Heuristic Analysis

Nathan Findley

March 2017

Contents

1 Non-Heuristic Solutions **2**

2 Consume The Opponent's Movement Space **5**

3 Remain Near The Center **7**

4 Central Hover Followed By Most Moves **8**

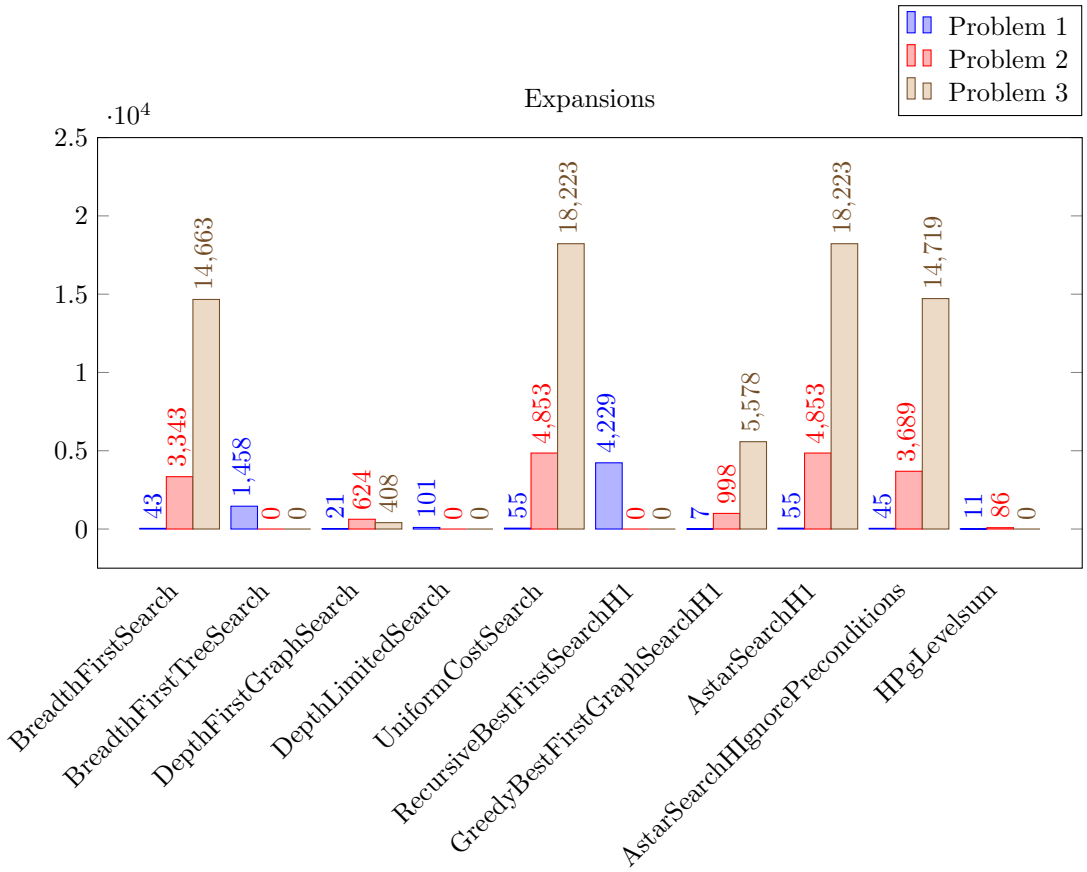
5 Results **9**

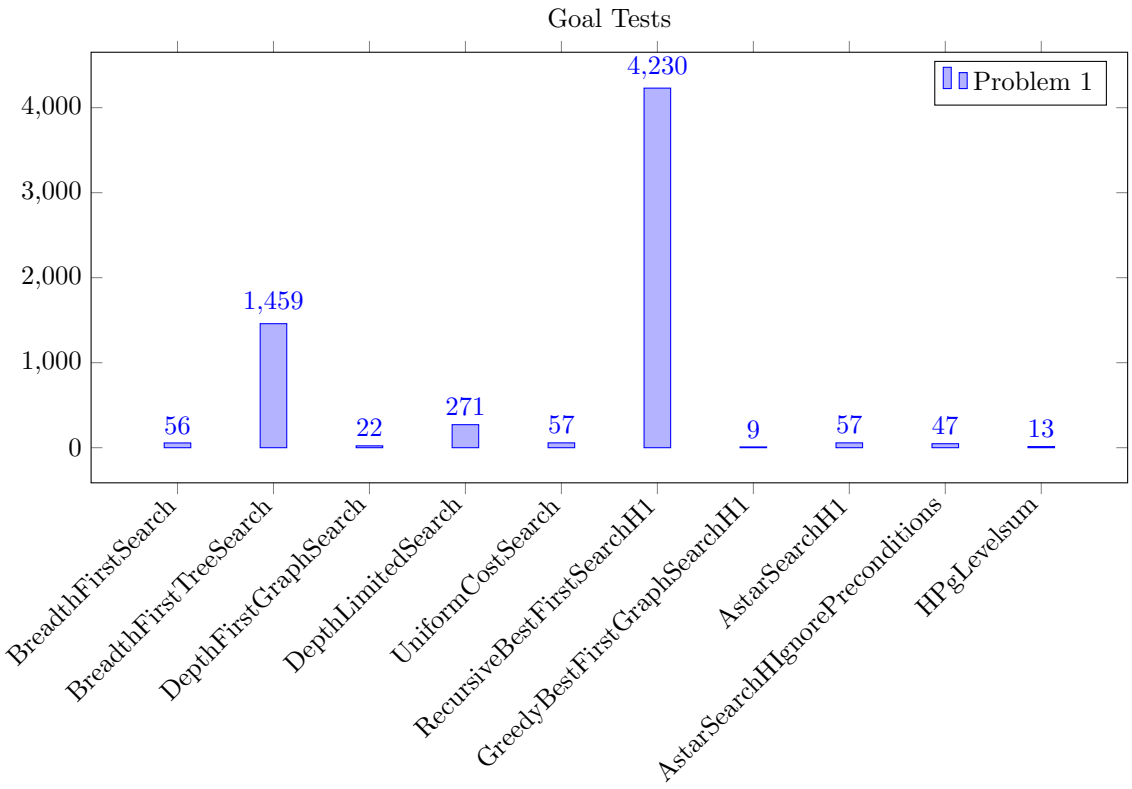
6 Beyond Project Scope - Breaking Changes **9**

