Evaluating effectiveness of Custom Heuristics in Game of Isolation

Udacity Artificial Intelligence Nanodegree - Term 1

Submitted By
Srikanth Mannepalle

Summary

In creating custom heuristics for the game of isolation, following three approaches were evaluated. For sake of consistency, five test simulations were run to gauge the effectiveness of each of these strategies with respect to ID_Improved.

A.) Create a substitute function for static evaluation at any node, that helps in determining the sensitivity of node value w.r.t to its values back propagated by its children
B.) Evaluate a relative score of the number of available moves for the player w.r.t its opponent
C.) Calculate the Manhattan distance of the player from the center of the board, and evaluate it w.r.t to the same distance of its competitor

Analysis

Custom Score: The rationale in creating this heuristic was to mimic the approach used in research paper “Game Tree Searching by Min/Max Approximation” by Ronald L. Rivest. At any node of the game, the rate of change of the Manhattan distance from center of the board is used as a deciding criterion to select the best node to expand. Moreover, rather than taking partial derivative of the distance w.r.t to a node value a simple proportion (d1 – d) / d. This approach performed as good as the AB_Improved.

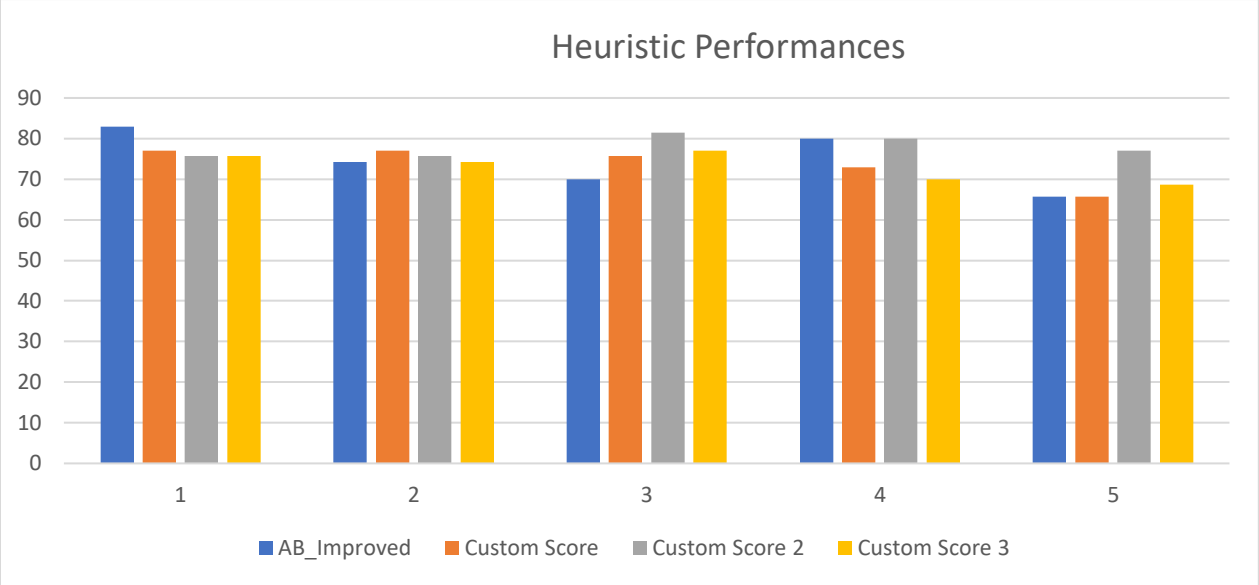
```
def custom_score(game, player):
    if game.is_loser(player):
        return float("-inf")

    if game.is_winner(player):
        return float("inf")

    moves_legal = game.get_legal_moves(player)
    w, h = game.width / 2., game.height / 2.
    y, x = game.get_player_location(player)
    d = float((h - y)**2 + (w - x)**2)
    min = float("inf")
    for m in moves_legal:
        x1, y1 = m
        d1 = float((h - y1)**2 + (w - x1)**2)
        val = (d1 - d) / d
        if val <= min:
            min = val
    return 1/(1+min)
```

Custom Score 2:
The rationale behind choosing this heuristic was to evaluate the relative score of the number of available moves for the player w.r.t its opponent. The inverted ratio results in a quadratic function i.e. the difference between the squares of the value of # available moves of player and the # available moves of the opponent. This strategy was better against AB_Improved, winning 4 – 1 times and that too with a good win rate

```
def custom_score_2(game, player):
    if game.is_loser(player):
        return float("-inf")

    if game.is_winner(player):
        return float("inf")

    own_moves = len(game.get_legal_moves(player))
    opp_moves = len(game.get_legal_moves(game.get_opponent(player)))

    if own_moves == 0:
        return float("-inf")

    if opp_moves == 0:
        return float("inf")

    return float(own_moves/opp_moves - opp_moves/own_moves)
```

