Minimax Algorithm and Alpha-Beta Pruning for Game Development Using Blockchain

Sumit S Shevtekar[1], Rutuja Kulkarni[2]

[1]ME Computer Engineering, Department of Computer Engineering, Pune Institute of Computer Technology
sshevtekar[at]pict.edu

[2]ME Computer Engineering, Department of Computer Engineering, Pune Institute of Computer Technology
rutujakulkarni[at]pict.edu

Abstract: ***Blockchain** has promise as an approach to developing various systems, games for a number of applications, domains. In Blockchain technology systems, data and authority can be distributed; all transactions are transparent and reliable. Some of the key advantages of Blockchain for cybersecurity applications are in conflict with privacy properties for game development. The platforms that are now accessible use centralized technologies. They lack transparency, scalability, and security. All those problems must be solved. Decentralization can help to tackle, secured these problems. Decentralization is provided through blockchain. In any search algorithm, searching best possible solution from the pool of every possibility known can lead to the construction of the whole state search space popularly called as minimax algorithm. The recursive backtracking algorithm known as Minimax is used to select the next action in a game of strategy for two players; the blockchain technology helps to improve security. The algorithm works well because it anticipates that your adversary will play well as well. However, as the tree's depth increases, we observe that minimax frequently investigates repetitive and unlikely situations. We'll also look at the Alpha-Beta Pruning approach, which serves as a minimax extension and stops us from looking at states that won't be chosen. We will also look into several traditional methods for solving two-player games, including adversarial search and other methods based on machine learning.*

Keywords: Blockchain, Decentralized Systems, Minimax algorithm, Alpha-beta pruning, Two-Player games, Blockchain, Smart-contracts, Ethereum, Game Theory, Game Tree Search Algorithms.

1. Introduction

The study of mathematical models of communication tactics between decision-makers is known as game theory. It is employed in numerous logics, computer science, and social science domains. The game theory, which now relies on a certain kind of behavioural interaction, has evolved into a catch-all phrase for the study of rational decision-making. Blockchain is a ready-made tool that developers use to build other platforms rather than building a blockchain from the ground up. There are countless platforms that can be utilized to implement blockchain-based projects. Most of them have similar functions to one another. Therefore, selecting a platform for developer's work is difficult [4][7].

Blockchain ensures decentralised and secure transactions by maintaining an immutable record of all transactions in a distributed ledger. A block containing the transactions is connected to the chain. We refer to the three technologies that make up blockchain technology as distributed ledger [7], consensus protocols, and cryptography [6]. Although these technologies are not new, blockchain is a novel technology because of the use of these technologies when combined. In digital partnerships, a Distributed Ledger is used to do away with the necessity for a reliable third party and lower the possibility of a single point of failure. In a peer-to-peer network like the one used by blockchain; each network node has a synced copy of the ledger.

Scholars have noted the importance of game theory as a tool for comprehending a variety of fields. Game theory has been applied to create theories of ethical or conventional behaviour, in addition to being used to describe, predict, and explain behaviour. Game theory-based principles are applicable to definition and modelling, business and economics, politics, project management, philosophy, computer science, and other fields. Recursive or backtracking algorithms include the mini-max algorithm. It is utilised in game theory and the decision-making process, as was previously mentioned. Assuming that the opponent is likewise playing really well, it provides the player a flawless move. Recursion is used by the Mini-Max algorithm to search across the game-tree. In 2-player games like chess, checkers, tic tac toe, go, and other 2-player games, the min-max algorithm is frequently utilised [1]. The minimax choice for the current state is computed using this algorithm. When playing this game with two players, each player plays the game so that they gain the most from it and the opponent player gains the least. They select this strategy so that they receive the greatest benefit and their adversary receives the least benefit.

When exploring the whole game tree, the Mini-Max algorithm uses a depth-first search method. The minimax method descends all the way to the tree's terminal node before recursively going back up the tree. Minimax frequently explores duplicate states and states that are unlikely to be picked by the players, however, as the depth of the tree rises. This is where we present the idea of an additional optimization method that helps to avoid this: pruning by alpha-beta.

