COMS4033A - Artificial Intelligence Assignment

School of Computer Science & Applied Mathematics
University of the Witwatersrand

Lisa Godwin - 2437980
Nihal Ranchod - 2427378

May 15, 2024

Table of Contents

All python scripts created for this assignment can be found at our GitHub Repository

1 Baseline Implementation

The baseline bot (RandomSensing) was constructed according to the guidelines outlined in James, April 2024. The results from the round-robin tournament against the RandomBot and the TroutBot are seen in Table 1. Each bot competed against every other bot twice, alternating between playing as black and white in each match.

Bot Result	**Score**	**Wins**	**Losses**	**Draws**	**Win %**
RandomBot	2.5	2	1	1	62.50
TroutBot	4.0	4	0	0	100.00
RandomSensing	0.5	0	3	1	0.00

Table 1: Round-Robin Results

2 Improvements

2.1 Improved Move Selection Strategy

The ImprovedAgent bot now utilises a set of predefined opening moves that significantly enhance the bot's strategic approach by ensuring a consistent and solid start to the game. The strategic moves are defined based on the bot's colour. If the bot is playing as white, it uses the `white_move` list, and if it's playing as black, it uses the `black_move` list.

Predefined opening moves for white: **b1c3** (Knight to c3) → **c3b5** (Knight to b5) → **b5d6** (Knight to d6) → **d6e8** (Knight to e8)

Predefined opening moves for black: **b8c6** (Knight to c6) → **c6b4** (Knight to b4) → **b4d3** (Knight to d3) → **d3e1** (Knight to e1)

Leveraging the strategic advantages of knights, the ImprovedAgent bot adopts a hyper-aggressive approach by prioritising knight movements in the opening phases. The sequence of predefined knight moves targets the weakness of reconnaissance blind chess by focusing on hyper-aggressive positioning without directly engaging opponent pieces.

The ImprovedBot retains adaptability in response to shifts in the game's dynamics and the opponent's actions. Should the predefined moves become unavailable or exhausted, in the `choose_move` function the bot now uses the Stockfish chess engine to analyse the board and suggest a move. This is done by setting the `board.turn` to the bot's colour, checking if the board is valid, and then calling the `engine.play` method with a time limit of 1 second. If the move suggested by Stockfish is in the `move_actions` list, the bot returns that move. This allows the bot to make more sophisticated moves based on the current state of the board.

If the bot is unable to use Stockfish for any reason, it falls back to choosing a random move from the `move_actions` list. This ensures that the bot always makes a move, even if it's not the best one.

2.2 Improved Sensing Strategy

In the `choose_sense` function, the ImprovedAgent now prioritises sensing squares with opponent's pieces. This is achieved by iterating over the `sense_actions` list and checking if any square is in the `opponent_piece_squares` list. If it is, the bot returns that square. This allows the bot to keep track of the opponent's pieces and make more informed decisions.

If no opponent's pieces are found, the bot then performs a priority-based sensing strategy where it priorities strategic squares based on predefined moves. This is done by iterating over the `strategic_moves` list and checking if the `to_square` of any move is in the `sense_actions` list. If it is, the bot returns that square. This allows the bot to focus on squares that are part of its strategic plan to reach its positional objectives.

If no strategic squares are found, the bot then priorities the most central squares on the chess board. This is done by iterating over the `central_squares` list and checking if any square is in the `sense_actions` list. If it is, the bot returns that square. This allows the bot to control the center of the board, which is a common strategy in chess and facilitates better piece mobility.

3 Results

The results from the round-robin can be seen in Table 2. Each bot competed against every other bot twice, alternating between playing as black and white in each match.

Agent Results	**Score**	**Wins**	**Losses**	**Draws**	**Win %**
RandomBot	2.0	2	4	0	33.33
TroutBot	4.0	4	2	0	66.67
RandomSensing	0.0	0	6	0	0.00
ImprovedBot	6.0	6	0	0	100.00

Table 2: Round-Robin Results

References

James, S. (April 2024). *Coms4033a assignment report* (tech. rep.). University of the Witswatersrand.