Andrew Sutton
Artificial Intelligence (Term 1)
August 20, 2017

Heuristic Analysis

This project developed and tested multiple heuristic 'value' functions for an AI agent to play the modified 'Isolation' game. Isolation is played on an 8 x 8 chess board with one piece per player. On each move, a player must move their piece to a never occupied square. In the game played during the development of this agent, the piece moves as a knight.
Several heuristic functions were developed to address this challenge. The best performing heuristic function is Custom 1, which approximates the number of open moves two turns ahead. This heuristic is found to beat the overall win-rate of the AB_Improved implementation, however in head-to-head testing the AB_Improved surpasses the Custom 1 algorithm.

Custom 1: Second Order moves via Direct Evaluation

This heuristic function calculates the number of spaces available to a piece after two moves, compared to the same metric for the opponent. This metric will be correlated with the number of spaces available to each player, and provides additional information about the game state that is not captured by the 'improved_score' value function.

The board is operated upon directly and no expansion of the game-tree is performed in this function. The value function pre-calculates the set of possible second-order moves and then evaluates the legality of these moves from the current position. The function returns the number of second order moves that are valid.

This function is an approximation to a two-level sub-tree expansion. The reasoning behind this approximation, is the claim that it better captures the future board state than the first order' move count implemented in the 'improved_score'. Direct evaluation of the move legality, whilst ignoring first order legality is offset by the speed of evaluation. This will enable a deeper sub-tree expansion relative to more 'accurate' algorithms.

To simplify calculation, the function ignores potential conflicts between player 1 and player 2, such that only the current agents moves are evaluated. Similarly, this implementation ignores the legality of the first move, since the set of possible second order moves is pre-calculated.

- Custom 2: Second Order moves via 'Full' Evaluation:

This heuristic function calculates the same 'goal' function as Custom 1, however it evaluates the number of legal second order moves

by explicit expansion the game board into child states. This means that the Custom 2 algorithm correctly calculates the legality of the ‘first order’ moves, a feature ignored in Custom 1. The cost of this is additional compute complexity, due to the need to explicitly forecast the board state.

- Custom 3: Custom 1 + Early-Game

The third heuristic attempted to expand the Custom 1 algorithm by including an aditional set of value conditions to influence early game behavior. Specifically, the conditions attempted to ‘force’ specific moves in the early game, particularly related to centering of the player position on the board. This mechanism favors ‘centering’ moves in the early game, explicitly within the first 2 moves and via use of the ‘Center’ heuristic within the first 10 moves.

Results:
Below is a summary of the achieved performance for the different implementations. This demonstrates that, within the margin of repeatability, all three algorithm implementations achieve comparable performance to the improved_score metric.

****
Playing Matches
****

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	34	6	38	2	32	8	34	6
2	AB_Open	19	21	20	20	13	27	16	24
3	AB_Center	22	18	27	13	15	25	22	18
4	AB_Improved	20	20	18	22	16	24	19	21
	Win Rate:	59.4%		64.4%		47.5%		56.9%	

Discussion:

As is demonstrated in the results, the Custom 1 heuristic function performance exceeded the ‘improved_score’ function with a win rate of 64.4% to 59.4% overall. Thus, the Custom 1 metric can be applied to general games. However, in head-to-head between the AB_Improved and AB_Custom agent, which implements the Custom 1 heuristic function, the AB_Improved wins 22 - 18. This is attributed to the inter-play between algorithm assumptions: the Custom 1 algorithm makes an assumption relating the the availability of first

order moves for a player. Since the improved_score acts exclusively on the first-order basis, it is likely that the improved_score has an advantage in head-to-head play.

Note, the Custom 1 agent significantly out-performs the Custom 2 agent, whilst implementing the say underlying value function (the number of second order moves). This indicates that the increased accuracy of the Custom 2 function is not offset by the computational burden: the Custom 1 agent can search to a greater depth due to the reduced compute burden.

The Custom 3 achieves intermediate performance, indicating that the 'Centering' strategy does not contribute significantly to the performance of the Agent in the general. This is likely a result of the move randomization applied for the opening move in the tournament.py script.

To determine the impact of the Custom 1 assumptions, an additional heuristic Custom 4 was implemented as a fusion of Custom 1, improved_score and center_score as:

```
custom_4 = improved_score(game, player)
                    + 0.5 * custom_score(game, player)
                    + 0.5 * center_score(game, player)
```

The resulting heuristic was added to a simulation against the AB_Improved agent for the following result:

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3		AB_Custom_4	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	AB_Improved	20	20	19	21	10	30	17	23	20	20
	Win Rate:	50.0%		47.5%		25.0%		42.5%		50.0%	

In short, it appears that the fusion selected did not provide a statistically significant advantage. Similarly, the AB_Improved and AB_custom algorithms remain close in performance. With further efforts, the specific weighting used in the score fusion could be optimized using iterative techniques.