A more advanced variant of the minimax method, alpha-beta pruning is actually used to improve the minimax algorithm. As previously established, the minimax search algorithm's backtracking of the game tree iteratively causes the number of states it must evaluate to exponentially increase as the depth of the tree increases. Pruning is an approach that allows us to determine the proper minimax result when compared to the original result without looking at each node of the game tree. It is known as alpha-beta pruning because it involves the two threshold parameters "alpha" and "beta" for future growth.

2. Literature Survey

With the use of the blockchain, a decentralized transaction ledger for generating, validating and transact with other nodes that are part of the same network achieved. The level of security required for financial transactions is additionally increased by various cryptographic hash algorithms of particular cryptocurrencies. Financial services, healthcare services, as well as business and industry, can all use the blockchain [4][7]. The following table compares the 2-player game theory approaches and algorithms to demonstrate the literature review:

No.	Algorithm	Version of	Method	Examples
1	Monte Carlo Tree search	Alpha-Beta Pruning	It is a simulation-based best first search algorithm that has been expanded to support pruning in the Alpha-Beta pattern [3].	zero-sum games tic-tac-toe checkers
2	Principle Variation Splitting (PVSplit)	Alpha-Beta Pruning	It is a parallel Alpha-Beta pruning algorithm that stipulates that before exploring more branches, one must first search the initial branch at a PV node.	zero-sum games tic-tac-toe checkers
3	Negascout	Minimax	Reducing Calculation, that is, we do not thoroughly explore each node by excluding options that both players ignore.	zero-sum games tic-tac-toe checkers

3. Methodology

1) Algorithms

In the context of a game, this section discusses the search algorithms Minimax, Alpha-Beta Pruning, and NegaScout. A game tree that includes all of the potential movements a player might make serves as the foundation for all algorithms.

a) *Minimax*

A prominent backtracking method in game theory is called Minimax. The Minimax algorithm iterates around the game tree to find the optimum move, returning it depending on the score at the leaf node. You may find a more thorough explanation and pseudocodes ahead. [1].

We have explored two-player zero-sum games where both players play so that their potential loss is minimised (penalty), while their opponent's punishment is maximised. Minimax is a backtracking method used in game theory for decision-making. It presumes that both players are performing at their highest level.

The terms "maximizer" and "minimizer" refer to two players. The former seeks the highest score possible, while the latter seeks the lowest score possible. As a result, when one person wins, it automatically means that the other player loses, and the term "minimax" was born.

Every zero-sum game has a value attached to it, and at any given time, if the maximizer is in the lead, this value will be in the positive range, and if the minimizer is ahead, it will be in the negative range. To further demonstrate the idea, consider a game where, in the event that player 2 likewise plays optimally and chooses a course of action that results in a maximum payout of -V, player 1 can obtain V as the highest prize. Numerous zero-sum two-player games, including tic-tac-toe, chess, checkers, and others, may be played using this method. Here, with the aid of a diagram, we'll look at a tic-tac-toe example.

There are two participants in the game of tic tac toe: X and O. In this case, X is the maximizer and O is the minimizer; each has an equal probability of winning, losing, or drawing the game. They will pick that move if it puts them in a position where either of them has a strong probability of winning. Otherwise, if no move results in a win for the current player, the player will attempt to make a move that will result in a draw. With the aid of the graphic below, where a game has already been played up to a certain point and X must now participate, let's better comprehend this.

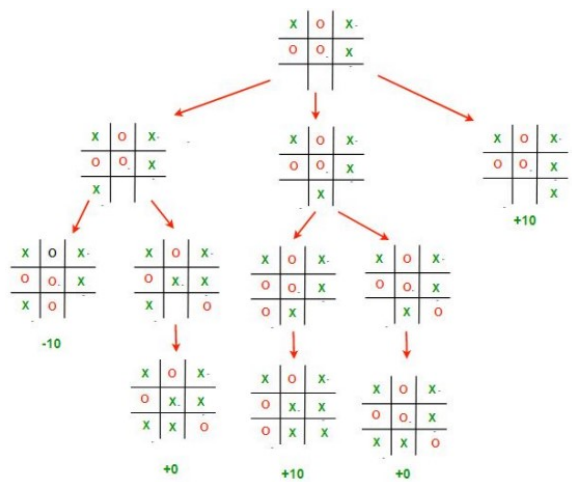

Figure 1: Minimax Algorithm using tic-tac-toe

As can be seen, in the first level of the game tree, player X has a choice of 3 alternative nodes. However, closer examination reveals that if player X plays the left node, [2,0], player O will have two alternatives for the following move, and as both players aim to maximise them. In order to win the game, player O will select the first node since it yields a score of -10, whereas the other move would result in a tie and a score of zero. Therefore, adopting this action for player X is not ideal since it will result in Player O receiving the most advantage and the least amount of punishment.