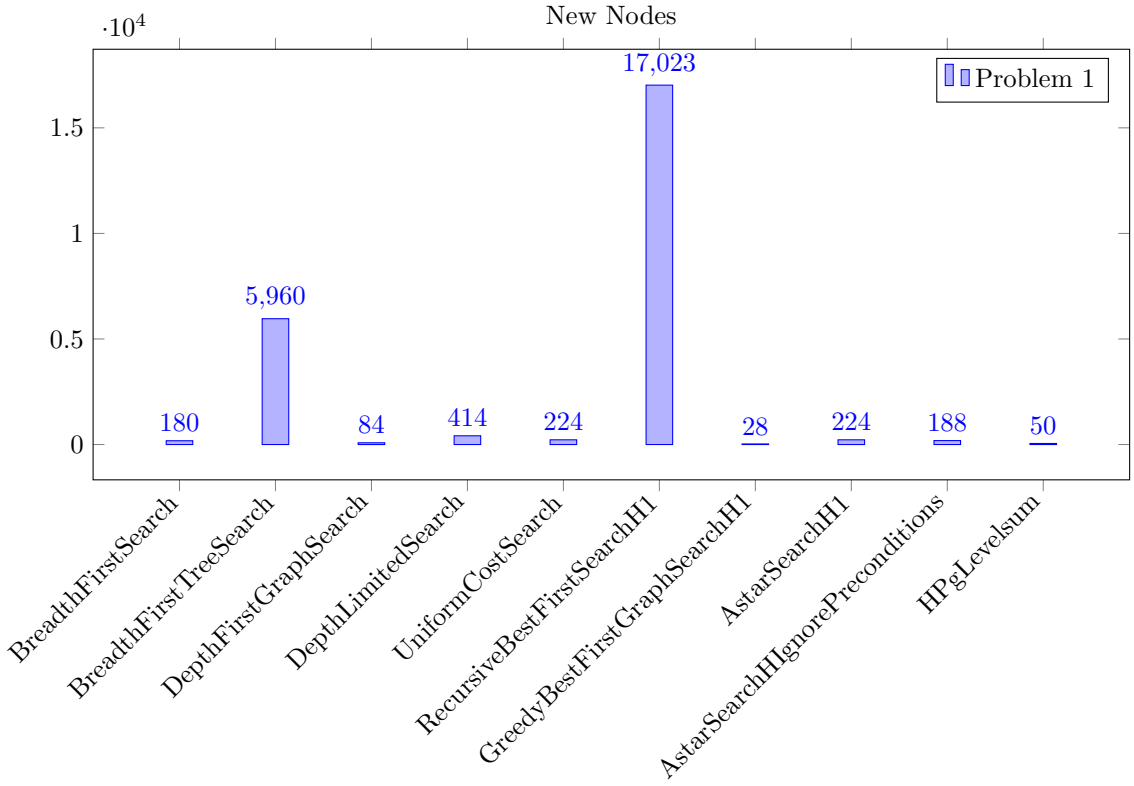
Abstract

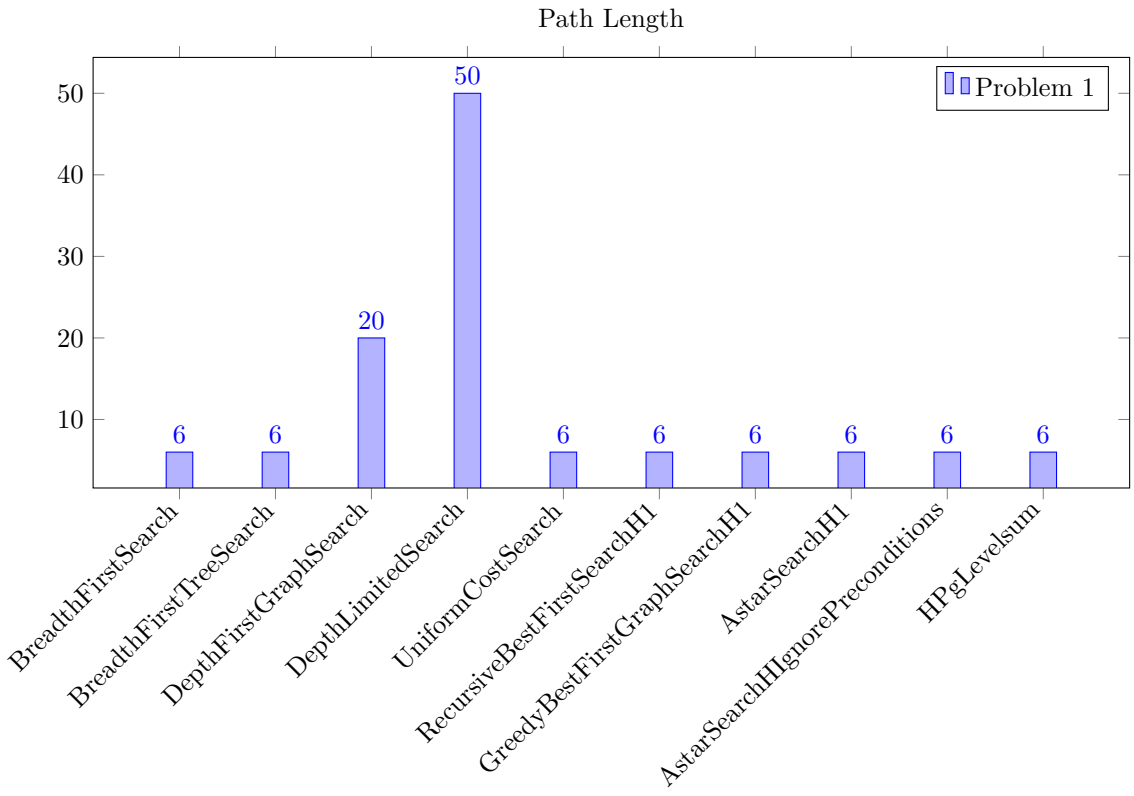
Abstract

1 Non-Heuristic Solutions




Goal Tests
Problem 1
4,000
3,000
2,000
1,000
0
56
1,459
22
271
57
4,230
9
57
47
13
BreadthFirstSearch
BreadthFirstTreeSearch
DepthFirstGraphSearch
DepthLimitedSearch
UniformCostSearch
RecursiveBestFirstSearchH1
GreedyBestFirstGraphSearchH1
AstarSearchH1
AstarSearchHIgnorePreconditions
HPgLevelsum


New Nodes
$\cdot 10^4$
Problem 1
1.5
1
0.5
0
180
5,960
84
414
224
17,023
28
224
188
50
BreadthFirstSearch
BreadthFirstTreeSearch
DepthFirstGraphSearch
DepthLimitedSearch
UniformCostSearch
RecursiveBestFirstSearchH1
GreedyBestFirstGraphSearchH1
AstarSearchH1
AstarSearchHIgnorePreconditions
HPgLevelsum



