WITS
UNIVERSITY

School of Computer Science and Applied Mathematics
UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG

AI RBC REPORT

Sayfullah Jumoorty (2430888)

Muhammed Muaaz Dawood (2425639)

Mujammil Mohsin Gulam Sakhidas (2436109)

May 16, 2024

UNIVERSITY OF THE WITWATERSRAND

SCIENTIA ET LABORE

JOHANNESBURG

Contents

1 Introduction

This report outlines the development and performance evaluation of an agent tailored for Reconnaissance Blind Chess (RBC). In this strategic pursuit, the agent's construction demanded smart planning. Through a systematic approach, this document elucidates the agent's design methodologies, operational strategies, and final performance outcomes within the RBC domain.

2 Round Robin Tournament

For this experiment, we organized a round-robin tournament format to assess the performance of our ImprovedAgent compared to other models like RandomSensing, RandomBot, and TroutBot. Each agent faced off against every other agent twice, once as white and once as black. This setup allowed for a comprehensive evaluation of strategic adaptability and performance across various opponents. Through this tournament structure, we aimed to gain insights into the relative effectiveness of each agent in the context of Reconnaissance Blind Chess.

2.1 Results

In the round-robin tournament, ImprovedAgent emerged as the most successful, winning 32 tournaments and securing 227 round wins with balanced performance across both white and black positions, registering 108 and 119 wins, respectively, and only two round draws. Trailing behind, RandomSensing exhibited competitive prowess, winning 15 tournaments and securing 182 round wins, albeit slightly favoring black positions with 99 wins compared to 83 wins as white. TroutBot claimed victory in 14 tournaments and secured 182 round wins. Conversely, RandomBot struggled, achieving no tournament victories and only 7 round wins. The tournament was conducted on the MSL cluster through a specialized pipeline, facilitating simultaneous game execution for efficient evaluation. These results offer insights into each agent's strengths and weaknesses, informing their adaptability and strategic proficiency in Reconnaissance Blind Chess.

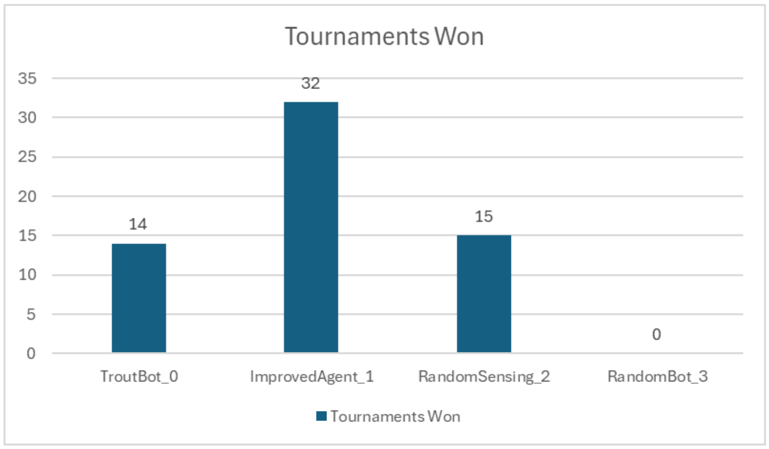

Figure 1: Tallied results of the round-robin tournament

Table 1: Results of Round-Robin Tournament

Bot	Tournaments Won	Round Wins	Wins as White	Wins as Black	Round Draws
TroutBot_0	14	182	88	94	0
ImprovedAgent_1	32	227	108	119	2
RandomSensing_2	15	182	83	99	2
RandomBot_3	0	7	5	2	0

3 Improvements

3.1 RandomSensing

RandomSensing's strategic performance was hindered by its reliance on random sensing, limiting its strategic adaptability and predictive capabilities. This approach led to suboptimal decision-making and reduced strategic effectiveness in Reconnaissance Blind Chess.

3.2 Key Improvements Implemented for ImprovedAgent

To enhance the agent's strategic capabilities, we implemented several key improvements to the sensing mechanism:

3.2.1 Prioritized Sensing

The ImprovedAgent incorporates a prioritized sensing algorithm to intelligently select the most advantageous square to sense during the Reconnaissance Blind Chess game. This algorithm aims to gather valuable information about the opponent's piece positions by prioritizing squares that have a higher potential for capturing enemy pieces in their surrounding region.

