Assignment 5: Implement Minimax Algorithm for Game Playing

Name: Vivek Mule
Roll: 381072
Div: A Batch 3
PRN: 22420145

Problem Statement

The goal of this assignment is to implement the Minimax algorithm to create an AI for a two-player game, such as Tic-Tac-Toe. This involves building a decision-making system that evaluates potential moves and selects the optimal one based on game theory.

Objectives

- Understand the principles of the Minimax algorithm.
- Implement the Minimax algorithm for strategic gameplay.

Theory

What is the Minimax Algorithm?

The Minimax algorithm is a recursive strategy used in two-player turn-based games to minimize the possible loss for a worst-case scenario. When playing optimally, it maximizes the minimum gain (or minimizes the maximum loss) for the player.

Methodology

1. **Generate a Game Tree**:
    - A game tree is constructed where each node represents a game state, and edges represent possible moves. The root node corresponds to the current game state.
2. **Evaluate Terminal Nodes**:
    - Terminal nodes are game states where the game has ended (win, lose, or draw). Each terminal node is assigned a score based on the outcome:
        - Win for the AI player: +1
        - Win for the opponent: -1
        - Draw: 0
3. **Backpropagate the Scores**:
    - Starting from the terminal nodes, backpropagate the scores up the tree. The AI player selects the move that maximizes its score (minimizes the opponent's score) at each level of the tree.

Working Principle / Algorithm

Here’s a simplified outline of the Minimax algorithm:

1. **Define the Game State**:

    - Represent the game board and the current player.
2. **Generate Possible Moves**:
    - For the current game state, generate all possible legal moves.
3. **Recursively Evaluate Moves**:
    - For each move:
        - Make the move and switch players.
        - If the new game state is a terminal state, return the score.
        - Otherwise, call the Minimax function recursively to evaluate the opponent's best response.
        - Undo the move to restore the game state for the next iteration.
4. **Choose the Optimal Move**:
    - Based on the scores returned from the recursive evaluations, choose the move with the highest score for the AI player.

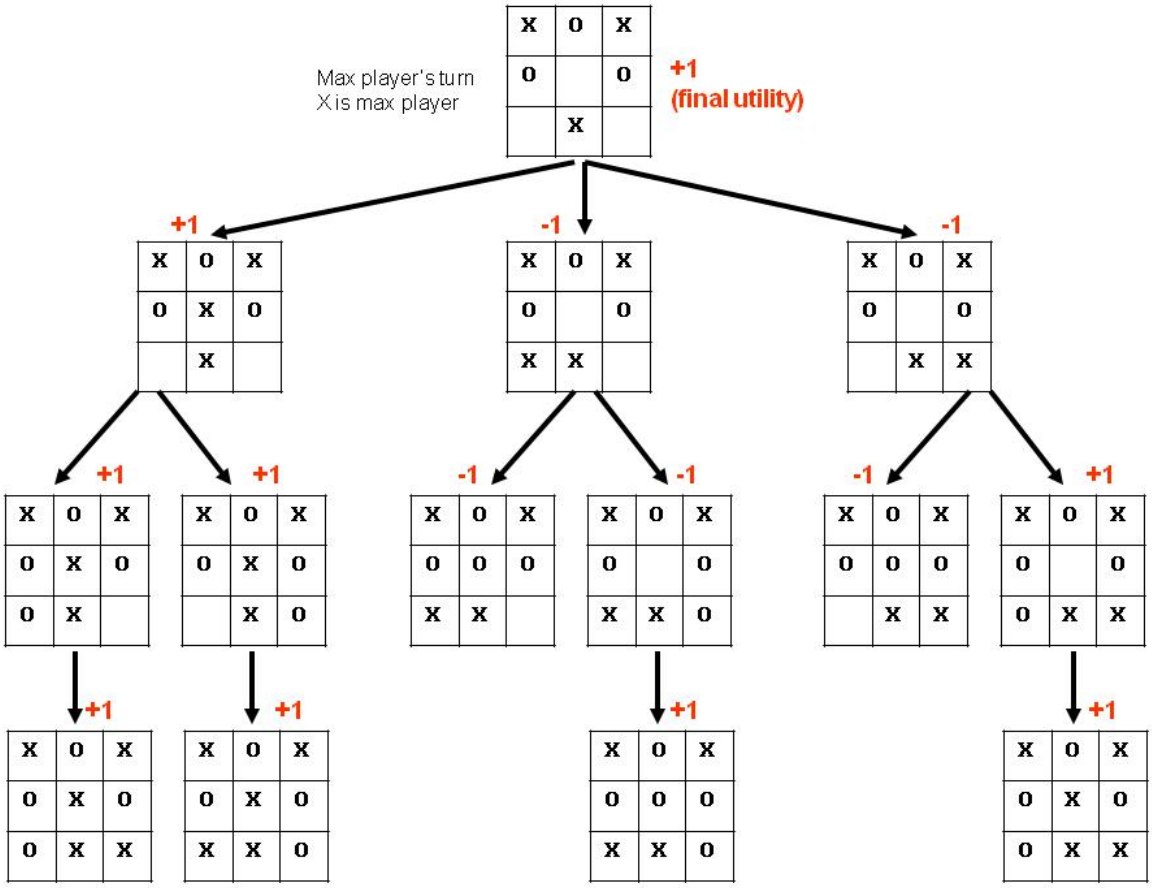
Advantages

- **Optimal Strategy**: The Minimax algorithm guarantees that the player makes the best possible move if both players play optimally.
- **Simple Implementation**: It is straightforward to implement for small game trees like Tic-Tac-Toe.

Disadvantages / Limitations

- **Computationally Intensive**: The algorithm can be slow for larger game trees, as the number of possible states increases exponentially with more complex games.
- **Memory Usage**: The size of the game tree can require significant memory, especially in games with larger boards.

Diagram



Conclusion

The Minimax algorithm provides a solid foundation for game-playing AI, ensuring optimal moves in strategic games. By systematically evaluating potential moves, the algorithm allows for the creation of intelligent opponents in two-player games.