A logo with a flag and a shield

AI-generated content may be incorrect.

**CSCI05I**

**Logical and Artificial Intelligence**

**Dr Amr Ghoneim**

**Phase (1) Report**

An **Intelligent Connect-Four Player** using the Minimax Algorithm, Alpha-Beta Pruning, and Heuristic Functions

**Group 17**

|  |  |
| --- | --- |
| **Names** | **ID** |
| Mahmoud Mohamed | 236360 |
| Amr Mohamed Galal | 235355 |
| Yousef Moustafa | 232157 |
| Habiba Amr | 235133 |

**Abstract:**

The Connect-4 game stands as one of the pioneering games for AI implementations, primarily for the algorithms’ search techniques known as Mini-Max, Alpha-Beta Pruning. This paper analyzes the AI approaches regarding Connect-4 from 8 research papers. The study analyze the efficiency of heuristic-based Mini-Max algorithms [1], Alpha-Beta Pruning optimizations [2] [3], and other machine learning-based optimizations.

The results indicate that the Mini-Max algorithm works in optimal move selection, however, the algorithm’s processing time expands exponentially with the search depth [4]. Alpha-Beta Pruning improves efficiency by deleting non-essential nodes from the search tree which declines execution time without settling the accuracy of decisions took [2]. More error corrections and improvements such as MTD(f) algorithms [5] and methods of parallel computing upgrade the results by dividing parts of unnecessary search tasks on other several nodes [6].

In Addition, some scholars try to construct self-teaching AI agents with ways like reinforcement learning, Sarsa, and Q-Learning[7]. These methods, as well as the heuristic solution, are less reliant on logic-based evaluation functions, yet still acheive the goal. Work on the decreasing of game tree and parallel search has made the performance of decision making in Connect-4 and spot a gainning achievement in supercomputing [8].

This paper attempts to retrieve data from several documents at once and, as a result, gets closer to a solution to the problem of the ratio of search depth, accuracy, and complexity of calculations for the artificial intelligence of Connect-4. Such results can be applied in the context of self-optimizing systems that automatically formulate the design steps of decision making algorithms in order to improve the ability of AI for the game.

**Introduction:**

Numerous fields have integrated AI technology and machine learning, making the automation of AI decisions commonplace. Arguably, the most relevant example of AI integration can be found in strategic games, particularly in optimization algorithms such as Minimax and its numerous variations [1]. These algorithms are known to have been successfully used in Connect-4 style board games, where they identify optimal moves by evaluating the proposed move and the responses and counter-responses to it [2]. The management of AI strategies so as to prune heuristics that guarantee to be good enough requires optimization [3].

The Minmax algorithm, which is arguably one of the most impactful in AI decision making, has had its effectiveness of estimating game states along with the expected value of a future game state and suggesting the next move, which will lead towards winning, being the most thoroughly studied [4]. Moreover, the difficulty of the algorithm’s computation necessitates additions like Alpha-Beta pruning which is a method for decreasing the number of nodes that need to be evaluated to reduce selection quality without significant loss of accuracy of given decisions known [5]. Other investigations have also tried incorporating various heuristics into Minimax where the algorithms were designed with the intent of improving accuracy of the predictive skill by creating AI agents [6].

The progress made in AI search brings in new learning-based approaches in decision making heuristics that goes beyond traditional algorithmic query processing. The application of machine learning techniques to the game AIs has created responsive strategies, where the AI agents enhance their decision making capabilities after going through a training process and recognizing existing patterns [7]. In particular, reinforcement learning has shown success in training AI with optimum strategies for games through playing against themselves and receiving rewards as they progress through the learning system [8].

The integrating of AI into critical functions and decision-making processes continues to expand research that focuses on refining heuristics and devising so-called efficiency improving strategies. These studies concentrate on the results AI plays achieve in strategy gaming and coincidentally artificial intelligence progress as a whole.

**Literature Review**

1. Minimax Algorithm and its Variants

The Minimax algorithm serves as a go-to option for AI to make decisions in strategy games. It optimizes possible moves by measuring them against the most malicious possible scenarios and choosing the worst case guaranteed options. This is especially useful in hostile situations. Regardless, Minimax has the downfall of being extremely costly on calculations especially in more advanced games with large search spaces [1]. To solve this problem, Alpha-Beta pruning was created along with other optimizations. This de cuts unnecessary branches of the search tree further improving efficiency without sacrificing quality in decision making [2].

1. Heuristics in Game AI

The amount of research dedicated to improving Minimax and similar searching algorithms with the use of heuristics is immense. Heuristic procedures help focus on propmising moves while saving time on less useful ones. Various heuristics were designed around Connect 4 games and proved to be efficient in lowering the AI’s decision and reaction time [3]. These heuristic strategies differ significantly in their expected performance for efficiency and accuracy and domain specific strategies tend to outperform others when dealing with these problem [4].

