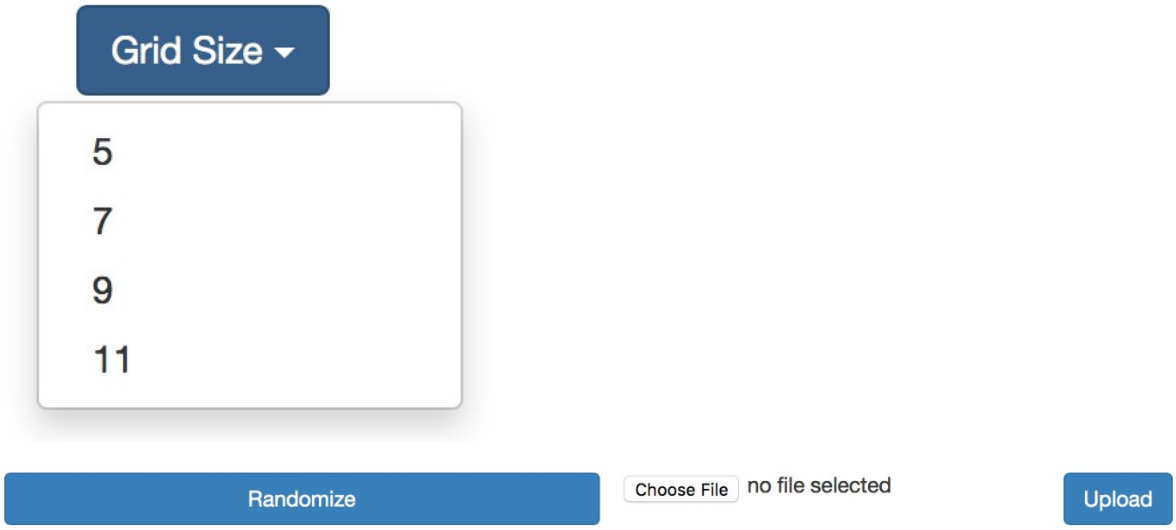
Brandon Smith
Matthew Freeman

CS 440: Assignment 1

Task 1: Puzzle Representation

The structure of the application and GUI are built on Python 3, and Flask which is a microframework for web development. We decided to use python because it's ability to quickly build applications with its simple syntax, and powerful libraries. Building a website for the GUI allows us to ensure display compatibility across systems. Each time the page is loaded a random puzzle is generated based on the chosen dimension in the GUI. The valid sizes for the puzzle are n=5,7,9, or 11, however other sizes could be added if necessary. Each cell in the n x n puzzle has a randomized number 1 to n-1 except the goal cell which is in position [n-1,n-1].



Task 2: Puzzle Evaluation

For this task we use a tree type data structure which holds all the unexplored nodes. We then perform a BFS type analysis on the tree to determine the distances from the root each node is located. To keep track of already visited points an auxiliary array of (1,0) size n x n is kept. As children are added to the queue the distance array is populated until we exhaust the entire tree.

