INSTITUTE OF INFORMATION TECHNOLOGY
UNIVERSITY OF DHAKA

AI Game : GoMoku Project Report

Course: CSE-604

Submitted To

Dr. Ahmedul Kabir
Professor
Institute of Information Technology(IIT)
University of Dhaka

Submitted By

AI-DLE

Mosamma Sultana Trina
BSSE Roll : 1313
Imamul Hossain Rafi
BSSE Roll : 1323
Fareya Azam
BSSE Roll : 1331

Date of Submission

12th November, 2024

Github Repo: https://github.com/Trina-SE/Game-Gomoku

Introduction

This project presents an AI agent designed to play Gomoku effectively against human or AI opponents, utilizing advanced game AI techniques. Core features include a Minimax-based search algorithm with Alpha-Beta pruning, a dynamic board evaluation function, and an early stopping mechanism for responsive gameplay. The AI aims to be both challenging and enjoyable, adapting its strategy based on the opponent's moves.

Implementation Overview

Game Tree Search with Minimax

In **minimax_alphabeta** function, the game tree is built dynamically through recursive calls, where each call represents a node in the game tree. Each level in this tree corresponds to a different "depth" of future moves and alternate turns for the maximizing (AI) and minimizing (opponent) players.

Alpha-Beta Pruning

Alpha and beta values are carried down through recursive calls, representing the best scores found so far for maximizing and minimizing players, respectively.

If a move at any point results in a score that definitively won't affect the outcome (e.g., if value >= beta for maximizing), the function stops evaluating further moves from this node. This is called pruning and helps skip branches that cannot influence the final decision.

Evaluation Function

The evaluation function assesses the "goodness" of each board state by scoring moves on multiple criteria:

- **Winning Potential**: Moves that create rows, columns, or diagonals of five stones receive the highest scores.
  -Consecutive 5 same pieces: 100000

  -Consecutive 4 same pieces(blocked): 5000*number of total consecutive 4 pieces
  -Consecutive 4 same pieces(free): 10000*number of total consecutive 4 pieces

  -Consecutive 3 same pieces(blocked): 500*number of total consecutive 3 pieces
  -Consecutive 3 same pieces(free): 1000*number of total consecutive 3 pieces

-Consecutive 2 same pieces(blocked): 50*number of total consecutive 2 pieces
-Consecutive 2 same pieces(free): 100*number of total consecutive 2 pieces

-Consecutive 1 same pieces: 1

- **Opponent Blocking**: Moves that prevent the opponent from completing a winning row are prioritized, especially if the opponent is close to winning.
- **Board Positioning**: Central positions have higher scores as they offer more strategic flexibility.
- **Pattern Recognition**: The AI scores open rows of three or four stones highly, prioritizing these patterns to enhance offensive and defensive setups.

Early Stopping

An early stopping feature is integrated to balance move quality with responsiveness. This feature allows the AI to return the best move found within a set time limit, creating a responsive and real-time experience. By setting time constraints, the AI can maintain performance during gameplay without sacrificing move quality.

Generate Moves Function

The AI focuses on evaluating moves close to occupied cells, concentrating on relevant board areas to reduce unnecessary calculations. The **generate_moves** method generates a list of possible moves by scanning the board for cells that meet certain criteria. The goal is to add moves to the list if a cell is adjacent to any occupied cells.

Evaluation Criteria and Testing

The AI was tested across various game scenarios to evaluate its performance based on the following:

- **Accuracy**: The AI's ability to identify winning moves or critical blocks in real time.
- **Efficiency**: The speed of decision-making, particularly with Alpha-Beta pruning.
- **Strategic Depth**: The AI's capacity to prioritize moves that provide long-term benefits and adapt to opponent strategies.

Challenges and Optimization

One major challenge was balancing search depth with decision speed, as deeper searches improved accuracy but increased computation time. Alpha-Beta pruning partially addressed this issue, but further optimization was needed to find the ideal balance. Additionally, tuning the evaluation function required adjusting scoring weights to ensure a balanced approach between offensive and defensive strategies, avoiding any bias that could compromise the AI's effectiveness.

Conclusion

This Gomoku AI combines intelligent search algorithms, a well-designed evaluation function, and responsive gameplay features to deliver a challenging experience for players. By meeting project requirements, it demonstrates a balanced approach to strategy and speed, making it both competitive and enjoyable. Future improvements could involve machine learning enhancements for more adaptive and nuanced gameplay.

References:

- *https://medium.com/@LukeASalamone/creating-an-ai-for-gomoku-28a4c84c7a52*
- *https://github.com/hesic73/gomoku_rl*
- *https://en.wikipedia.org/wiki/Gomoku*
- *https://github.com/canberkakcali/gomoku-ai-minimax*
- *https://github.com/hikalium/gomoku/blob/master/gomoku.c*
- *https://github.com/malikusha/Gomoku*