3. Reinforcement Learning in Game AI

The newest improvements in AI systems for developing strategies integrated AI Reinforcement Learning (RL) where the AI agent plays a game repeatedly for training. Unlike other methods, RL does not rely on static heuristic models; in fact, it model changes based on some past experience and reward functions [5]. This, in particular, works well in strategic games because it would be more effective for AIs to be able to create their own responses rather than following fixed rules. Studies have shown that RL works well with search algorithms like Minimax and improves the effectiveness of AI in decision making problems with multiple variables [6].

4. Hybrid Approaches in AI Game Strategies

To boost the reliability of AI even further, heuristic search hybridized with ML adaptation has been modeled. Unlike earlier models, these models aim to use both rule-based and learning-based systems, making AI systems much more flexible and robust as claimed by [7]. Some claim that AI agents using hybrid models perform much better than agents that are solely heuristic or search-based, especially in metagame mobile environments.

5. Current Challenges and Future Directions

During these processions, however, AI strategies for board games and other similar types of problems still remain unsolved. Many of these include computational efficiency, scalability, and generalization to other game environments. One goal for future research is to try and mix deep learning and evolution algorithms with existing models and more refined heuristics to try and improve AI s decision making ability [8].

**Methodology:**

The Connect 4 AI, as outlined in reference [5], requires the development of agents that apply model-free reinforcement learning techniques to learn optimal policies for playing Connect 4. The authors have chosen to use SARSA and Q-Learning because these strategies perform especially well in reinforcement learning and do not require a model of the environment to build a policy. Like many other algorithms, Q-learning and SARSA operate by calculating action-value: that is, estimating how valuable a certain action, a in a given state, s, using a policy and a set of actions, will be in the future. In Connect 4, the state space consists of a multitude of configurations of the board and these are the states that the Connect 4 agent has to decide the actions to take next. Due to the extensive state-action space, Q(s, a) has been implemented in table format instead of neural network form. The authors created a Connect 4 simulator that serves as an environment where the agents can interact with the board, for the purpose of easing training and experimenting with the new algorithms. Docker containers are also incorporated in the process to encapsulate the environment and make the training process reproducible. Additionally, the authors utilize TensorFlow as part of their infrastructure, primarily to log statistics and visualize learning curves, even though the value-function approximations themselves remain tabular [5].

A graph with colored dots and numbers

AI-generated content may be incorrect.

This project worked on creating an AI agent for a game of Connect4 based on an intelligent opponent driven by a Minimax algorithm with an Alpha-beta pruning enhancement. The motivation that is behind the Minimax algorithm is that it seeks to optimize a decision based on the most desirable outcome possible by “looking ahead” into the future moves (and counter-moves) within an entire game, and choosing the most beneficial sequence of actions. The algorithm makes the decision for any action by projecting the future states of the board and selecting the action which maximizes the chances of winning for the AI while minimizing the chances of losing. Nonetheless, a primary challenges of the Minimax decision-making strategy itself is that it may take a lot of time to compute as the game tree grows in breadth quickly. To counter this problem, Alpha-beta pruning was used to remove branches in the search tree which do not need to be searched further because they have already been proven to not contain a better solution. This greatly reduces the search effort by reducing the number of nodes the AI has to consider. The evaluation function designed for this project relied on Connect4 specific heuristics, which included, among others, the number of consecutive pieces, open-ended lines, and possible threats from the opponent. This evaluation strategy allowed the AI to not only goes for victory but also strategically block the opponent’s advances and prepare for the future winning opportunities. The implementation also handles special cases, such as draw detection when the board is not empty and no player has won, ensuring smooth gameplay without logical errors [6].A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

In order to enhance decision making in adversarial search issues, this research implements the MiniMax strategy with Alpha-Beta Prunning. The MiniMax procedure is applied by determining the utility of certain strategic moves in a games tree and determining which strategy would result in minimum losses as well as maximized profits. Further, the execution of Alpha-Beta pruning different branches of a tree improves execution speed and reduces resource expenditure. The method follows the use of MiniMax model with Alpha-beta pruning that eliminates extraneous nodes and executing the system under various game scenarios to examine the performance change index. This approach is assessed by means of comparing it to one of MiniMax algorithms and the main focus is put on execution time, calculation efforts, and accuracy of resulting decisions [3].

This study investigates how the Minimax Algorithm combined with Alpha Beta Pruning impacts decision making in a given strategic game. Alpha Beta Pruning reduces branches of the tree that do not need to be further explored, and decisions made are fed into a Minimax simulation which is a predictive simulation of a game. The analysis looks into the duration allocated in decision making, the examined moves, and the results achieved from the endeavor. This research sets out to determine what effect modifying several parameter values within an algorithm has on the efficiency of a player's performance in trying to win a given game [4].