Player O will have to select one of two nodes at the following level if player X takes the middle move, [2,1], however as can be seen, neither of the two movements will result in player O winning because the first node leads to a value of +10 and the second one results in net zero. Player O will choose to increase their opponent's penalty and decrease their own as they play optimally as well, and as a result, they will select the second node to conclude the game in a tie. As a result, player X will have the highest chance of drawing by selecting the centre node.

However, if player X selects the correct node, [2, 2], then it will immediately result in player X's win with a value of +10, and as player X always plays optimally, they will select this move to maximise their points.

b) *Alpha-Beta Pruning*

To find the best move, the Minimax algorithm must run through the whole game tree. The Minimax method is improved by the AlphaBeta Pruning technique, which prunes the tree nodes that have little probability of delivering a better move and does not assess them [7]. When pruning results in bypassing an entire sub-branch of the game tree, it saves a lot of time. However, the Alpha-Beta Pruning algorithm's worst-case performance is equivalent to that of the Minimax.

We introduce a searching algorithm called alpha-beta pruning to improve upon the minimax algorithm since it reduces the number of nodes that are inspected in the game tree. As we know that minimax explores every state of the game tree and that it grows exponentially as the depth of the tree increases. Since a more advantageous course of action has already been discovered, it prunes the unnecessary branches. Alpha-beta pruning was given this name because it does this by adding two more parameters to the minimax algorithm, specifically alpha and beta.

When used on a typical minimax tree, it produces the identical move that minimax would, but it also verifies its veracity by removing branches that cannot potentially affect the final choice. Alpha-beta pruning has the advantage of allowing the search tree's branches to be removed. In this manner, a deeper search may be done while still limiting the search time to the subtree that is "more promising."

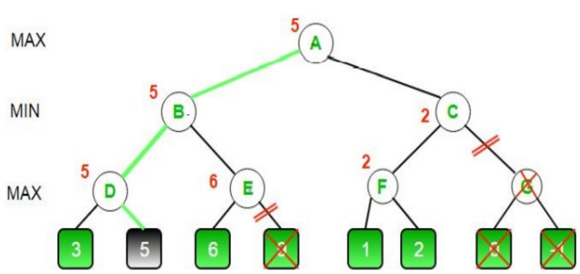

Figure 2: Alpha-beta illustration

Two parameters are maintained by alpha-beta pruning: alpha and beta. The maximum value that the maximizer can achieve at the specified level or above is alpha, whereas the greatest value that the minimizer can achieve at the specified level or above is beta.

Let's see how these 2 parameters are used in practise. As we know, the score rises positively for the maximizer and negatively for the minimizer, therefore in the beginning, alpha is negative infinity and beta is positive infinity, meaning that both players start with their lowest score. We shall now examine the circumstance in which a subtree may be removed. We can discard that subtree if ever a stage is reached when the highest score of the minimizer, or beta player, becomes smaller than the lowest score of the alpha player, or maximizer, as in a real-life scenario, that will never be taken into consideration.

c) *NegaScout*

The window (α, β), is where the Alpha-Beta Pruning algorithm starts the search, skipping any nodes that are outside of this window. By assuming that the first node found is the best node, the NegaScout algorithm seeks to raise the number of cut off nodes even further [5]. Using the exception of any nodes that violate the aforementioned presumption, the remaining nodes are only examined with a null window of (m, m+1) and a full window (α, β), research. The solver's performance is enhanced by the null window search's increased number of cut offs.

2) Pseudocodes

a) *Minimax*

As a result, we covered the minimax algorithm in the preceding part. Here, we'll try to implement the minimax pseudocode.

