Design and Analysis of Algorithms

Project Report

Minimax algorithm with Alpha-Beta Pruning

Implementation of an unbeatable TicTacToe game

Name: **AMIT KUMAR**

SRN: **PES1201701295**

Semester : **IV**
Section : **B**
Project No : **30**
Year: **2019**

Introduction

MiniMax algorithm is used to implement basic AI or game logic in 2 player games. It is a recursive backtracking algorithm which is used to choose an optimal move for a player assuming that the other player is also playing optimally.The most common scenario is implementing a perfect Tic-Tac-Toe player. All possible moves from a particular state is represented as a structure in the form of a tree known as the Game Tree allowing you to move from a state of the game to the next state.

Alpha-beta pruning is an optimization technique for minimax algorithm based on the Branch and Bound algorithm design paradigm, where we will generate uppermost and lowermost possible values to our optimal solution and using them, discard any decision which cannot possibly yield a better solution than the one we have so far.

Idea of Minimax Algorithm

Given that two players are playing a game optimally (playing to win), MiniMax algorithm tells you what is the best move that a player should pick at any state of the game. So, the input to MiniMax algorithm would be –

- **the state of the game**
- **information on which player's turn it is.**

It returns –

- **max/min score which can be achieved for the given player for the given game state.**
- **best move which can be played by given player.**

Optimization

Game trees are, in general, very time consuming to build, and it's only for simple games that it can be generated in a short time. If there are

b legal moves, i.e., b nodes at each point and the maximum depth of the tree is m, the time complexity of the minimax algorithm is of the order (b^m)(O(b^m)).

To curb this situation, there are a few optimizations that can be added to the algorithm. Fortunately, it is viable to find the actual minimax decision without even looking at every node of the game tree. Hence, we eliminate nodes from the tree without analyzing, and this process is called pruning.

Idea of Alpha-beta pruning

Two additional parameters α and β is maintained. α is the best score achievable by the max player so far and β is the best score achievable by the min player so far. These values will be passed down to recursion calls via arguments. We will try to use α and β to prune our search tree by skipping choices which can't possibly give us a better solution.

The pruning happens for these cases –

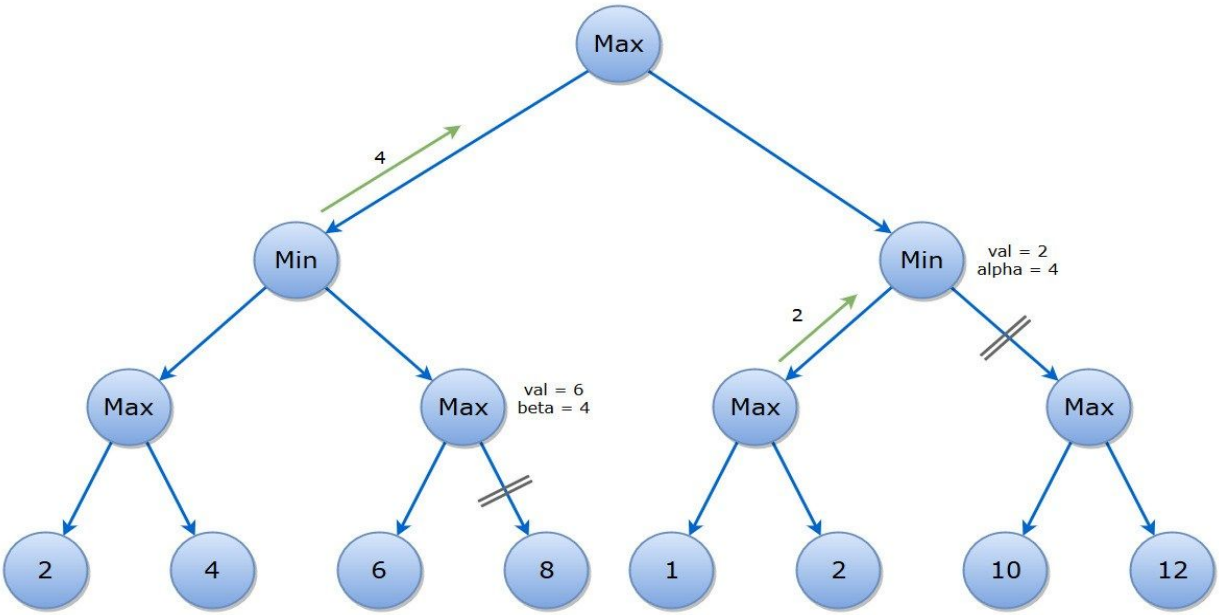
- **val ≥ β in a Max node**
- **val ≤ α in a Min node**

The Game

In the game of Tic-Tac-Toe, there are two players, player X and player O. There's a scoreboard which records a number called "score".In the game of Tic-Tac-Toe, there are two players, player X and player O. Now imagine there's a scoreboard which displays a huge number called "score", and –