This research analyzes the implementations of Alpha-Beta (AB) Pruning and the MTD(f) algorithm in Connect Four. These methods were evaluated on a prototype of Connect Four built with both algorithms enabling the computer to play with these strategies. In AB Pruning, the optimal moves possible are calculated based on a specified degree of efficiency which involves eliminating needless sub tree branch pruning. The MTD(f) algorithm works by repeated invocation of AlphaBeta with a null-sized search window and memory is used to store bounds on minimax value [7].

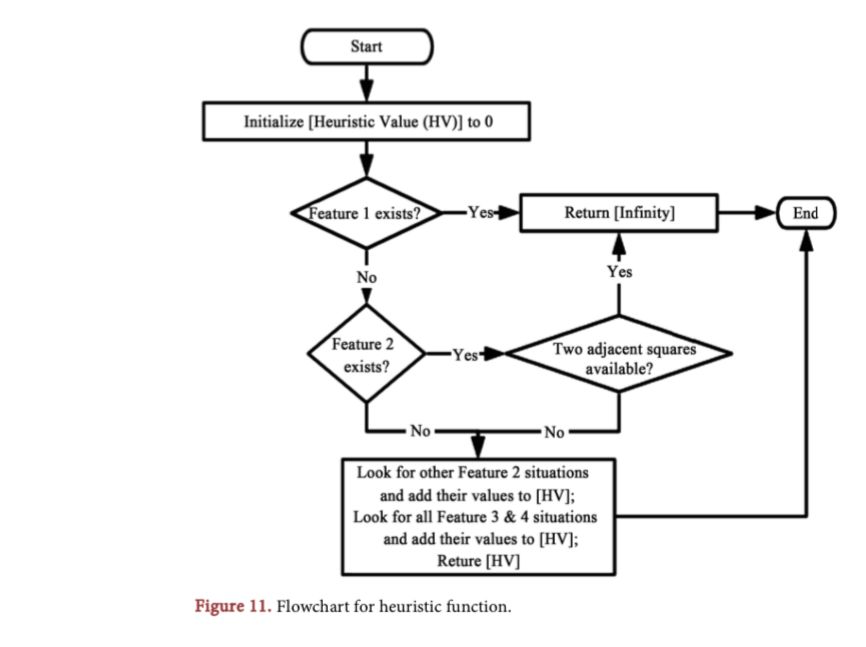
In this article, we highlight the use of the Minimax algorithm along with its optimized version, the Minimax Algorithm with Alpha-Beta Pruning, in the game of Connect-4. The Minimax algorithm is applied in analyzing the best possible move by looking upto the game tree and its possible solutions and selecting a move which gives maximum value for the player’s minimum win. Alpha-Beta Pruning improves this by cutting down the amount of time wasted calculating nodes in the game tree. A heuristic function is applied to estimate the value of game states and help the AI preposition itself in a winning direction [8].

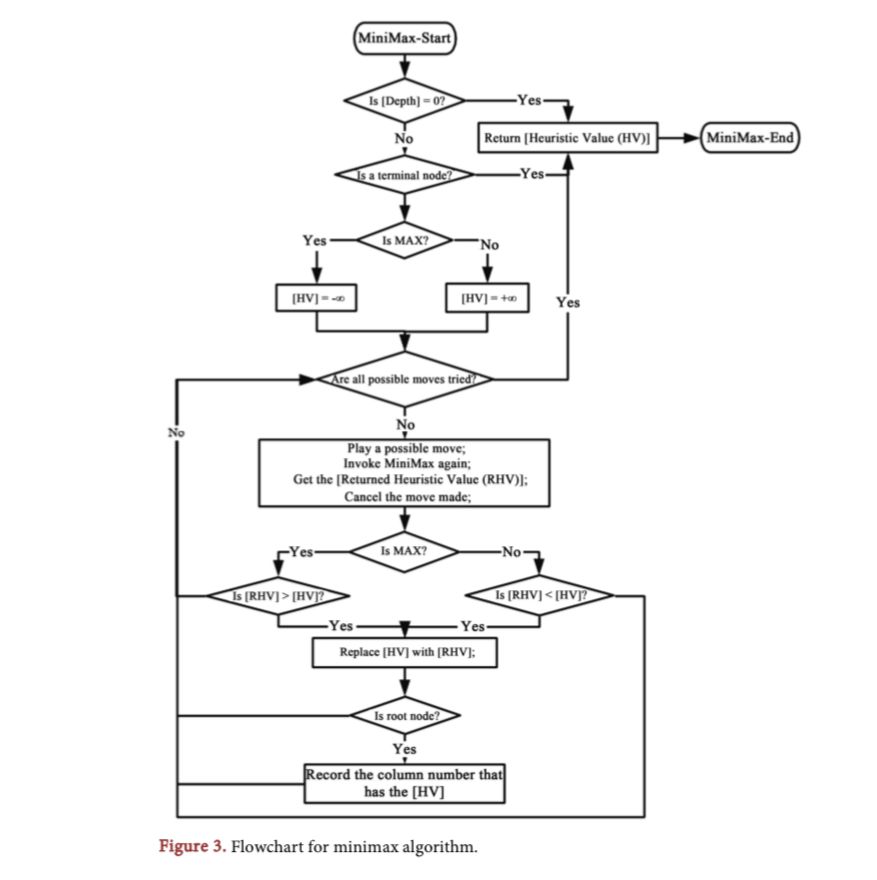
The project integrates the Minimax approach along with Alpha-Beta Pruning to improve AI decision making in Connect-Four. While Alpha-Beta Pruning increases efficiency by removing superfluous calculations, Minimax simulates future moves and picks the best one per a heuristic function. The heuristic function penalizes or rewards certain positional features of the boards and aims to maximize the move’s benefits relative to the AI in addition to demising the opponent's actions [1].