2 Consume The Opponent's Movement Space

Favor moves where the player encroaches on the opponent's moves or follow them around the board. Also uses the idea of trying to stop an opponent that suddenly has limited movement.

```
playerMoves = game.get_legal_moves(player)
opponentMoves = game.get_legal_moves(game.get_opponent(player))
if len(opponentMoves) == 1:
                    for o in opponentMoves:
                                        for p in playerMoves:
                                                            if o == p:



matchedMoves = 0
centerAt = board_center(game)
for o in opponentMoves:
                    for p in playerMoves:
                                        if o == p:
                                                            matchedMoves +=
if matchedMoves > 0:
                    return float(matchedMoves)
```

```
                playerAt = game.get_player_location(player)
                opponentAt = game.get_player_location(game.get_opponent(player))

                xDiff = playerAt[0] - opponentAt[0]
                yDiff = playerAt[1] - opponentAt[1]
                denom = float(xDiff*xDiff + yDiff*yDiff)
                if denom == 0:
                                return float("+inf")
                return float(1.0/denom)
```

```
*************************
 Evaluating: ID_Improved
*************************

Playing Matches:
----------
Match 1: ID_Improved vs   Random     Result: 184 to 16
Match 2: ID_Improved vs   MM_Null    Result: 174 to 26
Match 3: ID_Improved vs   MM_Open    Result: 157 to 43
Match 4: ID_Improved vs MM_Improved  Result: 141 to 59
Match 5: ID_Improved vs   AB_Null    Result: 161 to 39
Match 6: ID_Improved vs   AB_Open    Result: 136 to 64
Match 7: ID_Improved vs AB_Improved  Result: 123 to 77


Results:
----------
ID_Improved         76.86%

*************************
 Evaluating: Student
*************************

Playing Matches:
----------
Match 1:   Student   vs   Random     Result: 181 to 19
Match 2:   Student   vs   MM_Null    Result: 168 to 32
Match 3:   Student   vs   MM_Open    Result: 146 to 54
Match 4:   Student   vs MM_Improved  Result: 138 to 62
Match 5:   Student   vs   AB_Null    Result: 156 to 44
Match 6:   Student   vs   AB_Open    Result: 127 to 73
Match 7:   Student   vs AB_Improved  Result: 121 to 79


Results:
----------
Student             74.07%
```

Seeing that the value drops below 75% is not encouraging.

3 Remain Near The Center

Favor moves that position the player closer to the center of the board.

```
centerAt = board_center(game)
playerAt = game.get_player_location(player)

xDiff = playerAt[0] - centerAt[0]
yDiff = playerAt[1] - centerAt[1]
denom = float(xDiff*xDiff + yDiff*yDiff)
if denom == 0:
                return float("+inf")
return float(1.0/denom)
```

```
*************************
 Evaluating: ID_Improved
*************************

Playing Matches:
----------
Match 1: ID_Improved vs   Random     Result: 179 to 21
Match 2: ID_Improved vs   MM_Null    Result: 176 to 24
Match 3: ID_Improved vs   MM_Open    Result: 148 to 52
Match 4: ID_Improved vs MM_Improved  Result: 146 to 54
Match 5: ID_Improved vs   AB_Null    Result: 154 to 46
Match 6: ID_Improved vs   AB_Open    Result: 127 to 73
Match 7: ID_Improved vs AB_Improved  Result: 126 to 74


Results:
----------
ID_Improved         75.43%

*************************
 Evaluating: Student
*************************

Playing Matches:
----------
Match 1:   Student   vs   Random     Result: 180 to 20
Match 2:   Student   vs   MM_Null    Result: 162 to 38
Match 3:   Student   vs   MM_Open    Result: 138 to 62
Match 4:   Student   vs MM_Improved  Result: 131 to 69
Match 5:   Student   vs   AB_Null    Result: 142 to 58
Match 6:   Student   vs   AB_Open    Result: 138 to 62
Match 7:   Student   vs AB_Improved  Result: 127 to 73


Results:
----------
```

```
Student             72.71%
```

This ends up being the most disappointing set of results. This is clearly not a winning strategy.

