Heuristic Analysis for Isolation Game Playing Agent

AI Nanodegree Project

-Manasi Kulkarni

In this project, we are developing an adversarial search agent to play the game "Isolation". To give the little bit of background about the game, Isolation is a deterministic, two-player game of perfect information in which the players alternate turns moving a single piece from one cell to another on a board. Whenever either player occupies a cell, that cell becomes blocked for the remainder of the game. The first player with no remaining legal moves loses, and the opponent is declared the winner.

This project uses a version of Isolation where each agent is restricted to L-shaped movements (like a knight in chess) on a rectangular grid. The agents can move to any open cell on the board that is 2-rows and 1-column or 2-columns and 1-row away from their current position on the board. Movements are blocked at the edges of the board; however, the player can "jump" blocked or occupied spaces (just like a knight in chess).

The main objectives of this project are:

- ➢ Implement Minimax algorithm for searching the game tree.
- ➢ Implement Alpha-Beta pruning to improve the efficiency of the minimax game tree search.
- ➢ Implement fixed-depth and iterative deepening search where the Isolation game AI agent returns the best next move within the time provided to it.
- ➢ Implement heuristic evaluation functions that perform equally or better than the provided heuristic evaluation functions.

For this project, we experimented with five different heuristics. To estimate and compare the performance of the proposed heuristic evaluation functions to the ID-Improved agent, we simulate 5 tournaments each consisting of 20 matches against every opponent for a heuristic evaluation function.

The proposed heuristic evaluation functions are described as follows:

Heuristic1

With this heuristic, the more available moves ‘player` has available from the evaluated position, the better. This function simply returns the difference in number of legal moves left between the players. If `player` and its opponent have the same number of moves, then the returned value is zero. If the returned value is positive, then `player` is doing better than its opponent. If the returned value is "inf", then `player` has won the game and vice a versa.

Table 1: Table showing the results for Heuristic1 in 5 tournaments each consisting of 20 matches per opponent.

Tournament No.	ID_Improved	Student (Heuristic1)
1	77.14%	61.43%
2	81.14%	79.29%
3	80.00%	70.71%
4	66.43%	60.71%
5	80.71%	81.43%

It is clear from the table it is not a great heuristic. On the positive side, it is very fast to compute and easy to interpret but it performed poorly due to lack of strategy implementation in the function for the game.

Heuristic2
With this heuristic too, the more moves the player has available from the evaluated position, the better. At the same time, it also has positional advantage. If a player's position is closer to the center of the board, it is more probable that a player can do better than the player whose remaining moves are near the edge of the board because of which they will have less options to move down the line.

Manhattan distance was used instead of the Euclidean distance for speeding up the runtime execution. The results for this heuristic is shown in the following table.

Table 2: Table showing the results for Heuristic2 in 5 tournaments each consisting of 20 matches per opponent.

Tournament No.	ID_Improved	Student (Heuristic2)
1	80.00%	72.14%
2	82.86%	75.71%
3	83.57%	85.71%
4	79.29%	76.43%
5	75.71%	75.00%

This heuristic does have a positional advantage and performs a bit better, but still lacks the strategic awareness of the game.

Heuristi3

For heuristic 3 we modified the 'Open moves score' heuristic provided to us by weighting each open move by a weight that depends on the position the move leads to on the game board.
Open Moves Score: This function returns the number of moves available to player in a given board state as the score for non-terminal states.
The motivation is that positions in the center of the board provide more open moves than those near the edges. The weight for a position is set as the maximum number of moves available from that position.

Table 3: Table showing the results for Heuristic3 in 5 tournaments each consisting of 20 matches per opponent.

Tournament No.	ID_Improved	Student (Heuristic3)
1	77.14%	74.28%

2	87.29%	84.29%
3	81.43%	85.00%
4	79.80%	77.45%
5	81.30%	81.00%

Heuristic4

Difference in Weighted Open Moves: In Heuristic4 we are calculating the difference in the weighted open moves scores (Heuristic3) between the current player and it's opponent and use that as the score for the current game state.

Table 4: Table showing the results for Heuristic4 in 5 tournaments each consisting of 20 matches per opponent.

Tournament No.	ID-Improved	Student(Heuristc4)
1	78.57%	80.00%
2	88.57%	77.86%
3	88.58%	89.29%
4	78.57%	75.00%
5	85.71%	78.57%

Heuristic5

Difference in Open Moves One Ply Ahead: In Heuristic5 the difference in the number of available moves between the current player and its opponent one ply ahead in the future is used as the score of the current game state.

Table 5: Table showing the results for Heuristic5 in 5 tournaments each consisting of 20 matches per opponent.

Tournament No.	ID_Improved	Student(Heuristic5)
1	72.86%	80.00%
2	80.00%	88.57%
3	77.86%	80.00%
4	88.57%	90.71%
5	86.43%	89.29%

Results and Discussion

The performance of the Isolation playing AI agents is evaluated using a tournament setup which consists of a Random AI agent that moves randomly and a set of AI agents utilizing minimax search (MM) and minimax with alpha-beta pruning (AB) search algorithms with fixed depth game trees as the opponents which use the Null, Open, Improved heuristic evaluation functions and the player (student) includes agents that use iterative deepening (ID) with the Improved (Heuristic1, Heuristic2 and Heuristic3, Heuristic4, and Heuristic5) heuristic evaluation functions. To estimate and compare the performance of the proposed heuristic evaluation functions to the ID-Improved agent, we simulate 5 tournaments each consisting of 20 matches against every opponent for a heuristic evaluation function. The results of which is shown Table1, Table2, Table3, Table4, and Table5 respectively.

From the performance observed we can recommend ‘Heuristic5’ heuristic evaluation function over all other heuristic functions mentioned above and can be justified as follows:

- We observe that the student/ player with ‘Heuristic5’ consistently performs better than all the other heuristics considered.
- This heuristic (Heuristic5) is based on the number of moves looking ahead one ply in the future. This is important in this version of isolation where only L-shaped knight like moves are allowed. Due to these kinds of moves, its difficult to predict the value of a game state by just counting the number of immediately available moves. Hence one ply lookahead should provide a more accurate evaluation function.
- Also, the ‘Heuristic5’ running time complexity is comparable to the ‘Improved’ heuristic and hence it should not adversely affect the maximum depth searched.
- Finally, as noted from the results, the ‘Heuristic5’ does perform better in practice and hence can be recommended.

.