Custom Score 3:
The rationale behind choosing this heuristic was to first calculate the Manhattan distance of the player from the center of the board, and evaluate it w.r.t to the same distance of its competitor. The assumption was that the nearer the player is to the center, the more are the number of moves available and hence can result in cornering the opponent easily. This might not be true in later stages of the game as mentioned in the lecture videos. Again, this strategy performed as good as the Custom score strategy

```
def custom_score_3(game, player):
    if game.is_loser(player):
        return float("-inf")

    if game.is_winner(player):
        return float("inf")

    w, h = game.width / 2., game.height / 2.
    y, x = game.get_player_location(player)
    y1, x1 = game.get_player_location(game.get_opponent(player))
    d1 = float((h - y)**2 + (w - x)**2)
    d2 = float((h - y1)**2 + (w - x1)**2)
    return float(d2/d1)
```



Recommendation:

As can be seen, the accuracy of prediction can be improved by playing more than 10 games, say 50, against each of the 7 opponents. However, with the given facts and figures, it is evident that the logic used in Custom Score 2 function results in better win rate. The method incorporates Alpha beta pruning, Depth restriction under time constraint, and the custom score logic. Therefore, of the three methods, I recommend the strategy that involves calculation of the available number of moves for a player to be more rewarding in the game of isolation

Appendix:

This script evaluates the performance of the custom_score evaluation function against a baseline agent using alpha-beta search and iterative deepening (ID) called `AB_Improved`. The three `AB_Custom` agents use ID and alpha-beta search with the custom_score functions defined in game_agent.py.

Simulation 1:

```
 Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random      10  |   0    10  |   0    10  |   0    10  |   0
    2       MM_Open     10  |   0     8  |   2    10  |   0     8  |   2
    3      MM_Center    10  |   0    10  |   0    10  |   0    10  |   0
    4     MM_Improved   10  |   0     8  |   2     8  |   2     8  |   2
    5       AB_Open      5  |   5     7  |   3     6  |   4     7  |   3
    6      AB_Center     8  |   2     6  |   4     5  |   5     4  |   6
    7     AB_Improved    5  |   5     5  |   5     4  |   6     6  |   4
--------------------------------------------------------------------------
           Win Rate:      82.9%        77.1%        75.7%        75.7%

Your agents forfeited 110.0 games while there were still legal moves available to play.
```

Simulation 2:

```
 Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random      10  |   0    10  |   0    10  |   0     9  |   1
    2       MM_Open      9  |   1     9  |   1     7  |   3     8  |   2
    3      MM_Center     9  |   1     9  |   1    10  |   0    10  |   0
    4     MM_Improved    7  |   3     8  |   2     9  |   1     9  |   1
    5       AB_Open      7  |   3     7  |   3     7  |   3     6  |   4
    6      AB_Center     5  |   5     6  |   4     5  |   5     4  |   6
    7     AB_Improved    5  |   5     5  |   5     5  |   5     6  |   4
--------------------------------------------------------------------------
           Win Rate:      74.3%        77.1%        75.7%        74.3%

Your agents forfeited 104.0 games while there were still legal moves available to play.
```

Simulation 3:

```
 Match #    Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                         Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random        10  |   0    10  |   0    10  |   0    10  |   0
    2       MM_Open        8  |   2     7  |   3    10  |   0     7  |   3
    3      MM_Center      10  |   0    10  |   0    10  |   0     9  |   1
    4     MM_Improved      6  |   4     9  |   1     9  |   1    10  |   0
    5       AB_Open        5  |   5     5  |   5     6  |   4     5  |   5
    6      AB_Center       6  |   4     5  |   5     6  |   4     6  |   4
    7     AB_Improved      4  |   6     7  |   3     6  |   4     7  |   3
--------------------------------------------------------------------------
           Win Rate:      70.0%        75.7%        81.4%        77.1%

There were 1.0 timeouts during the tournament -- make sure your agent handles search
timeout correctly, and consider increasing the timeout margin for your agent.


Your agents forfeited 105.0 games while there were still legal moves available to play.
```

Simulation 4:

```
 Match #    Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                         Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random        10  |   0    10  |   0    10  |   0    10  |   0
    2       MM_Open        9  |   1     5  |   5     9  |   1     7  |   3
    3      MM_Center      10  |   0     9  |   1    10  |   0    10  |   0
    4     MM_Improved     10  |   0     9  |   1     8  |   2     6  |   4
    5       AB_Open        5  |   5     6  |   4     6  |   4     5  |   5
    6      AB_Center       6  |   4     5  |   5     8  |   2     5  |   5
    7     AB_Improved      6  |   4     7  |   3     5  |   5     6  |   4
--------------------------------------------------------------------------
           Win Rate:      80.0%        72.9%        80.0%        70.0%

Your agents forfeited 102.0 games while there were still legal moves available to play.
```

Simulation 5:

```
 Match #    Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                         Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random        10  |   0    10  |   0    10  |   0    10  |   0
    2       MM_Open       10  |   0     7  |   3    10  |   0     7  |   3
    3      MM_Center      10  |   0    10  |   0    10  |   0     9  |   1
    4     MM_Improved      5  |   5     8  |   2     9  |   1     7  |   3
    5       AB_Open        4  |   6     5  |   5     5  |   5     6  |   4
    6      AB_Center       5  |   5     4  |   6     5  |   5     5  |   5
    7     AB_Improved      2  |   8     2  |   8     5  |   5     4  |   6
--------------------------------------------------------------------------
           Win Rate:      65.7%        65.7%        77.1%        68.6%

There were 1.0 timeouts during the tournament -- make sure your agent handles search
timeout correctly, and consider increasing the timeout margin for your agent.
```