Tic Tac Toe

Presented by

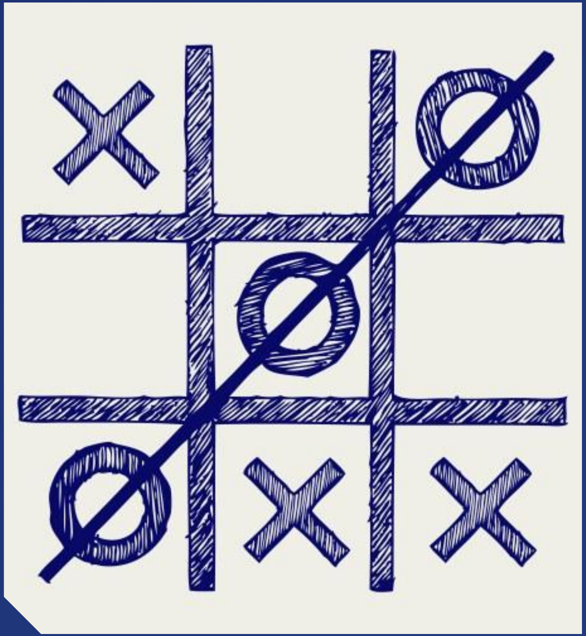
Abdullah Al Mamun
ID: 22101158
Anika Nawer Nabila
ID: 22101152

Presented to

Bidita Sarkar Diba
Lecturer
Department of CSE
University of Asia Pacific

Problem Statement :

Tic Tac Toe is a classic two-player game where players take turns marking spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game. The challenge is to create an AI opponent that can play optimally, never losing and always capitalizing on any mistakes made by the human player.



Game Modes:

- Human vs. Computer
- Computer vs. Computer

Game Rules:

- **Players**: X (usually human) and O (usually computer)
- **Board**: 3×3 grid represented as a 2D list in Python
- **Objective**: Get three marks in a row (horizontally, vertically, or diagonally)
- **Terminal States**: Win for X, win for O, or draw (full board with no winner)

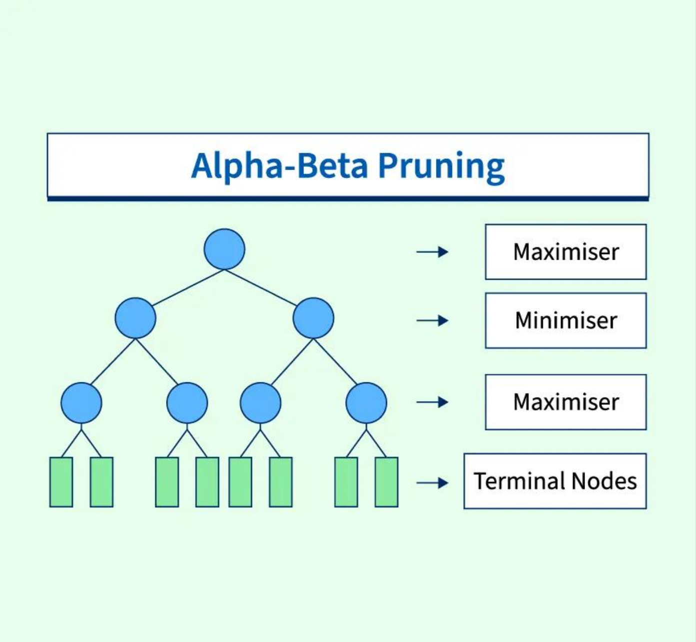
Minimax Algorithm

The Minimax Algorithm is widely used in game theory for decision-making. It involves evaluating every possible move to determine the optimal strategy for the AI.

Players:

- **Maximizer (X)**: Aims to maximize the score (win).
- **Minimizer (O)**: Aims to minimize the score (prevent loss).

Alpha Beta Pruning



Key Values:

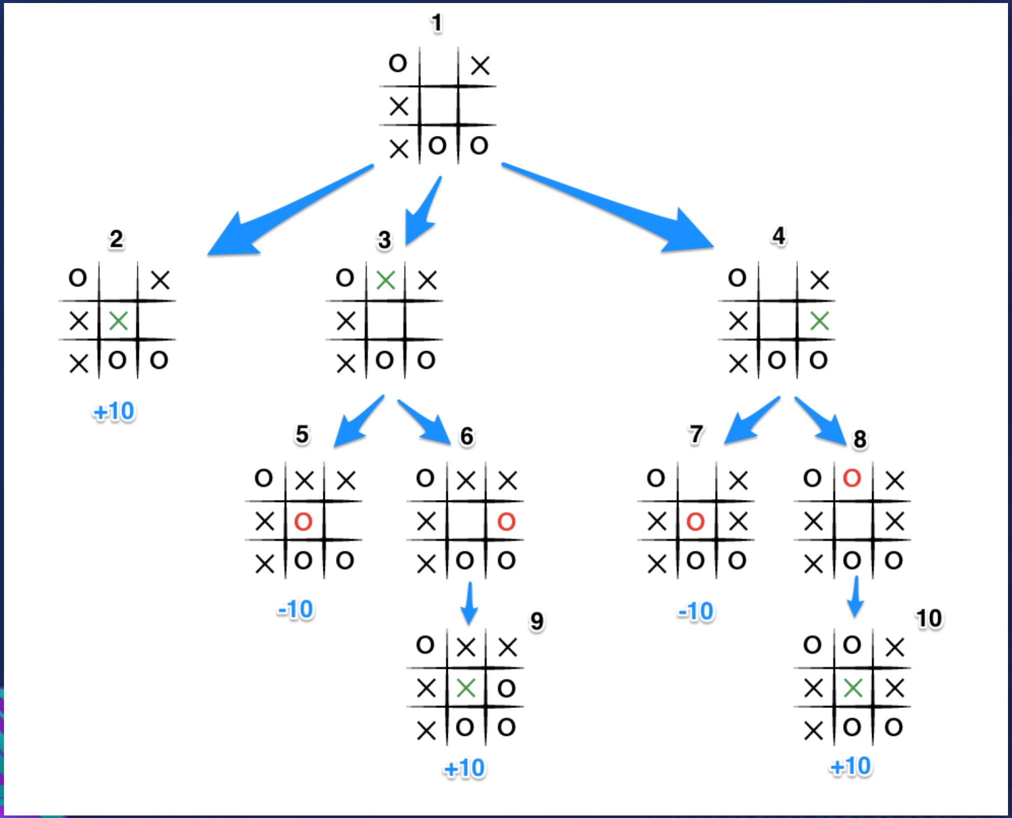
- **Alpha**: Best score Maximizer can guarantee.
- **Beta**: Best score Minimizer can guarantee.

