**Heuristic Analysis**

Rob Straker

I reviewed the outcomes of 3 distinct evaluation functions:

* **ratio\_score = # own moves / (# opponent’s moves + 1)**   
  This uses a variation of the mobility function presented in the lectures, but instead of calculating the difference in number of legal moves between you and your opponent, it calculates a ratio (adding 1 to denominator to avoid division by zero).
* **edge\_score = ratio\_score \* (opponent’s edge factor / own edge factor)**This is a variation of ratio\_score, which multiplies a bonus factor based on the number of edges near your opponent compared to the number near you. This stems from the idea that you have fewer options when you are within one square of an edge due to the L-shape of the move. The bonus factor ranges from 1 (where your situations are comparable) to 3 (where your opponent is in a corner and you are not near any edges).
* **space\_score = (# blank spaces - # opponent’s moves) / (# blank spaces - # own moves)**

This is another variation of ratio\_score, except that the ratio involves differences between total blanks spaces left on the board and the number of legal moves possible for each player in the current position.

The tournament.py test produces slightly different results each time it is run. To account for this variation, I ran the test 5 times for each evaluation function, and took the average differential compared to the ID\_improved program as a measure of its performance. If I had more time, I would have run tournament.py 100 times for each evaluation function to improve the accuracy of the comparison.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test** | **ID\_Improved** | **Student** | **Average Differential** |
| ratio\_score | 59.29 % | 72.14 % | + 3.43 % |
| 66.43 % | 71.43 % |
| 75.00 % | 74.29 % |
| 66.43 % | 65.71 % |
| 70.71 % | 71.43 % |
| edge\_score | 67.86 % | 70.71 % | + 4.57 % |
| 67.14 % | 78.57 % |
| 68.57 % | 67.86 % |
| 71.43 % | 76.43 % |
| 71.43 % | 75.71 % |
| space\_score | 67.14 % | 61.43 % | - 2.57 % |
| 70.00 % | 66.43 % |
| 72.14 % | 65.71 % |
| 67.14 % | 66.43 % |
| 72.14 % | 75.71 % |

The agent\_test.py unit test produces slightly different time measures every time it is run, so the test was run 5 times for each heuristic. Again, if I had more time, I would run this function 100 ties to get more consistent results.

|  |  |  |
| --- | --- | --- |
| **Test** | **Time to Run 7 Tests** | **Average Runtime** |
| ratio\_score | 9.658 s | 9.15 s |
| 9.186 s |
| 9.229 s |
| 8.787 s |
| 8.882 s |
| edge\_score | 9.344 s | 9.29 s |
| 9.250 s |
| 9.394 s |
| 9.066 s |
| 9.400 s |
| space\_score | 9.340 s | 9.10 s |
| 9.660 s |
| 9.065 s |
| 8.794 s |
| 8.649 s |

The edge\_score heuristic appears to the best evaluation function tested for the following reasons:

1. Highest Winning Percentage. The average winning percentage for edge\_score was a full 1% higher than ratio\_score, and 7% higher than space\_score.
2. Multiple Variable Function. Edge\_score was the only function to take into account more than one variable. In this case, it considered both the number of valid moves and proximity to edges (which reduces the number of degrees of freedom). This makes it a more sophisticated function than the other two.
3. Speed. Although the average runtime of agent\_test.py was slowest for edge\_score, the variation between all three heuristics was less than 2%. This implies very little degradation in speed of edge\_score versus the other heuristics.