CS311 Final Project – Gomoku Solver

Jiayi Chen and Siyuan Niu

Background

Gomoku, or five in a row, is an abstract strategy board game where two players place a stone—either black or white—on a 15x15 board. The winner is the first to form an unbroken chain of five stones horizontally, vertically, or diagonally. Minimax algorithm is widely used in board games involving two-player competition such as Tic-Tac-Toe and Gomoku [1]. In 1992, Vardi [2] added expected values and utility into the minimax algorithm, making it possible to use heuristic functions to predict and calculate the possible scenarios. However, due to minimax's $O(b^d)$ time complexity, researchers have incorporated various methods to improve minimax's search performance in Gomoku, including alpha-beta pruning that eliminates unpromising nodes [4], Monte Carlo Search Trees that adds randomization [5], and various heuristic functions [4]-[6]. Researchers have also explored other machine learning algorithms to solve complex games like Gomoku, such as convolution neural network [3].

In this project, we intend to implement the minimax algorithm with alpha-beta pruning and heuristic functions to solve Gomoku. We want to explore how limiting irrelevant moves and different depth limits affect algorithm performance based on both search time and win rates.

Methods

Assuming that the opponent will always play optimally, minimax helps the agent to identify the next optimal move through a recursive tree that propagates minimax values back "upward". Given that a tree has exponential features, we restrict the branching factor by limiting the next possible moves to the adjacent cells of all current stones on the board, since nonadjacent moves are often considered as unpromising in a game [7].

Considering a potential tradeoff between depth limit and minimax's optimality, we evaluate the algorithm with different depth limits. When the minimax tree reaches a predetermined depth limit, an evaluation function is invoked (Eq. 1) to analyze each possible move based on the number of threat patterns – the number of 2, 3, or 4 connected stones on the board. A pattern can be either "open" or "half," where an open pattern is not blocked by an opponent stone on either side and a half pattern is blocked by one.

$$\begin{aligned}\boldsymbol{Eval(board)} = {} & \mathbf{10000} * (N_5^{Black} - N_5^{White}) \\ & + \mathbf{5000} * (N_{open4}^{Black} - N_{open4}^{White}) + \mathbf{2500} * (N_{half4}^{Black} - N_{half4}^{White}) \\ & + \mathbf{2000} * (N_{open3}^{Black} - N_{open3}^{White}) + \mathbf{1000} * (N_{half3}^{Black} - N_{half3}^{White}) \\ & + \mathbf{250} * (N_{open2}^{Black} - N_{open2}^{White}) + \mathbf{50}(N_{half2}^{Black} - N_{half2}^{White})\end{aligned}$$

Eq. 1: Adopted from [2] and tuned some parameter values; we want to emphasize the importance of connected open and half 4.

Results

We ~~first~~ validate the efficiency of the alpha-beta pruning in the minimax tree process. As Fig. 1 illustrates, alpha-beta pruning improves the average time used by minimax for computation, and the difference becomes more significant as the depth limit increases. This is because the size of a complete minimax tree is related to the value of the depth limit, so pruning parts of a larger tree will save more time compared to a smaller one.

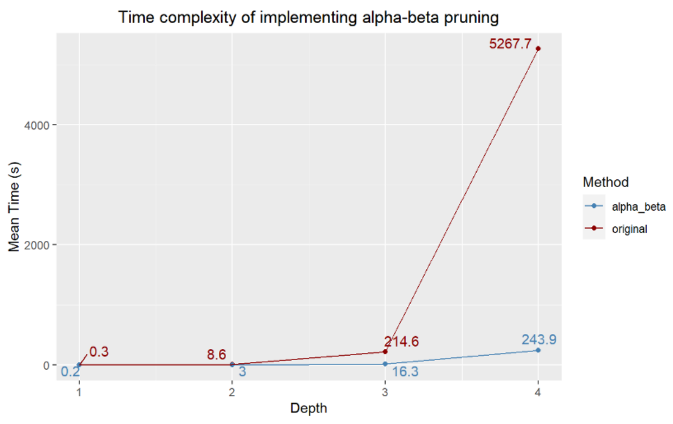

Fig. 1: Alpha-beta pruning reduces minimax's time complexity

Our algorithm is evaluated by comparing the win rates of nine different boards of three levels of difficulty: easy, medium, and hard. Difficulty is determined by the minimum number of moves it takes for black to win. For example, black may need only one correct move to win on an easy board, while five or more moves on a difficult one. Based on the controlled case where both black and white play randomly, we compare black's win rates across boards of different difficulties given a search depth limit of 0 or 1.

