Project Proposal: Adversarial Game AI (Min-Max Algorithm and Alpha-Beta Pruning)

Introduction

Artificial Intelligence (AI) has revolutionized decision-making processes in various fields, and game-playing is one of its most fascinating applications. This project, Adversarial Game AI, aims to implement an intelligent agent capable of playing adversarial games, specifically focusing on Tic-Tac-Toe and Connect Four. The agent will employ the Min-Max algorithm for strategic move evaluation and incorporate Alpha-Beta Pruning to optimize computational efficiency. The project will serve as an educational exercise in AI principles while producing a practical application of game-playing AI.

Deliverables

1. A fully functional AI agent capable of playing Tic-Tac-Toe and Connect Four.
2. A report detailing the implementation, challenges, and outcomes of the project.
3. A graphical user interface (GUI) to allow users to play against the AI agent.

Objective

The primary goal of the project is to develop an AI agent capable of:

1. Competing effectively in adversarial games like Tic-Tac-Toe and Connect Four.
2. Using the Min-Max algorithm to evaluate and select the best possible moves.
3. Optimizing decision-making speed and reducing computational overhead through Alpha-Beta Pruning.

Significance

Adversarial games provide an excellent platform for exploring the fundamentals of AI. Games like Tic-Tac-Toe and Connect Four involve decision-making under uncertainty, strategic planning, and competition, making them ideal for understanding core AI techniques. This project will contribute to the field by implementing scalable AI techniques and demonstrating their effectiveness in solving real-world challenges in a controlled environment.

Scope

This project will involve:
- Developing a Python-based AI framework capable of playing Tic-Tac-Toe and Connect Four.
- Implementing the Min-Max algorithm to explore the game tree and evaluate potential moves.
- Incorporating Alpha-Beta Pruning to eliminate unnecessary branches in the game tree, thus optimizing the algorithm's performance.
- Testing the AI against both human players and other AI agents to evaluate its efficacy.

Methodology

1. Game Representation

- Tic-Tac-Toe: Represented as a 3x3 grid. The agent will play as either 'X' or 'O'.
- Connect Four: Represented as a 6x7 grid. The agent will aim to connect four discs in a row, column, or diagonal.

2. Min-Max Algorithm

- A decision-making algorithm used to evaluate possible moves by maximizing the AI agent's score while minimizing the opponent's advantage.
- The algorithm will traverse the entire game tree to identify the best possible move.

3. Alpha-Beta Pruning

- Optimization applied to the Min-Max algorithm to eliminate branches that cannot influence the outcome.
- This reduces the number of nodes evaluated, enhancing performance and enabling deeper lookahead in limited time.

4. Implementation Details

- Language: Python
- Libraries: NumPy for mathematical operations, Matplotlib for visualizing the game grid, and optional GUI libraries such as Tkinter for user interaction.
- Modular design to allow flexibility and scalability for other games.

5. Testing and Evaluation

- Test scenarios will include human-AI matches and AI-AI matches.
- Metrics: Win/loss rates, average decision time, and computational resources used.

Expected Challenges

- Balancing the depth of the search tree with computational limitations, particularly for Connect Four.
- Ensuring accurate implementation of Alpha-Beta Pruning to achieve optimal performance gains.
- Designing a user-friendly interface for players to interact with the AI.