Self-Study Report

We will study a sorting algorithm called the Minimax algorithm, which is widely used to generate program code in games like Tic-Tac-Toe. It is usually used in a 2-player game and zero-sum game, which means when one player wins and gets marks, his opponents will lose one mark, and adding all gains and losses in the games will be zero. Adversarial search is not the same as ordinary searches such as linear search and binary search we learn in the lecture. The adversarial search considers the rivals' optimal move in the mix and generates the optimal move for the player against the opponent. In the algorithm, one player is assigned the maximizer and the other the minimizer. Maximizer wants to choose the move with the highest score and the minimizer will therefore choose the lowest score in a zero-sum game situation.

Assume there is a game tree. We commence with the root node at the top of the tree and the maximizer. We will choose the maximum score from its child node below. The minimizer starts from the leaf node and finds the minimum among the leaves. It keeps recursively running until it reaches the root node. Assume the height of the game tree is n, in other words, the tree has n layers from the top layer name 1 and the lowest layer name n. If the nth layer is set to be minimizer turn and the n-1 layer will be the maximizer turn and n-2 will be back to minimizer turn (and n-3 will be the maximizer turn) the algorithm will keep running till layer 1 which must be the maximum term. The nth layer of the tree can be set to be maximizer tern and the root node will therefore be minimizer turn.

Minimax algorithm with alternative moves:

The minimax algorithm is a recursive algorithm, usually used in two-player games to choose the next step. A value is associated with each position or state in the game. This value is calculated by the position evaluation function, and it represents how well the player got to the position. The player then makes a move that maximizes the minimum value of the position resulting from the opponent's possible subsequent moves. If it is A's turn to go, A assigns a value to each of their legal moves.

For example, one possible allocation method is to assign A a certain win as +1 and B as -1. This led to a combinatorial game theory developed by JH Conway. Another way is to set a rule that if the outcome of the move is that A wins, assign positive infinity, and if B wins, assign infinity. If any other move the value to A is the minimum possible value B could produce. Then A is the maximizing player, and B is the minimizing player, so it is called the minimax algorithm. The above algorithm will assign a value of positive infinity or negative infinity to any position, so the value of each position will be some value that wins or loses in the end. This is usually only possible in more complex games such as Go, Chess, etc., so unless the game is about to end, it is computationally infeasible to predict the completion of the game, and instead, positions are given constraints as to what they will be The probability that one or the other player will win.

This can be extended if we can provide a heuristic evaluation function that assigns values to non-final game states without considering all possible subsequent complete sequences. We can then restrict the minimax algorithm to only look at a certain number of advances. This number is called "look ahead" and is measured in "layers". For example, the chess computer Deep Blue (the first at the time to beat reigning world champion Garry Kasparov) looked ahead at least 12 layers and then applied a heuristic evaluation function.

The algorithm can be regarded as the nodes of the game tree, and the effective branching factor of the tree is the average number of points of each node. The number of points to find usually grows exponentially with the number of layers. Therefore, the number of nodes explored for analyzing a game is the branching factor to the power of the level. Therefore, using the minimax algorithm to conduct a complete analysis of games with many operating objects and complex operations is impossible.

Minimax algorithm with Tic Tac Toe:

When applying minimax with Tic Tac Toe, the code will be simpler than what we did in the final report. After every move each player picks, the game tree will be recursively generated to analyze the best possible move till the states exit. There will be two states to exit. One player wins and both players lose the game. We simply return integer one for winning the game and integer 0 for both players who lose to exit the game. After checking the stage, if it is not the exit stage, we create a list containing all the moves the player can pick. Then, we create another list called marks. The Minimax algorithm assigns +5 marks for winning the game and -5 marks for losing the game (zero-sum game) and 0 if no player wins.

We assign the minimax result to each of the moves to the new lists. If player one is the maximizer, return the maximum score (+5), and vice versa. If the stage is 0, a nested loop in the algorithm will generate all states and assign the mark to them.

Summary:

Overall, the minimax algorithm is one of the most popular algorithms for computer board games. It is widely applied in turn-based games. It can be a good choice when players have complete information about the game. But it may not be the best choice for games with exceptionally high branching factors.

Time complexity:

The time complexity of the minimax algorithm for Tic Tac Toe is O(n) which is linear time complexity, the first if-cause is the escape point of the algorithm when the node is the terminal node which is of O(1) time complexity. The elif cause is the recursive loop for the maximizer, the for node loop all the children of the maximizer. This complexity is of O(n). The mechanism is the same for the last else cause which is the recursive loop for the minimizer. This complexity is also of O(n). Therefore, the total time complexity is O(n) as there is no nested loop and reduction of the recursive size.

Reference:

1)Fox, J. (2021, November 10). *Tic Tac Toe: Understanding the minimax algorithm*. Never Stop Building - Crafting Wood with Japanese Techniques. Retrieved April 13, 2023, from https://www.neverstopbuilding.com/blog/minimax

2)Wikimedia Foundation. (2023, January 25). *Minimax*. Wikipedia. Retrieved April 13, 2023, from https://en.wikipedia.org/wiki/Minimax