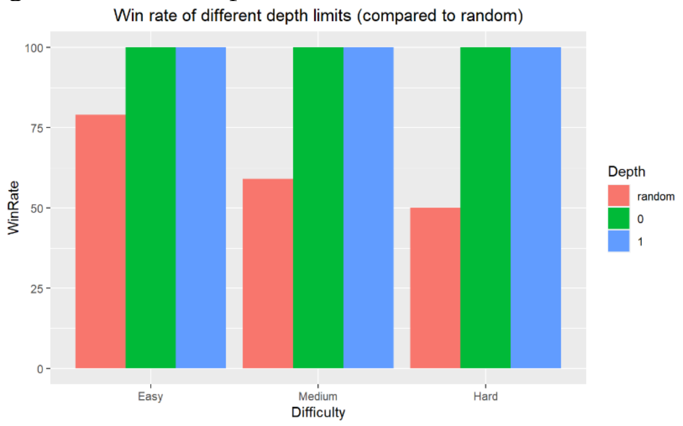

Fig. 2: As board difficulty increases, the win rate of black placing random moves drops from 76% to 50%, while black wins 100% against white placing random regardless of any search depth limits.

Conclusions

As previous research indicates, time complexity is a key constraint of minimax, and alpha-beta pruning optimizes the time complexity to $O(\sqrt{b^d})$ in the best scenario [4]. Our implementation demonstrates the effectiveness of alpha-beta pruning, especially given a larger depth limit (Fig. 1).

It is noteworthy that minimax beats an opponent with random moves in every game regardless of the depth limit, as shown in Fig. 2. This is due to the nature of the game: in the case of a depth limit of 0, for instance, black only uses the evaluation function once to play and does not predict any white's moves. Nevertheless, black makes progress towards connecting five stones, while white makes random moves and fails to block black's patterns.

The high win rate of black with random moves in easy boards indicates the effectiveness of limiting irrelevant moves. In easy boards, black typically only needs one more correct step to win, and that step is almost always one of the adjacent cells. Therefore, restricting black's moves to only those adjacent cells increases the chances of winning.

References

[1] Y. PIRILDAK, “Mastering Tic-Tac-Toe with Minimax Algorithm,” Medium, May 13, 2020. https://levelup.gitconnected.com/mastering-tic-tac-toe-with-minimax-algorithm-3394d65fa88f (accessed Dec. 10, 2022).
[2] A. Vardi, “New minimax algorithm,” J Optim Theory Appl, vol. 75, no. 3, pp. 613–634, Dec. 1992, doi: 10.1007/BF00940496.
[3] X. Fu, “GomokuPro: An Implementation of Enhanced Machine Learning Algorithm Utilizing Convolutional Neural Network in Gomoku Strategy and Predictions Model,” in 2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP), Apr. 2022, pp. 1671–1677. doi: 10.1109/ICSP54964.2022.9778476.
[4] E. Nygren, “Design Specifications for an Interactive Teaching Tool for Game AI using Gomoku”.
[5] Yu. (2019). AI Agent for Playing Gomoku. Retrieved December 9, 2022. https://stanford-cs221.github.io/autumn2019-extra/posters/14.pdf
[6] H. Liao, “New heuristic algorithm to improve the Minimax for Gomoku artificial intelligence”.
[7] “Minimax Improvements.” https://blog.theofekfoundation.org/artificial-intelligence/2015/12/18/minimax-improvements/ (accessed Dec. 17, 2022).