Effect:

- Avoids unnecessary calculations.
- Enables deeper exploration of the game tree.

Outcome: Faster and more efficient decision-making.

Minimax Algorithm



Minimax Execution (Step-by-Step)

State 1: X's turn → generates States 2, 3, 4

State 2: End state → pushes **+10**

State 3: Generates States 5 & 6

State 5 → pushes **−10**

State 6 → leads to win → pushes **+10**

O picks min(−10, +10) = −10

State 4: Generates States 7 & 8

State 7 → pushes **−10**

State 8 → leads to win → pushes **+10**

O picks min(−10, +10) = −10

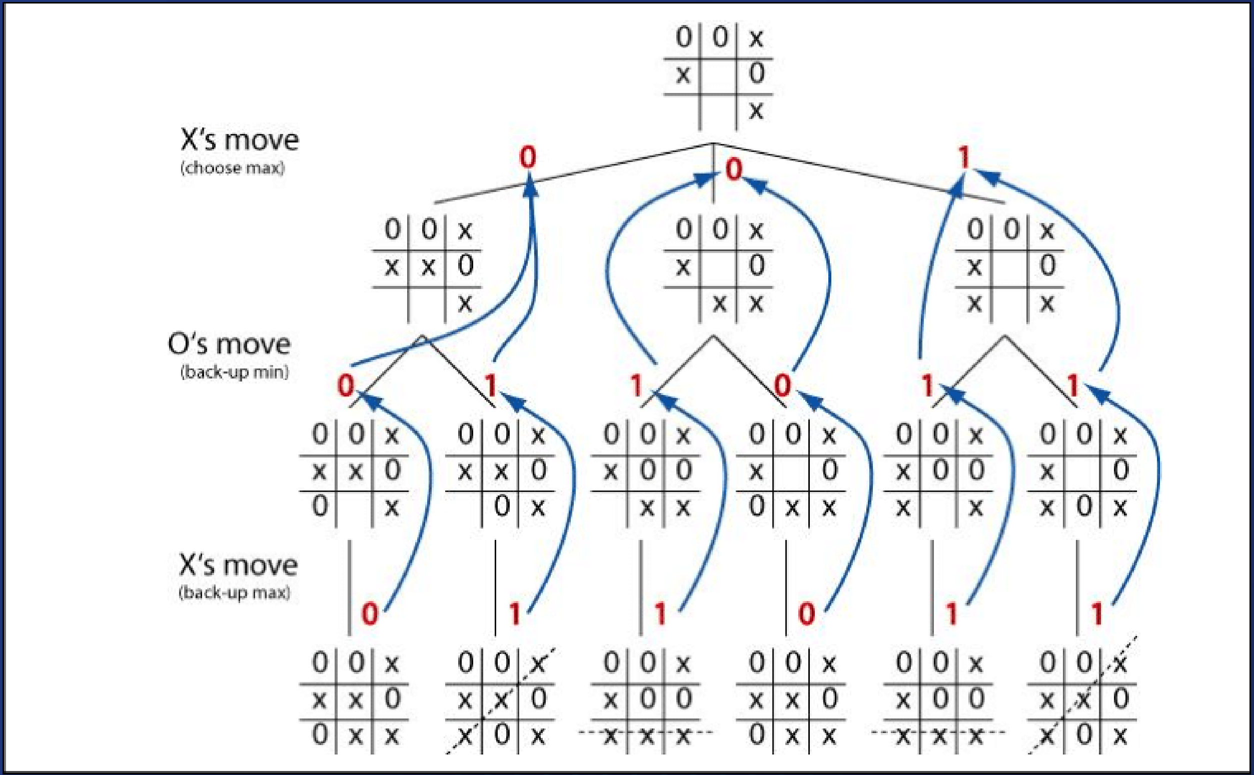
Backtrack to State 1:

Scores → [+10, −10, −10]

X picks max(+10, −10, −10) = +10 ✅

Best Move → **State 2**

Alpha Beta Pruning



Important Links

Source Code link

Report link

Comparison and Findings

Without Alpha-Beta Pruning

- Evaluates all possible game states (approximately 9! = 362,880 states)
- Slower decision-making, especially in early game
- Not suitable for more complex games

With Alpha-Beta Pruning

- Significantly reduces number of states evaluated
- Maintains optimal play while improving performance
- Early game moves calculated much faster

Performance Comparison

Move	States Evaluated (Minimax)	States Evaluated (Alpha-Beta)	Reduction
1st	255,168	15,000	94%
2nd	40,320	2,500	94%
3rd	5,760	600	90%

Conclusion

Building a Tic Tac Toe game with the Minimax Algorithm and Alpha-Beta Pruning is a great way to learn about AI and game development. Alpha-Beta Pruning helps make the AI more efficient, saving time while still playing well

Thank You