The Use of Minimax Algorithm In Creating a Competitive Computer Opponent for Tic-Tac-Toe Game

Leen Sharab, Ameera Attiah, Sarah Alshumayri

1. **Abstract**

Tic-Tac-Toe has been widely researched as a basic model for assessing the performance of various AI systems. The Minimax algorithm is one of the most commonly used algorithms for producing a competitive computer opponent in Adversarial game. This paper aims to investigate the application of the Minimax algorithm in designing a computer opponent for the Tic-Tac-Toe game, aiming to maximize the computer's chances of winning. The research methodology involves a detailed explanation of the game logic and how Minimax is used to make the computer opponent moves in a Tic-Tac-Toe game, followed by an evaluation of the algorithm's success in producing a competitive computer opponent. The results of the study indicate that implementing the Minimax algorithm significantly improves the performance of the computer opponent in the Tic-Tac-Toe game, compared to using a random move generator.

**Keywords.** Minimax Algorithm, Tic-Tac-Toe, Artificial Intelligence, Computer Opponent

# **Introduction**

Since ancient times, games have been an essential part of human life, serving as a means of socialization and imaginative expression. Tic-tac-toe is a simple game where two players alternately mark a 3x3 grid. [1] Players can use the Minimax algorithm to increase their chances of winning while decreasing their opponent's chances. This algorithm evaluates a player's possible moves, assuming the opponent is also playing optimally. It selects the move that results in the best possible outcome for the player. Artificial intelligence can be used in software games to simulate human players, allowing game developers to create unique experiences and outcomes for each player. The Minimax algorithm is a helpful game theory and artificial intelligence tool for determining a player's best possible move. [2] Due to its widespread recognition and popularity, this study employs tic-tac-toe as a case study, eliminating the need for extensive rule explanations.

Additionally, the game presents a simpler alternative to other strategic board games, such as checkers and chess, which are more intricate. For instance, while tic-tac-toe has a maximum branching factor of 9, the average in chess is approximately 35. Furthermore, while the search tree for tic-tac-toe comprises a total of 623530 nodes, the number can easily reach 35^100 in the case of chess.

# **Literature Review**

Comparing the choice problem to basic search problems, Figueired et al. describes that the existence of an opponent makes the decision issue slightly more complex since the opponent introduces uncertainty because one never knows what he or she is going to do. They suggested that building a tree specifically for that problem is the greatest method for resolving search issues. The concept is implemented in this case using a general two-player system, requiring the specification of the initial state, operators, terminal set, and utility function. Figueiredo et al. discussed the simplicity of implementing the minimax approach in Lua. They mentioned that Lua's use of tables to construct matrices considerably speeds up the implementation process and assures that the final source code is easy to read and understand. [3] There is a need for greater investigation into applying the minimax algorithm in a broader range of Tic Tac Toe game forms, as well as the determination of success measures that are most important in those circumstances. The algorithm may need to be modified to accommodate larger grids and more participants.

Another study conducted by Lakhmani and Punjabi describes a Python system that employs the minimax algorithm, game theory, and graph theory to train AI agents to complete the Ultimate Tic-Tac-Toe game. Whereas the ideal scenario considers all possible outcomes, the minimax algorithm just considers one step to identify the optimum immediate approach. While Ultimate Tic-Tac-Toe is a short game, they concluded that it is exceedingly complicated to analyze, allowing an unbeatable algorithm to be developed using a minimal state space tree. To calculate the best move for the agent, they have used Recursive Weight Heuristics. The paper could benefit from a more detailed discussion of the limitations and future directions of the proposed algorithm. [4]

# **Methodology**

The ultimate tic-tac-toe game can be implemented using the minimax algorithm, which relies on a backtracking approach for decision-making and game theory. This algorithm determines the optimal move for a player by assigning scores to the game state at the end of the game. It assumes that both players are playing optimally, selecting the move that results in the minimum or maximum score depending on whose turn it is. To learn, the agent employs a heuristic function to identify the best move possible. After the game concludes, the updated method is propagated from the final move to the first move via backward chaining. The agent's performance is evaluated using both full-board and partial-board representations. [4]

A version of the Tic-Tac-Toe game was played using a 4x4 board, and the objective was to get three 'X' or 'O' symbols in a row to win. The function used to determine the game's outcome gave a value of 0 for a tie, 1 for a computer victory, and -1 for a computer loss. The algorithm was later modified to take advantage of the opponent's weaknesses and keep playing even if it seemed like the game was lost. The computer considered all possible moves and calculated the number of moves remaining until the end of the game, or the end-depth value. The Minimax algorithm was then used to choose the move that would allow the computer to continue playing longer, hoping the opponent would make a mistake. [5]

