Algorithm computes the minimax decision for the current state.

You can think of the algorithm as a representation of the human thought process of saying, “OK, if I make this move, then my opponent can only make two moves, and each of those would let me win. So this is the right move to make.”

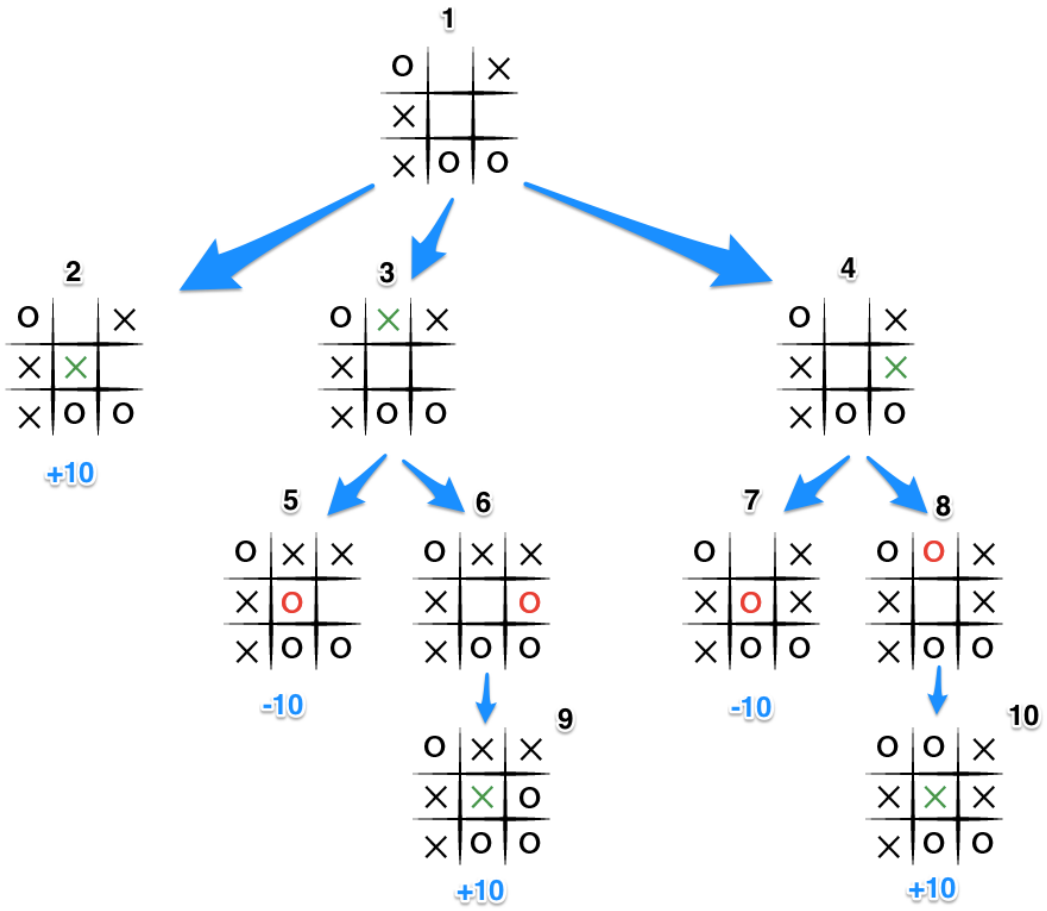
The key to the Minimax algorithm is a back and forth between the two players, where the player whose "turn it is" desires to pick the move with the maximum score. In turn, the scores for each of the available moves are determined by the opposing player deciding which of its available moves has the minimum score. And the scores for the opposing players moves are again determined by the turn-taking player trying to maximize its score and so on all the way down the move tree to an end state.

A description for the algorithm, assuming X is the "turn taking player," would look something like:

- If the game is over, return the score from X's perspective.
- Otherwise get a list of new game states for every possible move
- Create a scores list
- For each of these states add the minimax result of that state to the scores list
- If it's X's turn, return the maximum score from the scores list
- If it's O's turn, return the minimum score from the scores list

You'll notice that this algorithm is recursive, it flips back and forth between the players until a final score is found.