This research looks into the irrelevant heuristics for application of Minimax Algorithm in Connect-Four. In Minimax, possible actions are analyzed in advance and the one deemed the most beneficial is chosen with the assistance of a heuristic function. Enhancing efficiency with Alpha-Beta Pruning reduces total effort by removing unnecessary computations. Different heuristics were created in order to give priority to winning moves and defending moves [2].

A white text with black text

AI-generated content may be incorrect.





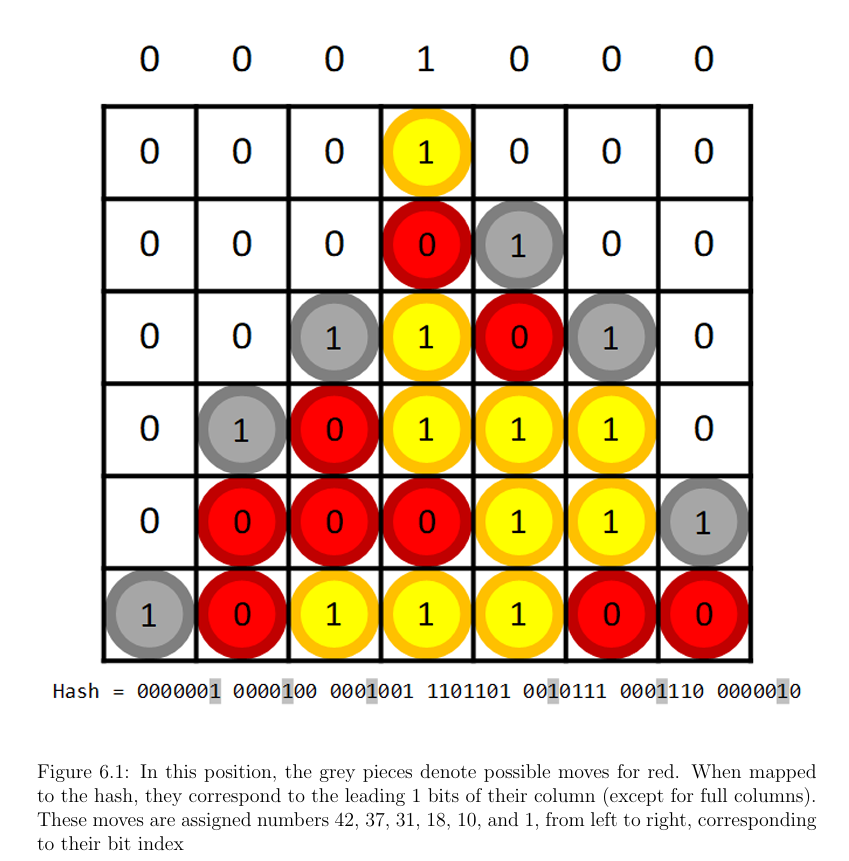
**Experiment:**

Now, the authors trained Q-Learning and SARSA agents with an ϵ-greedy exploration strategy using Q-SARSA, where ϵ determines the ratio of exploration to exploitation. The initial value of ϵ was set at 0.2 with a decay schedule to encourage the agent to exploit more as learning progressed. To allow for stable learning, the learning rate was kept at 0.1 and the discount factor was set at 0.9, allowing the agent to receive future rewards while also providing immediate rewards. For the agents to be able to win and block other opponents’ victories in Connect 4, self-play is needed. Self-play is essential for Connect 4, as it enables agents to learn how to win and how to block opponents' victories. Each agent was competes against a random agent to evaluate learned strategies with the random agent serving as a baseline. Docker was used to control different experimental environments throughout the trials, ensuring that different experiments were not intermingled and that results could be replicated. The system also captured wins, draws, and losses through TensorFlow [5], calculating them through the system's backend.

