AIND-Isolation Heuristic Analysis

Evaluation Heuristic Techniques

I chose the following three for this analysis because they had the highest win rates of all that I tested with. They can all be viewed from my github repo in the game_agent.py file.

Moves_available_chase_opponent (AB_Custom)

```
return float(own_moves  /  (1 + opp_moves ** 2))
```

Moves_available_defensive_start (AB_Custom2)

```
total_cells = game.width ** 2
cells = len(game.get_blank_spaces())
return float(cells * own_moves - (total_cells - cells) * opp_moves)
```

Moves_available_distance_to_center_negative (AB_Custom3)

```
return float(10 * (own_moves - opp_moves) - (own_x + own_y) + (opp_x + opp_y))
```

	AB_Improved	AB_Custom	AB_Custom_2	AB_Custom_3
Win Rate:	70.0%	74.3%	68.6%	68.6%
Win Rate:	74.3%	74.3%	68.6%	62.9%
Win Rate:	68.6%	67.1%	71.4%	71.4%
Win Rate:	71.4%	70.0%	64.3%	81.4%
Win Rate:	70.0%	71.4%	70.0%	65.7%
AVG WIN RATES:	70.8%	71.42%	68.5%	70%

Conclusion

I chose the "moves_available_chase_opponent" heuristic because it consistently outperformed AB_Improved. It looks at how many moves are available and chooses the next move that will increase it's own chance of winning.

AB_Custom2 divides the game board in half, looks at how many moves each player has left and chooses the next move to decrease the opponent's chance of winning. AB_Custom3 also divides the board and looks at the moves available, but it adds a negative coefficient to the player's location because I was curious to see how that will perform in the tournament.