The prioritized sensing algorithm is implemented in the `choose_sense` method of the ImprovedAgent class. The algorithm works as follows:

1. If the opponent has captured one of the agent's pieces in the previous move, the algorithm focuses on a 3×3 region around the captured square. It adjusts the rank and file of the captured square to ensure that the 3×3 region falls within the bounds of the board, even if the captured square is on the edge or corner. The algorithm then returns the adjusted square as the chosen sense action.

2. If no piece has been captured, the algorithm iterates through the set of possible board states (`self.possible_fens`) and the available sense actions (`sense_actions`).

3. For each square in `sense_actions` that falls within the inner region of the board (to avoid selecting squares on the edges or corners), the algorithm evaluates the potential benefit of sensing that square by calling the `evaluate_region` function.

4. The `evaluate_region` function calculates the value of the square by summing up the piece values of the opponent's pieces in the surrounding 3×3 region. The piece values are defined in the `piece_values` dictionary,

with each piece type assigned a corresponding numerical value (e.g., pawn: 1, knight: 3, bishop: 3, rook: 5, queen: 9, king: 10).

5. The algorithm keeps track of the maximum value found so far (`max_value`) and the corresponding best square (`best_square`).

6. After evaluating all the squares, the algorithm returns the `best_square` as the chosen sense action.

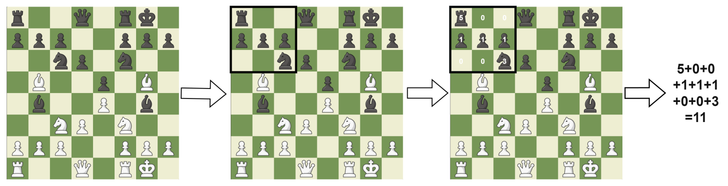

Figure 2: Visualization of the piece values used in prioritized sensing

We experimented with different piece values, particularly the king's value, to optimize the sensing prioritization. For example, when the king's value was set to 2, the ImprovedAgent achieved a win rate of 0.659, while the TroutBot's win rate was 0.340. Adjusting the king's value to 0 resulted in a win rate of 0.571 for the ImprovedAgent and 0.428 for the TroutBot. Further increasing the king's value to 10 led to the ImprovedAgent's win rate improving to 0.664, while the TroutBot's win rate decreased to 0.335. These results, summarized in Table 2, demonstrate the impact of prioritized sensing and the importance of tuning piece values to optimize the agent's performance. Since the king value of 10 showed the most significant improvement, we conducted extensive testing on this configuration, running numerous games in parallel on the MSL cluster through our pipeline to obtain an accurate understanding of its performance.

King Value	TroutBot Win Rate	ImprovedAgent Win Rate	Total Games Averaged on
2	0.340	0.659	555
0	0.428	0.571	810
10	0.335	0.664	7868

Table 2: Win rates for TroutBot and ImprovedAgent with different king values in prioritized sensing.

The rationale behind this approach is to prioritize sensing squares that are more likely to provide valuable information about the opponent's piece positions, specifically those squares where capturing enemy pieces is possible. By focusing on squares with higher potential for capturing, the agent can gather strategic insights and make more informed decisions during the game.

3.2.2 Captured Piece Information

When the opponent captures one of the agent's pieces, the agent utilizes this information to narrow down the possible board states. It identifies the captured square and focuses on a 3x3 region around that square, handling cases where the captured square is on the edges and corners of the board, mainting a full 3x3 sense region to get the most amount of information possible. The agent then filters out board states where the captured piece is still present in this 3x3 region, significantly reducing the search space.

3.2.3 Sense Result Filtering

After sensing a square, the agent filters out board states that do not match the sense result, further refining the possible board states. This allows the agent to maintain a more accurate representation of the board state based on the information gathered through sensing.

3.2.4 Check Handling

When the enemy king is in check, the ImprovedAgent prioritizes moves that capture the enemy king or moves that get the king out of check. For each possible board state, the agent identifies if the enemy king is under attack and, if so, attempts to find a move that captures the enemy king or a legal move that does not leave the king in check. This strategic consideration improves the agent's ability to handle check scenarios effectively.

4 Conclusion

By incorporating these techniques, the number of possible board states that the agent needs to consider is drastically reduced compared to a random sensing approach. This reduction in the search space allows the Stockfish engine, which is used for move evaluation, to spend more time analyzing each remaining board state and explore deeper variations. As a result, Stockfish can provide more accurate and informed move evaluations, enabling the ImprovedAgent to make better strategic decisions and improve its overall performance in Reconnaissance Blind Chess.