Tic Tac Toe

By Sudais Baig,
Burare Hassan,
and Omar Zia ud Din

June 2020

1 What is Tic Tac Toe?

A game in which two players seek in alternate turns to complete a row, a column, or a diagonal with either three O's or three X's drawn in the spaces of a grid of nine squares; noughts and crosses.

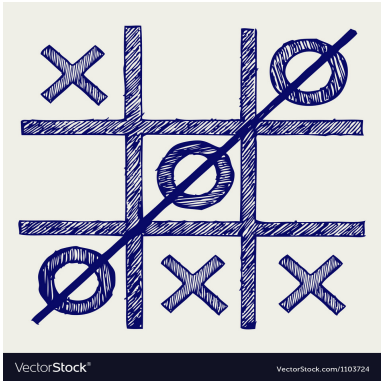

Figure 1: TicTacToe Board

2 General Idea of our Project

In this project, our group aims to use a recursive algorithm called 'The Minimax Algorithm'. This algorithm works on the basics of a Tree data structure performing simulations for every move the opponent makes and improvises it's moves accordingly. We thrive to design an algorithm that recursively calls itself with the current status of the grid and, the most recent move played by the opponent passed as it's parameters. The algorithm, then, keeps recursively going level by level deeper into the game until it reaches a terminal state and returns a score one level up. This way our program simulates all the possibilities to

win the game, with the current status of the grid, and executes the best move. There are only 3 9 = 19,683 possible combinations of placing X, O or <blank> in the grid, and not all of those are valid. Hence, due to the relatively small state space, the algorithm can easily search the whole game tree for an optimal solution.

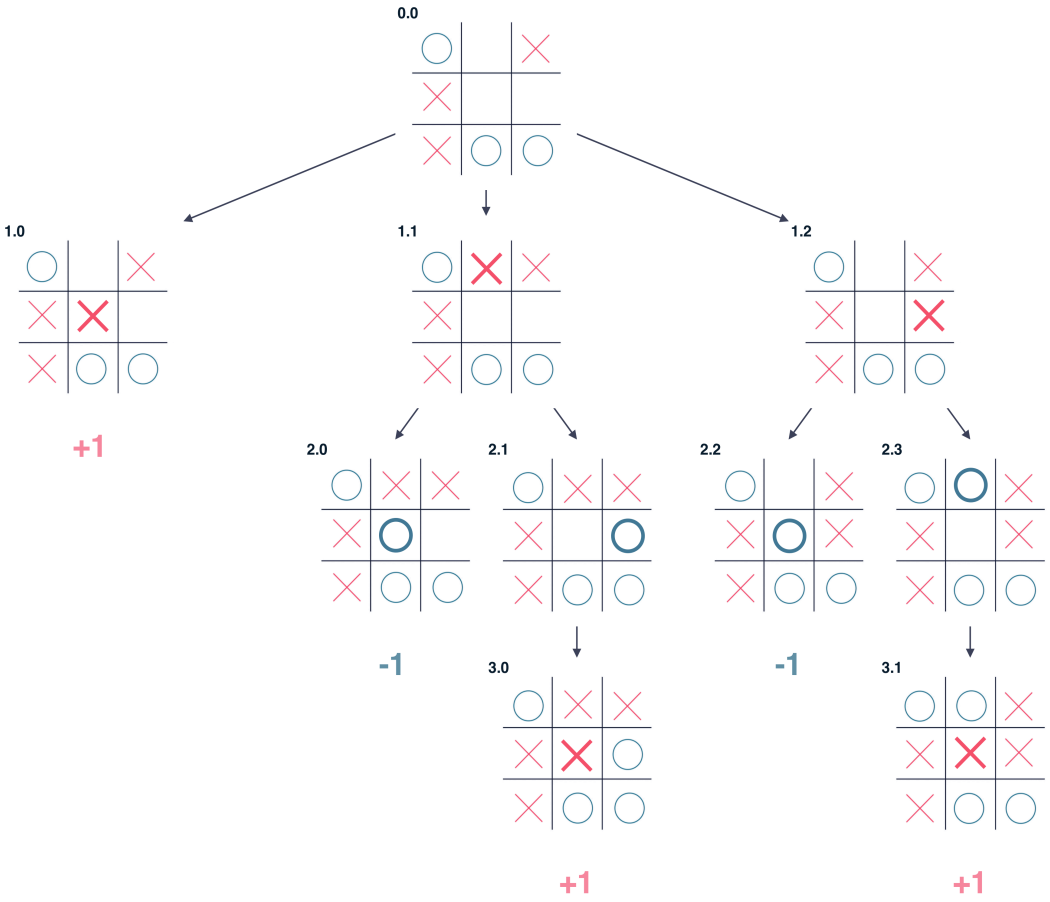

Figure 2: Minimax presentation

3 Functions in our Code

3.1 Checkwinner()

This function checks the the winner. After every move it iterates over the board horizontally, vertically and diagonally. If it finds 3 consecutives Xs or Os in any above format, it returns the 'Winner'. If it doesn't find any winner and there

are no open spots left on board then it returns 'Tie'.

3.2 Bestmove()

This function decides the best move of computer. It checks open spots on board, calls minimax function for each spot and calculates score for each choice.
Choices in which Computer is wining is given a score of 10. Choices in which Computer is losing is given a score of -10. Choices leading towards tie is given a score of 0.
As we find the best move(the move with best score), the move is printed on board and it returns checkwinner function to check winner if available.

3.3 Minimax()

This function recursively plays the whole game for each possible option(open spot) available on board for each turn of computer and in doing so it finds the best move. As for each spot it optimally chooses the best choice a human can make and then counters it with the best choice computer can make. The choices leading to best score is returned by this function.

3.4 Playersturn()

Human makes the first turn, so this function records the move made by human and prints it on board. Then, as now the computer now has to make a move so this function calls the bestmove function.
Also, to run the game only this function is called with parameter of current player.

4 Libraries for GUI

following are the libraries used to develop this program
- tkinter
- copy
- itemgetter

5 Time Complexity

The time complexity of minimax is $O(b^m)$, where b is the number of legal moves at each point and m is the maximum depth of the tree.