# HEURISTIC ANALYSIS

For an Adversarial Game Playing Agent for Isolation

Submitted by
Matt Toledo
As part of the AIND Program, Udacity

# TABLE OF CONTENTS

# SYNOPSIS

The project's goal is to develop an adversarial search agent to play the game of Isolation. This report focuses on the heuristics used for the Alpha-Beta agents.

Isolation is a deterministic, two-player game of perfect information in which players take turns moving their piece on a 7x7 chess-like board. Whenever an agent occupies a space, that space becomes blocked for the remainder of the game. The game ends when an agent is unable to make a legal move and loses the game, with the opponent being the victor.

This project uses a variation of Isolation in which the piece being used behaves like a knight in chess. The agents are allowed to move their piece in an L-shaped movement onto any available space. Movement is blocked at the edges of the board, but players can jump over blocked or occupied spaces.

Lastly, agents have a fixed time limit of 150 milliseconds to search and return an appropriate move. Should the time limit expire during an agent's turn, that agent automatically forfeits the game and the opponent is declared the victor. These rules are implemented in the isolation.py file under the Board class.

# Custom Heuristics

1. Custom 1:

   Custom 1 focuses on maximizing its available moves while still minimizing the opponent's available moves.

   $$\lambda len(own\ available\ moves) - len(opponent\ available\ moves),\ where\ \lambda\ \varepsilon\ (1,\ \infty)$$

   The value of $\lambda$ used was 1.5

2. Custom 2:

   Instead of focusing on its own moves, Custom 2 prioritizes minimizing the opponent's available moves. Maximizing its own moves comes second.

   $$len(own\ available\ moves) - \lambda len(opponent\ available\ moves),\ where\ \lambda\ \varepsilon\ (1,\ \infty)$$

   The value of $\lambda$ used was 1.5

3. Custom 3:

   Similar to Custom 1, but this time the available moves are scaled by the distance from the center of the board to the player's location. This results in moves closer to the center to be prioritized over those closer to the edge of the board.

   $$\lambda\frac{len(own\ available\ moves)}{own\ distance\ to\ center} - \frac{len(opponent\ available\ moves)}{opponent\ distance\ to\ center},\ where\ \lambda\ \varepsilon\ (1,\ \infty)$$

   The value of $\lambda$ used was 1.5

4. Custom 4:

Similar to Custom 2 and Custom 3. The goal here is to prioritize minimizing the opponent's available moves and trying to force him to the edges. At the same time we are trying to maximize our available moves while trying to stay closer to the center.

$$\frac{len(own\ available\ moves)}{own\ distance\ to\ center} - \lambda\frac{len(opponent\ available\ moves)}{opponent\ distance\ to\ center},\ where\ \lambda\ \varepsilon\ (1,\ \infty)$$

The value of $\lambda$ used was 1.5

# Evaluating Heuristics

To evaluate the effectiveness of the different heuristics, tournament.py is used to run a round-robin tournament. Multiple other predefined agents have been added as opponents for the tournament.

Since the performance of time-limited iterative deepening search is hardware dependant, an agent is used as a baseline measurement. This agent is called “ID_Improved” and it uses Alpha-Beta Search with Iterative Deepening and the “improved_score” heuristic from sample_players.py

The following agents are the opponents in the tournament:

- Random: Selects a random action each turn
- MM_Open: Uses Fixed-Depth Minimax Search with the “open_move_score” heuristic
- MM_Center: Uses Fixed-Depth Minimax Search with the “center_score” heuristic
- MM_Improved: Uses Fixed-Depth Minimax Search with the “improved_score” heuristic
- AB_Open: Uses Iterative Deepening Alpha-Beta Search with the “open_move_score” heuristic
- AB_Center: Uses Iterative Deepening Alpha-Beta Search with the “center_score” heuristic
- AB_Improved: Uses Iterative Deepening Alpha-Beta Search with the “improved_score” heuristic
- AB_Custom: Uses Iterative Deepening Alpha-Beta Search with the “custom_score” heuristic
- AB_Custom_2: Uses Iterative Deepening Alpha-Beta Search with the “custom_score_2” heuristic
- AB_Custom_3: Uses Iterative Deepening Alpha-Beta Search with the “custom_score_3” heuristic
- AB_Custom_4: Uses Iterative Deepening Alpha-Beta Search with the “custom_score_4” heuristic

# Results

Bellow are the final results of the tournament between the four custom heuristics and the baseline agent. Each agent played 500 matches against each opponent. With a match consisting of two games in which each agent takes a turn at going first.

| Rank | Agent | Win Rate |
|---|---|---|
| 1 | AB_Custom_1 | 65.59% |
| 2 | AB_Custom_2 | 64.96% |
| 3 | AB_Improved | 64.21% |
| 4 | AB_Custom_3 | 63.27% |
| 5 | AB_Custom_4 | 63.12% |

Based on the results of the tournament the best agent is AB_Custom_1 for the following reasons:

1. It achieved the highest win rate out of all the other agents at 65.59%
2. It won more than 50% of its games against the other AB_Custom agents
3. It had the highest win rate versus the other opponents in the tournament that were not ranked at 75.20%

# Appendices

## A. Appendix: Tournament Results

| Opponent | AB_Improved | | AB_Custom_1 | | AB_Custom_2 | | AB_Custom_3 | | AB_Custom_4 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Won | Lost | Won | Lost | Won | Lost | Won | Lost | Won | Lost |
| Random | 932 | 68 | 942 | 58 | 939 | 61 | 932 | 68 | 943 | 57 |
| MM_Open | 750 | 250 | 769 | 231 | 765 | 235 | 743 | 257 | 730 | 70 |
| MM_Center | 884 | 116 | 897 | 103 | 876 | 124 | 866 | 134 | 871 | 129 |
| MM_Improved | 722 | 278 | 749 | 251 | 756 | 244 | 730 | 270 | 708 | 292 |
| AB_Open | 514 | 486 | 537 | 463 | 510 | 490 | 518 | 482 | 531 | 469 |
| AB_Center | 588 | 412 | 618 | 382 | 584 | 416 | 572 | 428 | 579 | 421 |
| AB_Improved | --- | --- | 496 | 504 | 513 | 487 | 483 | 517 | 490 | 510 |
| AB_Custom_1 | 501 | 499 | --- | --- | 499 | 501 | 469 | 531 | 469 | 531 |
| AB_Custom_2 | 499 | 501 | 506 | 494 | --- | --- | 503 | 497 | 492 | 508 |
| AB_Custom_3 | 502 | 498 | 527 | 473 | 530 | 470 | --- | --- | 499 | 501 |
| AB_Custom_4 | 529 | 471 | 518 | 482 | 524 | 476 | 484 | 516 | --- | --- |
| **Win Rate** | **64.21%** | | **65.59%** | | **64.96%** | | **63.27%** | | **63.12%** | |