4 Central Hover Followed By Most Moves

Favor moves that position the player closer to the center of the board until the game board is 40% full. Otherwise, simply move as ID Improved would giving heavy weight on a set of moves that can pinch an opponent.

```
        blanks = game.get_blank_spaces()
        if len(blanks) > (3*game.width*game.height/5):
                centerAt = board_center(game)
                playerAt = game.get_player_location(player)
                xDiff = playerAt[0] - centerAt[0]
                yDiff = playerAt[1] - centerAt[1]
                denom = float(xDiff*xDiff + yDiff*yDiff)
                if denom == 0:
                        return float("+inf")
                return float(1.0/denom)

        playerMoves = game.get_legal_moves(player)
        opponentMoves = game.get_legal_moves(game.get_opponent(player))
        if len(opponentMoves) == 1:
                for o in opponentMoves:
                        for p in playerMoves:
                                if o == p:


        return float(len(game.get_legal_moves(player))-
                len(game.get_legal_moves(game.get_opponent(player))))
```

```
************************
 Evaluating: ID_Improved
************************

Playing Matches:
----------
Match 1: ID_Improved vs   Random     Result: 180 to 20
Match 2: ID_Improved vs   MM_Null    Result: 175 to 25
Match 3: ID_Improved vs   MM_Open    Result: 155 to 45
Match 4: ID_Improved vs MM_Improved  Result: 148 to 52
Match 5: ID_Improved vs   AB_Null    Result: 164 to 36
Match 6: ID_Improved vs   AB_Open    Result: 134 to 66
Match 7: ID_Improved vs AB_Improved  Result: 126 to 74


Results:
----------
```

```
ID_Improved         77.29%

*************************
 Evaluating: Student
*************************

Playing Matches:
----------
Match 1:   Student   vs   Random     Result: 189 to 11
Match 2:   Student   vs   MM_Null    Result: 177 to 23
Match 3:   Student   vs   MM_Open    Result: 159 to 41
Match 4:   Student   vs MM_Improved  Result: 149 to 51
Match 5:   Student   vs   AB_Null    Result: 162 to 38
Match 6:   Student   vs   AB_Open    Result: 136 to 64
Match 7:   Student   vs AB_Improved  Result: 133 to 67


Results:
----------
Student             78.93%
```

This is the first time one of my heuristics has outperformed ID Improved. Considering that this is a variation of ID Improved's heuristic, perhaps I shouldn't be surprised that it is highly competitive.

5 Results

The ID Improved heuristic consistently achieves scores from 75-78% based on testing.

Among all four contenders, "Central Hover Followed by Most Moves" appears to be the best option. By initially remaining near the center of the board, it seems that this should allow for more available moves, permitting more possibility for finding a winning branch. I like the idea of coming up with strategies that highly favor being able to see "horizon" results before they happen, unfortunately I am not sure how to do that in an isolation scenario with chess-like knight movement. As such, highly favoring a single move that includes the possibility of taking the opponent's final move is where my strategy stopped in that regard. Naturally, given that the original ID Improved heuristic was difficult for me to overcome, incorporating it into a more detailed strategy felt like a good approach.

6 Beyond Project Scope - Breaking Changes

The following results were obtained by changing the way that the iterative deepening evaluates layers above depth == 1. If one makes the changes below so that greater than and less than comparisons become strictly greater or strictly less than, the search space is expanded but the win percentages increase. Doing this will result in the agent_test.py failing so I have not included it in my final result, but this was an interesting accidental discovery. This version seems

superior particularly since it gives one of the highest win percentages that any of my testing has yet seen. Both agents ran with the default ID Improved heuristic.

```
# evaluate all branches and return the highest/lowest scoring tuple
for m in legal_moves:
        if current_move == (-1, -1):
                current_move = m

        if maximizing_player:
                ...
                # CHANGED if score >= beta:
                if score > beta:
                ...
        else:
                ...
                # CHANGED if score <= alpha:
                if score < alpha:
                ...
```

```
************************
 Evaluating: ID_Improved
************************

Playing Matches:
----------
Match 1: ID_Improved vs   Random     Result: 184 to 16
Match 2: ID_Improved vs   MM_Null    Result: 174 to 26
Match 3: ID_Improved vs   MM_Open    Result: 158 to 42
Match 4: ID_Improved vs MM_Improved  Result: 142 to 58
Match 5: ID_Improved vs   AB_Null    Result: 162 to 38
Match 6: ID_Improved vs   AB_Open    Result: 146 to 54
Match 7: ID_Improved vs AB_Improved  Result: 140 to 60


Results:
----------
ID_Improved         79.00%

************************
 Evaluating: Student
************************

Playing Matches:
----------
Match 1:   Student   vs   Random     Result: 185 to 15
Match 2:   Student   vs   MM_Null    Result: 178 to 22
Match 3:   Student   vs   MM_Open    Result: 159 to 41
Match 4:   Student   vs MM_Improved  Result: 139 to 61
Match 5:   Student   vs   AB_Null    Result: 157 to 43
```

```
Match 6:   Student   vs   AB_Open    Result: 138 to 62
Match 7:   Student   vs AB_Improved  Result: 128 to 72


Results:
----------
Student             77.43%
```