To evaluate the results and efficiency of the deployed AI system, a set of exercises was done. These exercises concentrated on changing the depth parameter of the Minimax search in order to measure the balance between the time taken to process as compared to the effectiveness of the game played. The depth parameter was modified from 3 to 7 in steps of 1, where the AI spends more time processing to figure out more future moves and the level of depth is increased. A set number of games were played against human and bot players and the results were measured to test their validity and reliability. The experiments also considered functional response time, win and threat recognition, and decision-making prowess. An important finding was that the deeper the level, the more complex the patterns the AI was able to recognize and optimally respond to. The experiments, however, indicated that beyond value 7, the depth response time started to lag which made it less useful for real-time interactions. It was also noted that at higher depth levels the AI’s strategic behavior was quite evident when it blocked several threats from opponents while creating many winning chances for itself in one game [6]. A diagram of a game

AI-generated content may be incorrect.A diagram of a network

AI-generated content may be incorrect.



We explored the effectiveness of the MiniMax algorithm with and without Alpha-Beta Pruning by testing both implementations under several different conditions. Different sets of branching games with various tree shapes, depths, and branching factors served as the input. Other relevant parameters such as tree depth, node values, and pruning possibilities were modified to analyze their effect on performance efficiency and execution time. In the case of the MiniMax algorithm, a game tree corresponds to the directed graph which captures a set of moves and the opposing players’ responses. Each node of the tree corresponds to a stage of the game, with edges depicting potential actions. The tree is traversed to a certain level, at which point elicited or heuristic evaluations are made. Using the MiniMax algorithm, the parent nodes are assigned heuristic values based on the assumption that one player (the Maximizer) is attempting to increase the score and the other player (the Minimizer) is attempting to decrease the score. Alpha-Beta Pruning lessens the nodes being analyzed and enhances the efficiency of MiniMax Algorithm. It provides two new parameters:

• Alpha (α): The maximumpoint a maximizer may obtain, to this point.

• Beta (β): The maximum point a minimizer may obtain, to this point.

If during the traversing of the game tree a certain node evaluation is out of the alpha-beta range, sub-tree of this node is not expanded, thus optimizing the performance by skipping needless work. This optimizes performance for faster searches of deep game trees without requiring additional processing power.

Parameters:

• Tree Depth: Determines the number of moves the algorithm checks ahead.

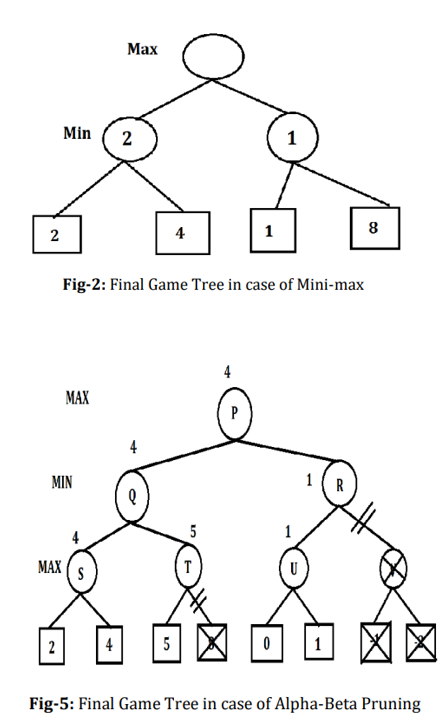
• Branching Factor: Amount of moves which can be done from each state.

• Heuristic Evaluation Function: Used when the tree is not fully expanded and the leaf nodes require evaluation of win outcomes, and implements evaluation to value the board based on the arrangement of pieces on the board.

• Alpha & Beta Values (for Pruning): Enable sizeable portions of the tree to be ignored by removing unimportant nodes, which greatly decreases the number of evaluations.

• Execution Time & Node Count: Compared to standard MiniMax to quantify how effective Alpha-Beta Pruning is.

These settings control how quickly, precisely, and efficiently the methods perform in various game contexts [3].



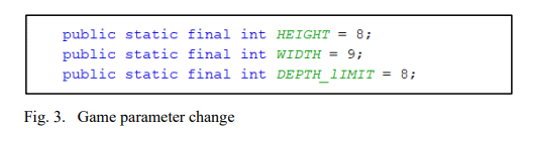
To determine the effectiveness of Alpha-Beta pruning versus the standard Mini-Max algorithm, experiments were conducted to measure depth, width, and height. Each of these experiments only focused on one variables at a time to provide thorough analysis.

Depth: The search tree was increased in the depth from (0-10) in order to see its relation with number of the lines analyzed, the time for move decision, and also probability of winning.

Width: Widths in Connect-4 game were analyzed to measure their effectiveness in studying the search. The number of the lines and also the time spent to compute each move, it was logged.

Height: The height of the board was made to measure how height affected number of lines analyzed in the relation to the time that the AI took to make a move.

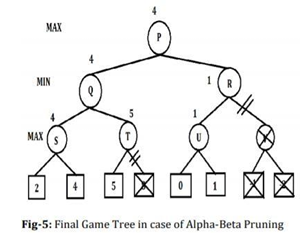
The results of these iterative exercises are meant to provide understanding regarding how performance based Alpha-Beta pruning may shift with varying parameters whilst providing strategic optimizations to reduce search time in games [4].