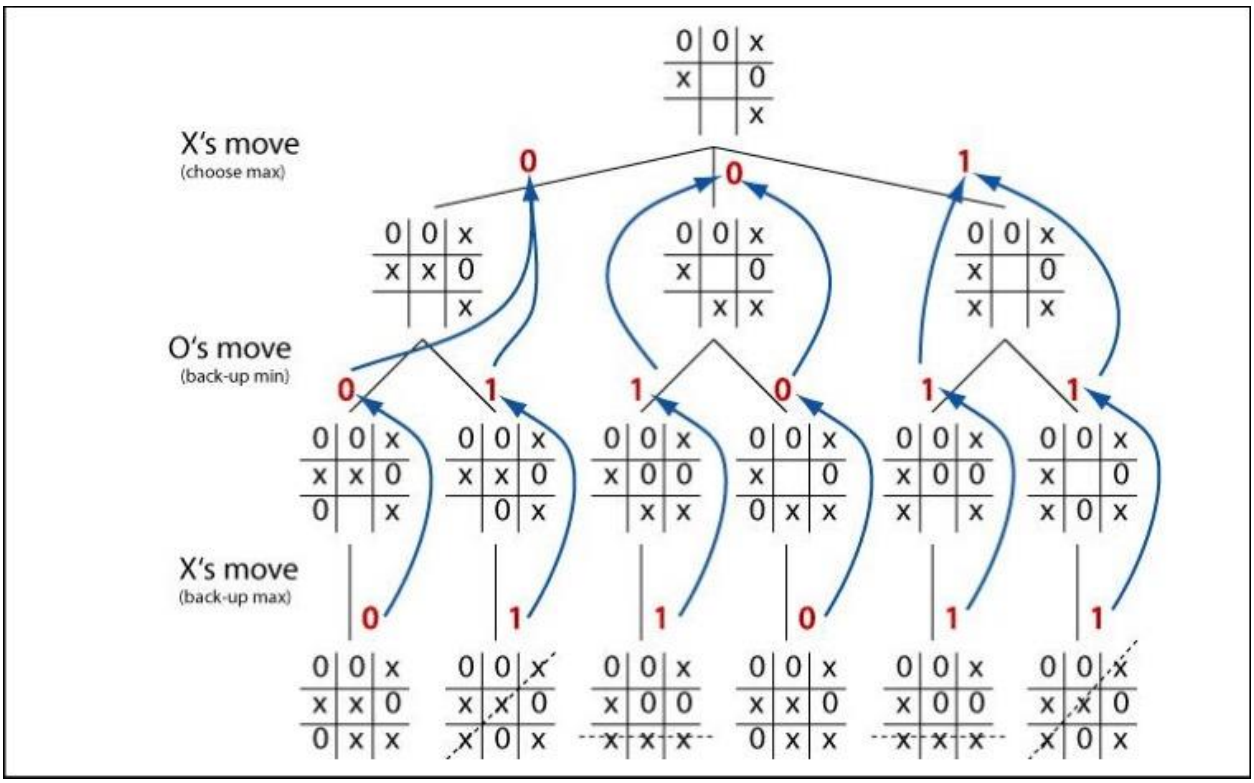
Let's walk through the algorithm's execution with the full move tree, and show why, algorithmically, the instant winning move will be picked:



- It's X's turn in state 1. X generates the states 2, 3, and 4 and calls minimax on those states.
- State 2 pushes the score of +10 to state 1's score list, because the game is in an end state.
- State 3 and 4 are not in end states, so 3 generates states 5 and 6 and calls minimax on them, while state 4 generates states 7 and 8 and calls minimax on them.
- State 5 pushes a score of -10 onto state 3's score list, while the same happens for state 7 which pushes a score of -10 onto state 4's score list.
- State 6 and 8 generate the only available moves, which are end states, and so both of them add the score of +10 to the move lists of states 3 and 4.

- Because it is O's turn in both state 3 and 4, O will seek to find the minimum score, and given the choice between -10 and +10, both states 3 and 4 will yield -10.
- Finally the score list for states 2, 3, and 4 are populated with +10, -10 and -10 respectively, and state 1 seeking to maximize the score will chose the winning move with score +10, state 2.

That is certainly a lot to take in. And that is why we have a computer execute this algorithm.



