Three-Player Connect 4 with Minimax and Alpha-Beta Pruning

Course: Artificial Intelligence
Institution: FAST NUCES - Karachi
Submitted by:

Muhammad Junaid Shaikh 22k4300

Izaan Mujeeb 22k4635

Hassan Shaikh 22k4440

Date: 10 May 2025

1. Executive Summary

Project Overview:
This project focuses on extending the classic two-player Connect 4 game into a competitive three-player version. The goal was to design a strategic, turn-based game environment and integrate artificial intelligence to simulate intelligent decision-making. We modified the traditional gameplay rules and implemented the Minimax algorithm with Alpha-Beta pruning to create AI agents capable of competing fairly against human players and each other.

2. Introduction

Background:
Connect 4 is a popular two-player board game where players alternately drop colored discs into a 6x7 grid, aiming to connect four discs horizontally, vertically, or diagonally. We selected this game due to its simple rules but rich strategic potential. To explore multi-agent AI interactions, we transformed it into a three-player game, which significantly increases complexity and introduces dynamic interactions.

Objectives of the Project:

- To design a functional three-player version of Connect 4.
- To build and implement an AI capable of playing competitively using Minimax and Alpha-Beta pruning.
- To evaluate the AI's performance in real-time game scenarios.

3. Game Description

Original Game Rules:
In the standard Connect 4 game, two players take turns dropping tokens into columns. Each token occupies the lowest available space in the column, and the first player to align four of their tokens in any direction wins.

Innovations and Modifications:

- The game now supports three players instead of two.
- Turn logic was modified to rotate between three players in a $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$ sequence.
- Token values are assigned as 1, 2, and 3 for identification.
- The win-checking algorithm was extended to handle three sets of tokens.
- A non-zero-sum dynamic is introduced, requiring more complex AI heuristics.

4. AI Approach and Methodology

AI Techniques Used:
We implemented the Minimax algorithm with Alpha-Beta pruning to minimize the number of game states explored while still selecting optimal moves in a three-agent environment.

Algorithm and Heuristic Design:

- A custom heuristic evaluation function was designed to score game states based on:
  - Immediate winning opportunities.
  - Blocking potential against two opponents.
  - Strategic control of center columns.
  - Line-building possibilities (e.g., open-ended 2s and 3s).
  - Recognition of future threats (e.g., forks).
- The game tree is generated up to a configurable depth (usually 3).
- Alpha-Beta pruning helps eliminate irrelevant branches, improving efficiency by roughly 50%.

AI Performance Evaluation:

- The AI made timely and strategic decisions, typically forcing humans into defensive plays.
- Alpha-Beta pruning reduced computation time and allowed for deeper game analysis.
- Heuristic tuning enabled the AI to recover from early-game disadvantages.
- The AI was tested in various randomized and structured board states for robustness.

5. Game Mechanics and Rules

Modified Game Rules:

- Players: 3 players using distinct tokens (1, 2, 3).
- Turns: Follow a fixed cycle (Player 1 → Player 2 → Player 3 → ...).
- Grid: The standard 6x7 grid is used.
- Victory: The first player to form a sequence of four tokens wins. Only one player can win a match.

Turn-based Mechanics:

- Each turn allows one player to drop a token into a column.
- The game continues until a player wins or the grid is filled.

Winning Conditions:

- A win is achieved when any player aligns four of their tokens vertically, horizontally, or diagonally.
- If the grid fills up with no winner, the game is declared a draw.

6. Implementation and Development

Development Process:

- The game was implemented in Python using a simple Command-Line Interface (CLI).
- Modules were created to separate the game logic, AI decision-making, and rule-checking.
- Initial development focused on board logic and rule modifications, followed by AI integration.

Programming Languages and Tools:

- **Programming Language:** Python
- **Libraries:** None (custom implementation)
- **Tools:** GitHub for version control

Challenges Encountered:

- Adapting Minimax for a non-zero-sum 3-player game posed logical challenges.
- Designing a heuristic that worked against two opponents simultaneously.
- Preventing biases in turn ordering and ensuring fairness.
- Handling edge cases such as potential alliances or trap setups between two players against one.

7. Team Contributions

- Junaid: Implemented Minimax algorithm and Alpha-Beta pruning logic.
- Izaan: Designed and coded the 3-player rule modifications and turn logic.
- Hassan: Worked on integrating the evaluation function and AI tuning.

8. Results and Discussion

AI Performance:

- The AI demonstrated competitive behavior, often reacting effectively to threats from both opponents.
- Decision-making was efficient, with average move times ranging between 0.4–1.0 seconds.
- The evaluation function provided consistent and balanced decision-making under dynamic conditions.
- Although the game was primarily human-vs-human, when AI was involved, it posed realistic and strategic challenges to human players.

9. References

- Russell, S. J., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach* (4th ed.).
- Wikipedia contributors. "Minimax." *Wikipedia, The Free Encyclopedia*.
- Python.org. Python Official Documentation.
- Online game tutorials and GitHub repositories for Connect 4 AI.