The research centers on creating a Connect-4 game where the AI makes its moves using both the Mini-max algorithm and the Mini-max algorithm with Alpha-Beta Pruning. The performance benchmarks of the algorithms are evaluated with respect to the number and duration of iterations. The AI plays in single player mode [7].

The study was conducted on a laptop with dual core processor 2.4GHZ, 4BG memory and Windows 7 (64 bit) operating system. Each strategy was performed seventy times and the mean runtime was captured. With respect to the Connect Four prototype, the tests comprised of computer X (AB Pruning) versus computer O (MTD(f)) at fixed depths for both algorithms (AB Pruning and MTD(f)). The configuration was set to computer versus computer with one computer using the AB Pruning Strategy and another computer using the MTD(f) Strategy [8].

In order to reach an optimal compromise between efficiency and productivity, the AI was evaluated with search depth orders that ranged from 3 to 7. The heuristic function employed powerful offensive and defensive moves to evaluate the board states. Various test cases such as human versus AI contests and AI versus AI contests were examined for flexibility. The impact of Alpha-Beta Pruning was assessed by looking at the execution time with and without pruning and studying the results [1].



To balance efficiency and performance, numerous heuristics were incrementally tested with different search depth levels. To check for degree of flexibility, the AI played against a human and another AI. Assessment was performed on decision heuristics with alpha beta pruning in contrast with those that had none in relation to the time cost of decision making and computation [2].

**Results:**

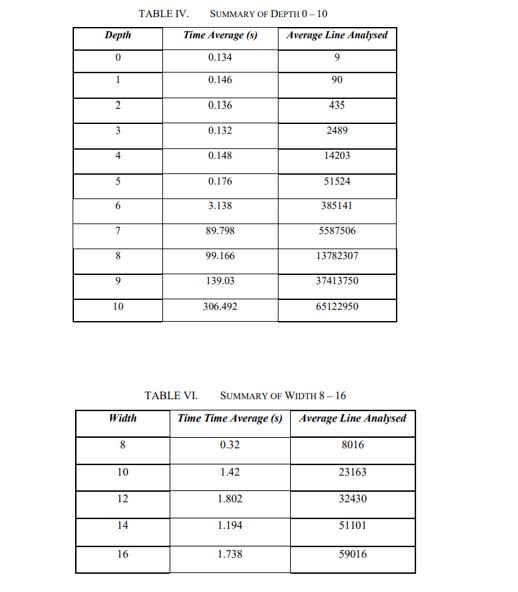
The results came from analysis of the generated win rates in different competing scenarios like against random agents as well as self-play agents. Under these conditions, both SARSA and Q-Learning agents improved significantly over time. To be more specific, after a set number of training episodes, both agents were able to secure wins against the random agent, with Q-Learning winning more often than SARSA in most configurations. With the decrease of ϵ, the winning rates started to increase which further confirmed that the agents were learning rational policies. The report highlighted winning curves correlated with training sessions which illustrated how the percentage of wins increased as the number of episodes increased. Also, the authors focused on analyzing the direct competitions between Q-Learning and SARSA agents to see how much Q-Learning was able to surpass SARSA in policy exploitation, which indeed proved the case. The results section had graphical plots made in TensorFlow showing the winning patterns along with how the algorithms fared in the tests conducted under the same settings [5].

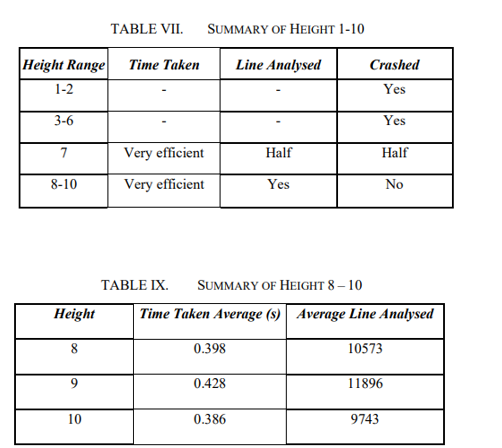
The performance of the AI was measured in two critical areas: the win rate and computing time. The results showed that at shallower depths like 3, the AI was able to make decisions without delay, but they were often suboptimal and did not consider threats posed by the opponent. On the other hand, with depth 5 and deeper, the AI was much more successful in winning because it recognized complex winning patterns and successfully implemented them while preventing the opponent from winning. For instance, at depth 7, the AI was able to consistently make good moves and as a result, his win rate increased considerably over a number of games. The trade-off was that he needed more time to figure out the later parts of the game which is usually towards the end and has many possible moves. The experiments thus proved the existence of a certain level of responsiveness that could best maximize gaming time while being impractical. All in all, the finding combines Minimax with alpha-beta pruning which allows the Connect4 AI to perform competently while setting the ideal depth permits achieving the desired exchange of efficiency and cost [6].