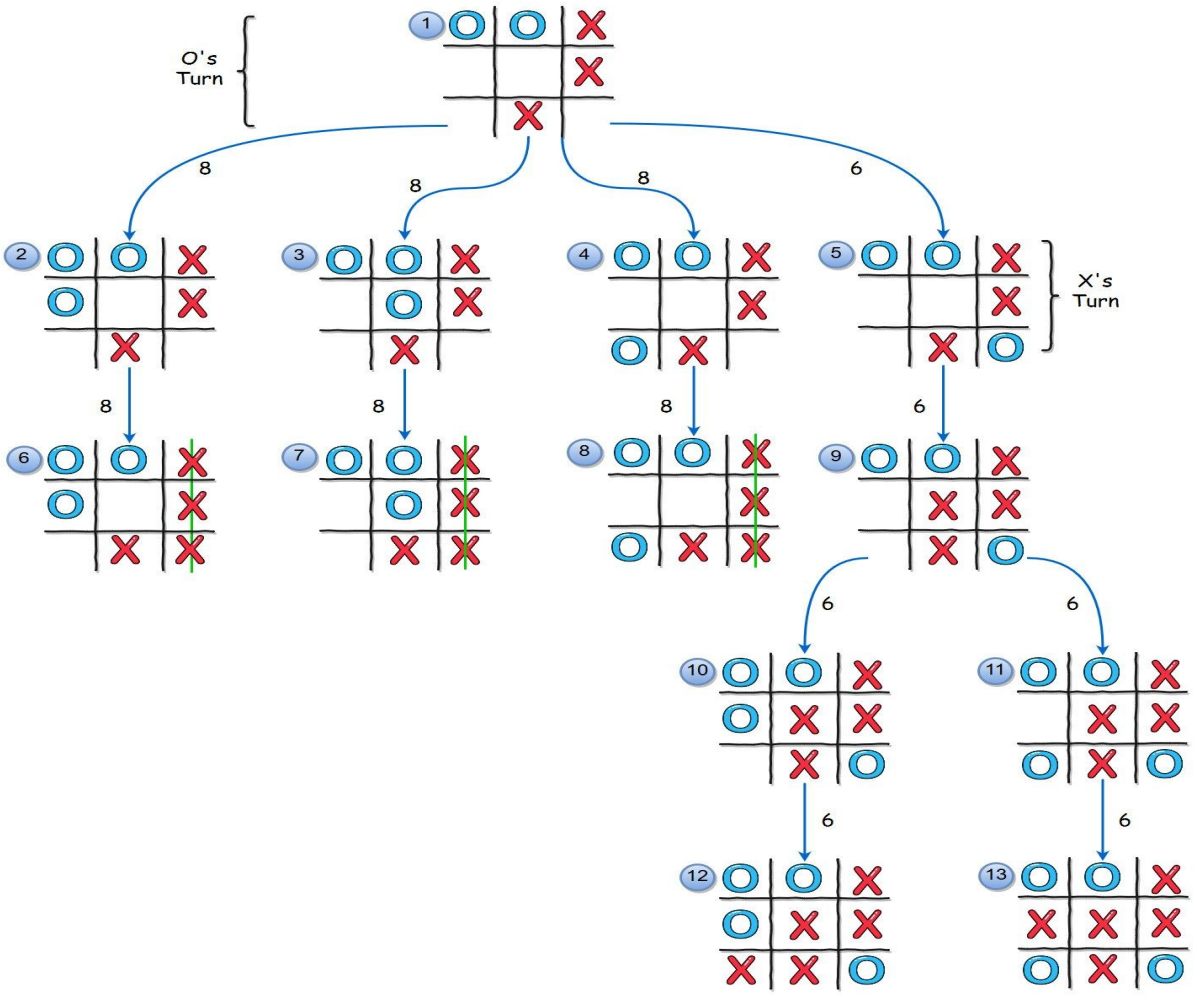
- **If X wins, the score he gets is 10 - Depth.**
- **If O wins, the score he gets is Depth - 10.**
- **If it is a draw, then the score remains unchanged.**

Given below is an example of a game tree with marks branches that end up getting pruned based on the specified conditions.



Thus based on the maximiser's turn , the minimiser plays the best possible move . This concept is used in developing a tic-tac-toe game which is unbeatable. The best that any real player can do is to draw the game if he plays optimally because for his every move , the minimiser looks ahead and plays optimally too.

The following is a state-space tree of the tic-tac-toe game where the root serves as a current state. The score of the end states or leaves are assigned based on their depth; which are static values.



Conclusion

The naive way of computing the most optimal or to say the best move for the current player based on the given state of the game and opponent's possible future move can lead to a combinatorial explosion of states for games like chess. The normal DFS(depth-first-search) ; exhaustive search leads to wastage of time and resources. The Static Evaluation for the leaf nodes which could be the possible final state is a hectic task too. Thus, alpha beta is an optimisation to the minimax algorithm which reduces the run time and number of static evaluations by pruning down parts of the tree and thereby reducing the Time complexity by a great factor. Other optimisations like Progressive deepening with state reordering can be applied to overcome the large number of static evaluations.