AI PROJECT

Aaro Karhu

University of Lorraine
Intelligence Artificiel

Contents

Introduction

Objective Of The Project

In the context of our Master's program in Computer Science AI course, we were tasked with developing the most efficient algorithm possible to simulate a player for the Awélé game. The bot is expected to compete in a tournament facing bots from other groups. There are specific constraints to consider in creating this bot, such as decision-making time and memory limitations.

Awéle

Awélé is a strategic board game where two players distribute seeds across 12 holes, aiming to capture more seeds than their opponent. Capturing occurs when seeds are distributed in a way that the last seed lands in a hole, making the total 2 or 3 seeds, which are then captured by the player. The game ends when one player can no longer make a move, usually because all seeds are concentrated in one player's holes or there aren't enough seeds to perform a capture, leading to the player with the most captured seeds being declared the winner.

MinMax Algorithm

Implementation and benefits

For my bot I chose to use the MinMax algorithm since the algorithm is well suited for turn based games. This algorithm is a robust solution for such games as it explores all possible moves from a given situation, aiming to maximize the player's gains while minimizing the opponent's gains. In the `BotBot` implementation, we iterate over all possible moves, applying a recursive depth-first search to evaluate the outcomes, adjusting for maximizing and minimizing strategies.

Limitations

Depth Dependency

The algorithm's capacity to forecast game outcomes hinges on the depth of its search tree. A deeper search allows for a more accurate assessment of a move's potential by examining future possibilities. However, deeper searches escalate computational demands, creating a trade-off between predictive accuracy and

practical execution time. Balancing this trade-off is crucial for maintaining `BotBot`'s competitiveness under the constraints of decision-making speed and computational resources.

Evaluation function value

The evaluation function's accuracy is vital for determining the desirability of a game state, directly impacting the MinMax algorithm's decisions. A well-designed evaluation function can somewhat mitigate the limitations imposed by depth dependency by providing insightful assessments of partially explored game trees. Thus, the function's ability to accurately value game states based on future potential and immediate gains is essential for strategic decision-making in Awélé.

AlphaBeta Pruning

Based on the MinMax algorithm, AlphaBeta pruning is implemented to reduce the number of explored nodes by cutting off branches that won't provide a better outcome than already discovered paths. It uses two values, α (alpha) and β (beta), where α is the minimum score that the maximizing player is assured of and β is the maximum score that the minimizing player is assured of. Once β becomes less than or equal to α, further exploration of the node is unnecessary. The `BotBot` class uses this technique to efficiently prune the search tree and optimize decision-making.

State Caching

State caching significantly boosts the `BotBot`'s efficiency by storing the outcomes of previously analyzed game states in a `HashMap`, reducing redundant evaluations. It leverages a unique hash for each state and includes depth information to ensure only relevant, accurate data is used for decision making. This approach not only speeds up the bot's response time but also conserves computational resources by avoiding the re-evaluation of game states, making the bot more competitive in tournaments.

Heuristics

For my bot I chose to use the MinMax algorithm since the algorithm is well suited for turn based games. This algorithm is a robust solution for such games as it explores all possible moves from a given situation,

aiming to maximize the player's gains while minimizing the opponent's gains. In the `BotBot` implementation, we iterate over all possible moves, applying a recursive depth-first search to evaluate the outcomes, adjusting for maximizing and minimizing strategies.

My Heuristic for KalaKala

Favorable Position for Capture

The heuristic prioritizes game states where the player's holes contain exactly 2 or 3 seeds. These configurations are deemed ripe for capturing, as they align with Awélé's capturing rules, allowing `KalaKala` to potentially gain more seeds on its next turn. Each instance where the player's holes meet this criterion adds a score increment of 0.6 to the layout score, reflecting the strategic advantage of such positions.

Minimizing Opponent's Capture Opportunities

Conversely, the evaluation function also accounts for the opponent's potential to capture seeds. It scrutinizes the opponent's holes for the same criteria—holes with 2 or 3 seeds—which would be advantageous for the opponent. To counteract this, the heuristic deducts 0.4 from the layout score for each of these conditions met, signaling the bot to avoid or alter these situations when possible.

Reflections

This simple heuristic worked best. I tried implementing much more complex heuristics as well, but when benchmarking them against this heuristic they all lost. By this simple method the bot can calculate the best next move to capture or to play defensively.

Improvements

The next improvement to `BotBot` would involve transitioning from the MinMax algorithm to the NegaMax algorithm. The NegaMax algorithm presents a promising evolution, simplifying the decision-making logic by utilizing the principle that a gain for one player is equivalent to a loss for the other. This symmetry allows the evaluation of game states through a single unified score, negated appropriately to reflect the perspective of the current player. By adopting NegaMax, we aim to streamline the recursive search process, potentially enhancing the clarity and efficiency of the bot's core algorithm.

The switch to NegaMax is expected to not only simplify the implementation but also maintain, if not improve, the strategic depth and computational efficiency of `BotBot`. This improvement aligns with our

objective of optimizing the algorithm within the constraints of decision-making time and memory usage. By refining the core algorithmic structure, `KalaKala` can become more adept at navigating the complexities of Awélé, leveraging a refined heuristic evaluation within a more streamlined algorithmic framework.

Conclusion

This AI project has been an invaluable part of my computer science journey, blending the complexities of algorithmic strategy with games. The endeavor to outsmart our peers pushed us to delve deep into optimizing algorithms within constraints, teaching us the delicate balance between theory and practical application. Moving forward, the potential shift to NegaMax opens exciting avenues for further enhancing our bot, promising simplicity and strategic depth. Overall, this experience has been enriching and interesting.

Sources

https://www.youtube.com/watch?v=M3LjiN7BSRs

https://www.lecomptoirdesjeux.com/l-awale.htm

https://www.joansala.com/auale/strategy/en/