Algorithm 1: Minimax Algorithm

```
function minimax(node,depth, maximizingPlayer)
 if depth = 0 or node is a leaf node then
 |    return heuristic value of node
end
if maximizingPlayer then
 |    value = -∞
 |    while every child of node do
 |     |    value = max(value, minimax(child, depth-1, FALSE))
 |    end
      return value
end
else
 |    value = +∞
 |    while every child of node do
 |     |    value = min(value, minimax(child, depth-1, TRUE))
 |    end
 |    return value
end
```

Algorithm 2: Sequential Alpha Beta Pruning Algorithm

```
function ALPHA-BETA(node, depth, α, β,
maximizingPlayer)
```

```
if depth = 0 or node is a leaf node then
    return heuristic value of node
end
if maximizingPlayer then
    value = -∞
    while every child of node do
        value=max(value, ALPHA-BETA(child,depth-1,α,
        β, FALSE))
        α = max(α, value)
        if β ≤ α then
            break
        end
    end
    return value
end
else
    value = -∞
    while every child of node do
        value=max(value, ALPHA-BETA(child,depth-1,α,
        β, TRUE))
        β = min(β, value)
        if β ≤ α then
            break
        end
    end
    return value
end
```

Algorithm 3: Negascout Algorithm

```
function NegaScout(game Position,depth,alpha,beta)
 if depth = 0 or game is over
    return Eval(gamePosition)
end
n=beta
score=∞
Generate(gamePosition)
for i=1 to sizeof(moves) do
    Make(moves[i])
    if curr>score then
        cur = -NegaScout(gamePosition, depth-1, -n, -alpha)
    end
     if n=beta or d<=2 then
        score=cur
    end
    else
        score= -NegaScout (gamePosition, depth-1, -beta, -
        cur)
    end

    if (score > alpha)
        alpha = score
    end

    if (alpha >= beta)
        return alpha;
    end

    undo(moves[i]);

    n = alpha+1;

        return score;

end
```

4. Conclusion

Blockchain is future, security measure standard system and useful for implementation of secured systems, game theory, other secured communications and data transfer. Game theory is a plain, easily understood idea that is also evident in everyday life. In two-player games, the Minimax algorithm always selects the best move for the player, supposing that the other player would always play strategically as well. Minimax is so named because, as can be seen, it seeks to maximise player profit while minimising that of the opponent. Numerous 2-player zero-sum games, including tic-tac-toe, chess, checkers, and others, have been taken into consideration.

We attempt to optimise the method by utilising iterative-deepening with alpha-beta pruning such that a good move is returned even in the situation of an interruption because the minimax algorithm is depth-first and its states expand exponentially. Furthermore, simultaneous alpha-beta pruning, which aims to accelerate the present alpha-beta pruning by an average of 3.03, has recently come up for discussion. If not, it would be necessary to investigate each of the tree's exponential game states, which would be incredibly expensive. Consequently, alpha-beta pruning improves the minimax method by preventing state exploration.

References

[1] Pranav G., Satvik M., Neeta P. “Realization of Game Tree Search Algorithms on FPGA: A Comparative Study”, 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), 2019, pp. 1-3.

[2] Shubhendra P. S., M. Sridevi. “Comparative study of performance of parallel alpha Beta Pruning for different architectures” 2019 IEEE 9th International Conference on Advanced Computing (IACC), 2019, pp. 115-119.

[3] Maciej Świechowski, Konrad Godlewski, Bartosz Sawicki, Jacek Mańdziuk. “Monte Carlo Tree Search: a review of recent modifcations and applications” Artif Intell Rev (2022) 10462-022-10228.

[4] Sachchidanand Singh, Nirmala Singh, “Blockchain: Future of Financial and Cyber Security”, 978-1- 5090-5256-1/16/$31.00 c 2016 IEEE.

[5] S. Ariyurek, A. Betin-Can and E. Surer, "Enhancing the Monte Carlo Tree Search Algorithm for Video Game Testing," 2020 IEEE Conference on Games (CoG), 2020, pp. 25-32.

[6] Sachchidanand Singh, Nirmala Singh, “Blockchain: Future of Financial and Cyber Security”, 978-1- 5090-5256-1/16/$31.00 c 2016 IEEE.

[7] M. J. Hossain Faruk, H. Shahriar, M. Valero, S. Sneha, S. I. Ahamed and M. Rahman, quot;Towards Blockchain-Based Secure Data Management for Remote Patient Monitoring, quot; 2021 IEEE International Conference on Digital Health (ICDH), 2021, pp. 299-308, doi: 10.1109/ICDH52753.2021.00054.