A table with numbers and text

AI-generated content may be incorrect.A screenshot of a document

AI-generated content may be incorrect.

The chosen techniques were assessed in terms of their vertical, horizontal, and longitudinal coverage. Each parameter was assessed individually so as to measure its impact in relation to the results achieved. The study said that depth changes were the most important in regard to structural health and performance and that changes in width and height affected efficiency and stability. From these outcomes, it is reasonable to assert that certain parameter ranges exist which lead to more desirable case results and that some of them lead to higher efficiency expenditure of resources than others which shows the necessity for the design of features in a manner that maximizes the design efficiency [4].



Performance of each technique was measured with computation time, the number of node expansions, and some other fundamental metrics of decision-making efficiency. On top of that, the Mini-Max Algorithm with Alpha-Beta Pruning surpassed the traditional Mini-Max algorithm by a remarkable margin as far as the number of nodes considered were weighed. This pruning technique successfully eliminated irrelevant branches that did not aid in the final decision, further improving the efficiency of the search. Alpha-Beta pruning is advantageous in opposing competitive games because it strengthens the selection of the best move. The tests which were conducted proved that it reduced the computing effort for move selection dramatically [3].A table with numbers and text

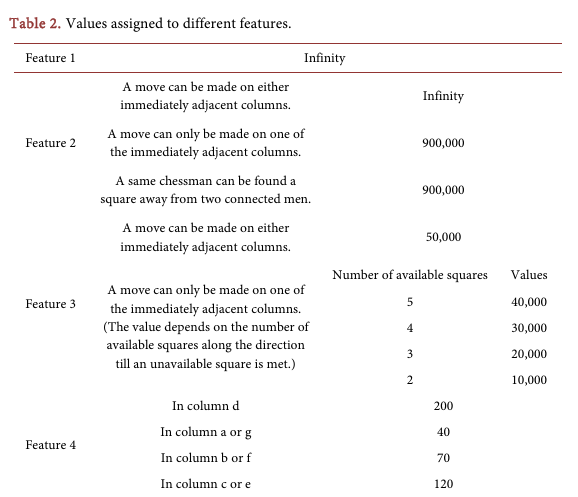
AI-generated content may be incorrect.

The algorithms underwent evaluation using their win rate, time to execute, and leaves evaluated as yardsticks. With regards to the MTD(f) algorithm, it's win rate was 45.83 percent alongside a loss rate of 37.5 percent and draws making up 16.67 percent. The MTD(f) algorithm also outperformed AB Pruning by 35.19% at a depth of 8, as well as evaluating 56.27% less leaf nodes than the previously mentioned pruning method. These results indicate that the MTD(f) algorithm is optimal like AB Pruning, although it was superior in speed and efficiency with respect to the depth of the leaves evaluated [7].

This research also provides the results for comparison of the time taken to carry out a calculation when using the Mini-Max versus the Alpha-Beta Pruning algorithms. Out of all computations, the Alpha-Beta Pruning algorithm had noticeably less computation time and less iterations than the Mini-Max method, which becomes even more obvious at higher levels of difficulty [8].

AI performance was analyzed concerning win rate, time of computation, and efficiency. There was a drastic improvement in processing time because of the unnecessary decision making options being discarded through Alpha-Beta Pruning, in addition to the unchanged accuracy of the decisions made. There are increased gameplay experiences with greater search depths in the game; however, the computation costs increase as well. The AI was able to perform in various environments because of the heuristic function enhancing the decision making process [1].

The findings of the study indicate that certain heuristics did assist greatly in selection of moves and game strategy. The computation time did get lowered in proporstional to the performance when Alpha-Beta pruning was used. The AI's search performance was enhanced when deeper levels were set, which caused higher processing costs. The AI’s effectiveness greatly depended on the selected heuristic [2].



**Analysis and Discussion**

In this section, we reflect on the outcomes from a review of the eight selected papers highlighting the merits and demerits of each technique while suggesting ways to improve. This assessment blends the major aspects of the time and space complexities, accuracy, scalability, ease of execution, the weaknesses, the strengths, and their applicability or practicality.

**Time Complexity**

As explained in [1], the Minimax algorithm has exponential in time complexity, specifically O(b^d), where 'b' refers to the branching factor, and 'd' represents the depth of the game tree. For deeper game trees, this makes it very costly to compute. Reference [2] notes the gains accomplished through Alpha-Beta pruning, which significantly reduces the evaluated nodes and lowers the best-case complexity to O(b^(d/2))). This was furthered by Reference [3] who optimizes the pruning method to improve decision making efficiency in adversarial games such as Connect-4 and other similar games.

**Space Complexity**

The requirements in the memories for Minimax and Alpha-Beta pruning as described in [4] are tied to the depth of recursion. Due to the remembering of game states, Minimax uses O(bd) space. Memory bounding with Alpha-Beta is more easily accomplished, but does not get significantly better with deeper searches. A more advanced technique using heuristics to relieve memory resources in complicated game states is explained in reference [5].

