Johnson & Wales University

Tic-Tac-Toe using Minimax Algorithm

Craig Sampson

ENGN2062 – Artificial Intelligence

Dr. DiFilippo

The minimax algorithm is a recursive algorithm, for choosing the next move in a two-player, zero sum game. Tic-tac-toe, checkers, chess, and connect 4 are some of the games that the minimax algorithm can be applied to. It can also be described as a decision rule, for use in game theory.

One player is the minimizer, trying to get the lowest possible score, and the other player is the maximizer trying to get the highest possible score. Each board state has a value associated with it and based on who has the upper hand it will either be a positive value, negative value, or 0 if a tie is going to happen. In my code, the player/user is the minimizer, and the computer/AI is the maximizer.

The minimax function is a sub-section of rule based expert systems, this is shown through a game tree. In more detail how this algorithm works, depending on how the board looks, the function in the code will go through each possible move that could be made and once the game ends in a win, lose or tie it will give that move a value, then move to the next possible move until all are covered.

My code was written in python using the Spyder IDE. I started my code with creating the tic-tac-toe game first. The way I did that was making various functions that all work together. At first, I created an empty 9 value dictionary, for the 9 spots of the tic-tac-toe board. My first function prints out the board with the values from the dictionary, anytime the player needs to see the board, this function is called. My next function checks if a certain space is free on the board, by returning true if it is empty and false if it is occupied. The next few functions I created do a few simple things, checks for a win, checks who won, X or O, and checks for a draw. These functions are used in my insertLetter() function that does most of the work in creating the game. It checks if a space is open, then inserts a letter. And after a letter is inserted, it checks for a draw, and a win.

The two functions that I needed in order for the computer to be the second player is minimax, which applies the minimax algorithm and compMove, which utilizes the minimax function to determine where the best move is. the minimax function first checks for a draw, player win, or computer win, then if none of those functions are true it continues to the actual application of minimax. It first sets a best score which can be any negative number (since the computer is the minimizer). It then goes through each empty space on the board, gives it a value by recursively calling the minimax function, then sets the best score as the highest value and returns the best score value.

The output of this code does exactly what you would expect, it allows you to play tic-tac-toe vs a computer. It first shows you what number associates to which part of the board (1-9, top to bottom and left to right). It then allows you to choose whether or not you want to go first, then the game starts and as soon as you make your move the board is printed out and the computer automatically makes its move, and the board is printed out again.

In order to make this program better, I wanted to create an interface that you can click on and play the game, instead of playing it in the spyder console. I would use the TKinter library or something similar to do this.