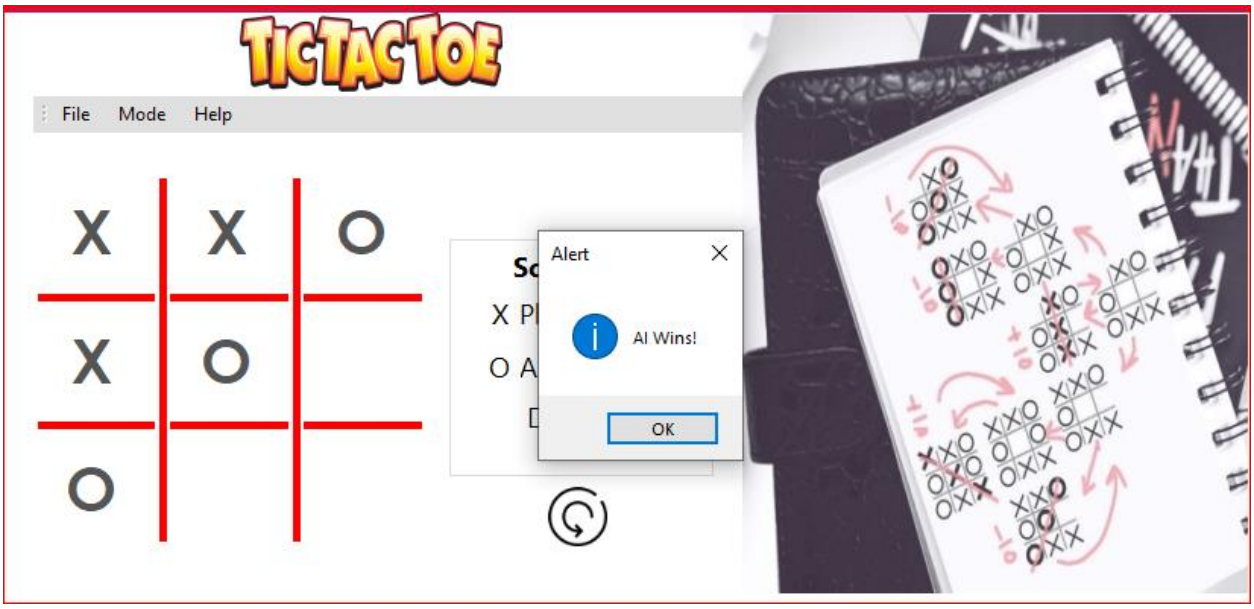
Output (GUI) :-

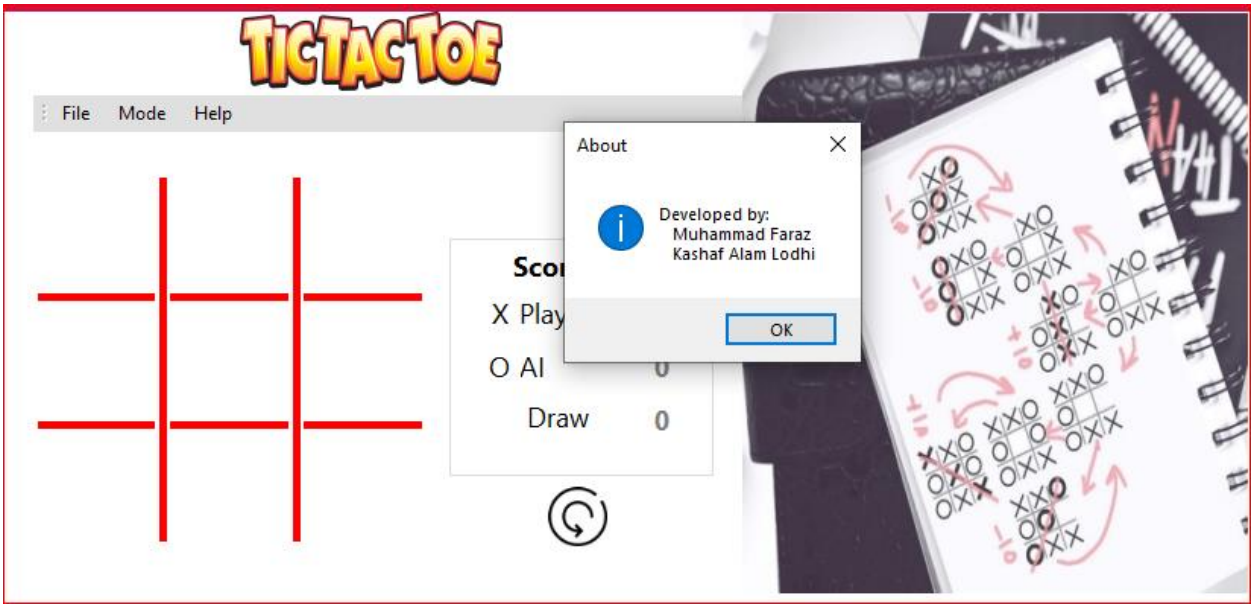
In our project we have 2 modes of tic tac toe.

1. Player 1 VS Player 2.
2. Player VS AI.

1. Player 1 VS Player 2 :-

2. Player VS AI :-




TIC TAC TOE
File
Mode
Help
About
Developed by:
Muhammad Faraz
Kashaf Alam Lodhi
OK
X Play
O AI
Draw
0
0