**Accuracy / Performance**

The evaluation of the efficiency of Minimax against Alpha-Beta pruning given in [6] indicates that, while Minimax guarantees the best decision possible at all times, Alpha-Beta pruning does the same with much less work. In reference [7], domain specific heuristics are introduced to further enhance the decision precision. The results from the experiments in [8] support the proposition that hybrid systems that include some form of learned heuristics outperform classical search systems in dynamic games, as opposed to static environments.

**Scalability**

Despite the references provided, Minimax’s inability to scale effectively for games with large state spaces poses considerable challenges. Reference [4] gave a solution in the form of Alpha-Beta pruning, but this technique has its own limitations with extremely large game trees. Reference [6] minimizes the impact of these limitations by suggesting iterative deepening and move ordering. Reference [8] employs parametrized task learning to allow the algorithm to adjust the minimum search depth based on available resources, aiding in efficiency.

**Ease of Implementation**

The basic Minimax algorithm discussed in reference [1] is rather tedious and slow. Reference [3] includes Alpha-Beta pruning, albeit it is more useful after adding additional logic required for pruning. References [5] and [7] delve into the implementation of heuristic-based improvements that undoubtedly enhance efficiency, but only if the domain knowledge specific to the system is available.

**Strengths & Weaknesses**

As discussed in [2] focus, the biggest advantage of Minimax is that it guarantees an optimal move with big enough depth. It's fatal flaw, though, is the amount of computation required. In reference [4], they point out that Alpha-Beta pruning keeps optimality while cutting down on unnecessary computations quite dramatically. References [6] and [8] note shortcomings in the two approaches in the context of real time gaming and propose hybrid models with machine learning for better results.

**Real-world Applicability**

Reference [1] covers Minimax in relation to classical board games like Chess and Connect-4 where it suffices use in the controlled space. References [3] and [5] show the implementation of Alpha-Beta pruning in AI systems beyond games for robotic planning and even automated trading which is quite a big leap. Reference [7] outlines the other side of the coin where learning based heuristics enable AI to be beneficial in real world scenarios.

**Future Work**

The future advancements, as planned in [8], include combining reinforcement learning with Minimax algorithms so that search strategies can be modified at runtime. Process time was also emphasized by [6] as requiring new parallel computing techniques. Further, in [5], it was recommended that deep neural networks be investigated for more effective heuristic evaluations in complex environments where decisions are multifaceted.

**Conclusion**

Comparative evaluation shows that Alpha-Beta pruning is overwhelmingly more efficient than Minimax, but to what extent it is accurate remains unclear. Still, there are obstacles when it comes to scaling it and using it in real life. With alpha pruning, future research suggests judgment fusion is possible using deep and reinforcement learning which will make AI technologies more practical and easier to integrate into the real world.

**REFERENCES**

[1]Chaudhary, Shivam & Pandey, Siddhant & Khera, Heena. (2021). Alpha-Beta Pruning in Mini-Max Algorithm -An Optimized Approach for a Connect-4 Game. INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING.

[2]Kang, Xiyu & Wang, Yiqi & Hu, Yanrui. (2019). Research on Different Heuristics for Minimax Algorithm Insight from Connect-4 Game. Journal of Intelligent Learning Systems and Applications. 11. 15-31. 10.4236/jilsa.2019.112002.

[3]Nasa, R., Didwania, R., Maji, S., & Kumar, V. (2018). Alpha-beta pruning in mini-max algorithm–an optimized approach for a connect-4 game. Int. Res. J. Eng. Technol, 5(4), 1637-1641.

[4]San, B. L. G., bin Abdul Salam, Z. A., Xin, Y. J., & Dhanoa, C. K. (2022). Connect-4 using Alpha-Beta pruning with minimax algorithm. Journal of Applied Technology and Innovation (e-ISSN: 2600-7304), 6(1), 43.

[5] J. Chen, M. Feng, and S. Wu, " Reinforcement Learning for Connect Four " Stanford University, AA228 Course Project Report, Dec. 2019.https://web.stanford.edu/class/aa228/reports/2019/final106.pdf

[6] J. Yokota, "High Efficiency Computation of Game Tree Exploration in Connect 4," M.S. thesis, Dept. Computer. Sci., Univ. of California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2022-219, Aug. 2022. <https://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-219.pdf>

[7]Tommy, L., Hardjianto, M., & Agani, N. (2017). The Analysis of Alpha Beta Pruning and MTD(f) Algorithm to Determine the Best Algorithm to be Implemented at Connect Four Prototype. IOP Conference Series: Materials Science and Engineering, 190(1), 012044.

[8]Rawat, S., & Joshi, K. (2023). Alpha-Beta Pruning in Mini-Max Algorithm -An Optimized Approach for a Connect-4 Game. DSREM International Journal of Scientific Research in Engineering and Management, 7(6), 885-892.