Chess Game

Alessandro Pio n.294417

October 29, 2023

1 Introduction

The code for the chess game utilizes the "Chess" library to create a fully functional chess game. This library simplifies the development of chess games, providing the ability to create a chessboard, track its status, and execute moves based on the player's turn. The code implements three different heuristics to enhance gameplay, each with its unique approach to evaluating the chessboard.

1.1 Heuristics Implemented

Three distinct heuristics have been integrated into the game, each influencing the agent's decision-making. **NB** In the context of heuristics, a move that leads to checkmate or a check on the opponent's king is considered particularly significant and is factored into the heuristics calculations.

1. **Material Heuristic**: This heuristic evaluates the chessboard by considering the sum of the pieces' values. It calculates the difference between the player's pieces and the opponent's pieces, where piece values are assigned as follows: (p) = 1, (b) = 3, (n) = 0 (r) = 5, (q) = 9, (k) = 3.

2. **Greater Number of Pieces Heuristic**: This heuristic focuses on the number of pieces possessed by the player whose turn it is. It calculates the difference between the player's pieces and the opponent's pieces.

3. **Most Move Heuristic**: This heuristic prioritizes moves that result in the maximum number of available moves in the subsequent turn. It aims to provide strategic advantages.

1.2 Minimax Algorithm

The code employs the Minimax algorithm to evaluate board states and determine the best move for an agent. This algorithm is used to make strategic decisions during gameplay, considering the opponent's responses.

1.2.1 Algorithm Overview

The Minimax algorithm is a decision-making strategy used in two-player games. It seeks to minimize the possible loss for a worst-case scenario while maximizing the potential gain. In the context of the chess game, Minimax aims to identify the move that maximizes the agent's advantage while considering the opponent's best responses.

1.2.2 Algorithm Implementation

The Minimax algorithm is applied with a specified search depth, allowing the agent to explore potential moves up to a certain level. The algorithm recursively evaluates the chessboard state, considering both maximizing and minimizing players.

1.3 Breaking Loop

In cases where heuristics yield equal global minimum costs, measures have been taken to prevent the creation of cycles. A random move is chosen from the set of global minima to avoid repetitive patterns. This strategy is particularly relevant at the start of the game, where heuristics often produce identical values, ensuring that a random move is executed.

2 Final Analysis

In the series of chess game simulations using various heuristics and agents, several observations can be made.

2.1 Combining "Number of Pieces" and "Material" Heuristics

- These games often revolve around material advantage, with an emphasis on capturing and maintaining pieces.
- Matches can end with one side having a substantial material advantage, potentially leading to checkmate.
- Draws are possible when both agents are cautious about material loss.

2.2 Using the "Material" Heuristic

- These games strongly emphasize material advantage and may lead to checkmates when one side gains a significant material lead.
- Draws can transpire if both agents are careful about material preservation.

2.3 Using the "Number of Moves" Heuristic

- The games tend to be dynamic, with an emphasis on mobility and strategic positioning.
- The number of moves in each game is relatively high, as both agents seek opportunities for maneuvering.
- Checkmates can occur, but draws are frequent due to the shared focus on mobility.

In summary, the choice of heuristic plays a vital role in shaping the gameplay and outcomes of chess matches. However, it is challenging to conclusively determine which heuristic is superior, as the games typically result in draws or occasional victories for one of the agents.