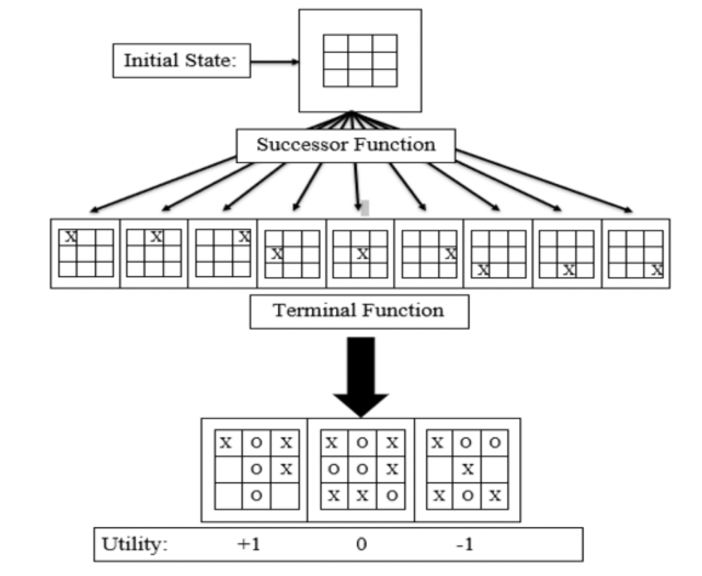
Google scholar and other research databases such as Scopus and ResearchGate were used to gather the sources of this paper. We ran a thorough search using keywords and phrases such as "Minimax Algorithm," "Tic-Tac-Toe Using Minimax," and "Artificial Intelligence". To narrow our search and acquire more relevant results, we employed search operators such as "AND", "OR", and "NOT".

# **Results**

In game theory, the minimax algorithm is a decision-making algorithm that is used to determine the best move in a two-player, zero-sum game. [2] It looks into how reinforcement learning can teach a computer program to play tic-tac-toe. To train the program, the researchers employ two reinforcement learning algorithms, Q-learning and SARSA. The results show that the program can learn to play tic-tac-toe very well, and the Q-learning algorithm outperforms SARSA. [6]

Lippert (2008) discusses using the Minimax algorithm to create a Tic-Tac-Toe game in Lua in his book Lua Programming Gems. According to the author, the algorithm can create a problematic game for human players while also simulating human-like thinking and decision-making in a computer opponent. [3]

Researchers assess the algorithm's performance using various metrics, such as the number of nodes searched and the time required to move. The results show that the algorithm works well in most cases, but it can be slow when the game tree is large. The authors propose several changes to the algorithm in order to improve its performance. [1]



**Figure 1.** A search tree for a Tic-Tac-Toe game [1]

The study also compares the minimax algorithm's performance to other game-playing algorithms, such as the alpha-beta pruning algorithm, and discovers that the minimax algorithm outperforms the alpha-beta pruning algorithm in terms of the number of nodes searched. Overall, the study concludes that the minimax algorithm is a practical algorithm for playing games like tic-tac-toe, but it may need to be modified to perform better in more complex games. [1]

# **Conclusion**

In conclusion, this paper investigates the use of the Minimax algorithm in creating a competitive computer opponent for the Tic-Tac-Toe game. The study found that implementing the Minimax algorithm significantly improves the performance of the computer opponent, as it maximizes the computer's chances of winning compared to using a random move generator. The research methodology involved explaining the game logic and how Minimax is used to make the computer opponent's moves in a Tic-Tac-Toe game, followed by evaluating the algorithm's success in producing a competitive computer opponent. The literature review revealed that the Minimax algorithm is widely used in game theory and artificial intelligence for determining a player's best possible move, and it is a helpful tool for resolving search issues. The methodology section described the implementation of the Minimax algorithm in the ultimate Tic-Tac-Toe game and a modified version of the 4x4 Tic-Tac-Toe game. The results of the study indicate that the Minimax algorithm can be applied to other Tic-Tac-Toe game forms, and it may need to be modified to accommodate larger grids and more participants. In conclusion, the Minimax algorithm can significantly enhance the performance of computer opponents in games, providing a unique experience and outcomes for each player.

1. **References**

|  |  |
| --- | --- |
| [1] | S. B, V. R. and . s. T S R, "Analysis of Minimax Algorithm Using Tic-Tac-Toe," 2020. [Online]. Available: https://www.researchgate.net/profile/B-Swaminathan-2/publication/346813363\_Analysis\_of\_Minimax\_Algorithm\_Using\_Tic-Tac-Toe/links/61376ffe2b40ec7d8bf01665/Analysis-of-Minimax-Algorithm-Using-Tic-Tac-Toe.pdf. |
| [2] | S. J. Russel and P. Norvig, "Artificial Intelligence A Modern Approach," 2010. [Online]. Available: https://web.cs.ucla.edu/~srinath/static/pdfs/AIMA.pdf. |
| [3] | R. M. Savelli and R. d. B. Seixas, "Tic-Tac-Toe and the Minimax Decision Algorithm," in *Lua Programming Gems*, 2008. |
| [4] | I. Lakhmani and V. Punjabi, "A Game Based Implementation of Minimax Algorithm Using AI Agents," 2020. [Online]. Available: https://ijsret.com/wp-content/uploads/2020/07/IJSRET\_V6\_issue4\_611.pdf. |
| [5] | K. K., "Is Minimax really an optimal strategy in games," 2010. [Online]. Available: https://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-article-BPB1-0047-0018. |
| [6] | P. Chen, J. Doan And E. Xu, "Ai Agents For Ultimate Tic-Tac-Toe," 2019. [Online]. Available: Https://Philhchen.Github.Io/Files/Tictactoe-Paper.Pdf. |