Evaluating Performance of Heuristics in AIND Project 2
by Levan Iordanishvili
Version 2

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	43	7	38	12	41	9	38	12
2	MM_Open	27	23	33	17	28	22	29	21
3	MM_Center	40	10	41	9	39	11	38	12
4	MM_Improved	36	14	33	17	31	19	32	18
5	AB_Open	25	25	23	27	26	24	27	23
6	AB_Center	27	23	30	20	32	18	33	17
7	AB_Improved	27	23	27	23	29	21	26	24
	Win Rate:	64.3%		64.3%		64.6%		63.7%	

(Fig. 1.0)

Custom Score

Divides the difference between own moves and opponents moves by the distance between players. If there are fewer moves left, a greater emphasis is based on aggressively going after the opponent (see line 83 in game_agent.py).

***Custom Score 2 [BEST]*:**

- Feature 1: *difference between "own moves" and "opponent's moves:"* covered in the lecture videos. The project rubric suggests that students should attempt to beat the *AB_Improved* function, and as a result I've used it as the feature with the most weight, and further enhanced it with Feature 2 and 3.
- Feature 2: *distance from center:* provides the distance between the player and the center of the board. The idea behind the feature is that its value could be used to encourage play from the center of the board. Playing from the center could provide the player with potentially more successful moves.
- Feature 3: *distance between players*: The intuition behind this feature is that it might be useful to control how closely the player tries to stick to, or stay away from, the opponent.

Custom Score 2 places a significant amount of weight on Feature 1, and about half of Feature 1's weight on Feature 2. The heuristic enhances the primary feature by placing greater emphasis on playing from the center of the board. Feature 3 plays a lesser role but has a clear effect on the performance of the function considering the 3.1% performance improvement over a *Custom Score 2* variant in an earlier experiment (see Fig 1.1).

It wasn't immediately clear which heuristic performed better due to variance in the win rates. However, the larger samples size, a run of 25 matches, established *Custom Score 2* as the best.

Inspired by the Deep Blue paper, by its description of the 8000 features and the way the Deep Blue team tuned the weights of the features (76), I decided to split the feature values from the weight values. The evaluation function relies on matrix multiplication as an efficient way to calculate the sum of weights multiplied by features. Though this hardly makes a difference considering the three features present in the evaluation functions, it does provide an extensible, cleaner way to decouple weights from features and adjust them easily.

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	44	6	39	11	44	6	43	7
2	MM_Open	34	16	33	17	31	19	33	17
3	MM_Center	42	8	43	7	44	6	39	11
4	MM_Improved	32	18	33	17	30	20	32	18
5	AB_Open	27	23	24	26	27	23	24	26
6	AB_Center	29	21	24	26	30	20	30	20
7	AB_Improved	20	30	27	23	30	20	24	26
	Win Rate:	65.1%		63.7%		67.4%		64.3%	

(Fig. 1.1)

Custom Score 2 vs its variants with different weights assigned to the three features

Custom Score 3

Provides a simple variant on *AB_Improved* by assigning double the weight to the opponent's moves, encouraging the player to adopt a more aggressive strategy.

Choosing the Best Evaluation Function

Custom Score 2 is recommended as the best evaluation function to use.

1) *Custom Score 2* has the best win rate. In the original experiment it performed 2.3% (Fig. 1.1) better than *AB_Improved* and in the latest experiment it performed 0.3% (Fig 1.0) better than *AB_Improved*. The weights could likely be further tuned to achieve even better performance.
2) The implementation of the function allows for clear decoupling of features and their weights, it allows for a simple and clear way of updating the weights without having to update the actual implementation of the score function -- the latter is taken care of by the matrix multiplication itself.
3) The feature and weight matrix approach allows the score function to be more generic and reusable. The matrix structure could also lend itself to greater extensibility in the future, weights could be optimized with a technique like gradient descent. For example, a generic score function could more easily be incorporated in an automated process which continually re-runs the tournament and adjusts the weights of the features automatically.

References:

* Campbell, Murray, et al. “Deep Blue.” *Artificial Intelligence*, Jan. 2002, pp. 57–83.
* High level ideas for the features came from my discussions with my mentor.

* The latest implementations, in Version 2, for Custom Score and Custom Score 3 came from suggestions by my project submission reviewer.