5x5 Puzzles

Brandon Smith
Matthew Freeman

Puzzle Visualizer: Default

Puzzle

1	1	4	3	3
4	3	1	1	1
1	4	4	1	3
3	2	3	3	3
4	2	1	4	G

Puzzle Move Cost

0	1	2	4	3
1	2	4	3	2
X	4	5	4	3
6	5	4	5	4
X	3	3	4	X

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

Solution value: -3

No valid path found

Puzzle Visualizer: Default

Puzzle

3	3	3	1	4
3	4	3	3	1
3	4	3	2	4
1	1	1	1	4
3	4	1	4	G

Puzzle Move Cost

0	4	2	1	2
3	X	X	2	X
2	3	4	3	X
1	2	3	4	5
2	3	4	3	3

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

Solution value: 3

3 Moves: right 3 -> right 1 -> down 4

7x7 Puzzles

Brandon Smith
Matthew Freeman

Puzzle Visualizer: Default

Puzzle

6	5	4	2	4	1	5
5	2	3	5	4	5	2
5	3	6	5	5	3	6
6	6	5	2	4	2	3
6	3	6	2	2	1	6
1	3	1	3	2	1	5
6	2	1	3	1	2	G

Puzzle Move Cost

0	2	X	X	X	X	1
11	9	X	9	X	10	X
X	4	X	7	5	X	X
6	9	X	8	5	9	7
8	8	6	X	9	11	9
7	3	5	6	4	10	2
1	7	6	7	10	11	2

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

Solution value: 2

2 Moves: down 6 -> right 6

Brandon Smith
Matthew Freeman

Puzzle Visualizer: Default

Puzzle

3	3	5	6	2	2	3
6	2	2	6	5	1	4
6	5	6	6	2	6	3
5	5	6	2	6	2	2
5	6	4	1	2	2	4
5	3	2	1	2	4	4
1	2	3	2	4	6	G

Puzzle Move Cost

0	X	5	1	6	4	6
6	X	5	4	4	3	4
X	5	6	7	5	4	6
1	4	6	3	6	2	5
X	4	4	3	4	X	5
6	4	5	4	5	3	5
6	3	X	2	5	3	X

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

Solution value: -7

No valid path found

9x9 Puzzle

Brandon Smith
Matthew Freeman

Puzzle Visualizer: Default

Puzzle

5	4	1	1	6	1	7	6	8
3	8	6	5	2	1	2	5	1
1	1	2	8	1	8	5	3	7
6	5	3	3	8	5	7	3	4
8	4	1	3	4	8	2	5	6
3	4	6	1	1	7	3	7	1
4	2	6	4	7	7	6	6	7
6	1	6	6	4	5	3	2	1
6	2	4	3	6	1	8	7	G

Puzzle Move Cost

0	6	5	6	2	1	2	5	5
3	4	4	4	3	2	3	5	4
2	3	4	4	5	3	5	X	5
3	4	6	X	4	7	4	X	X
4	6	5	3	4	7	4	X	5
1	X	3	2	3	4	X	5	4
6	5	7	3	3	X	5	4	5
9	X	5	4	X	8	3	7	6
2	5	X	6	5	8	3	X	6

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

Solution value: 6

6 Moves: right 5 -> down 1 -> right 1 -> right 2 -> up 1 -> down 8

Brandon Smith
Matthew Freeman

Puzzle Visualizer: Default

Puzzle

6	3	6	2	1	5	7	7	1
7	5	4	4	6	5	5	6	5
5	8	5	2	1	8	3	1	3
3	8	7	2	7	3	7	6	3
6	4	3	3	7	3	7	2	6
5	5	2	5	3	6	4	5	7
6	6	8	1	4	2	8	1	4
3	5	6	3	2	1	3	5	6
7	5	8	4	1	8	5	5	G

Puzzle Move Cost

0	8	16	7	9	8	1	11	17
X	7	9	5	10	X	8	6	10
X	9	X	8	12	9	9	8	9
X	9	11	X	13	X	X	9	X
5	X	X	4	X	X	3	X	X
11	8	10	6	11	9	9	10	7
1	8	17	X	11	X	2	9	10
4	X	8	3	11	X	2	7	9
X	X	15	13	12	13	X	14	X

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

Solution value: -23

No valid path found

11x11 Puzzle Size

Brandon Smith
Matthew Freeman

Puzzle Visualizer: Default

Puzzle

2	8	10	5	5	2	8	2	10	4	6
9	10	4	7	2	7	9	5	1	7	3
6	3	4	4	5	10	4	4	6	8	4
7	3	9	8	1	10	3	10	6	2	6
4	7	10	1	9	2	8	7	6	9	8
4	7	5	10	10	5	3	4	7	5	3
6	4	1	10	2	1	3	9	5	7	7
3	5	6	8	2	5	9	6	8	9	4
7	3	4	2	4	5	4	8	5	4	3
5	10	6	5	5	4	6	10	7	5	4
9	8	4	8	2	10	7	2	1	1	G

Puzzle Move Cost

0	X	1	6	X	5	5	4	7	5	X
3	X	5	X	X	X	6	6	X	4	X
1	5	3	6	6	6	2	5	8	7	3
X	X	10	5	9	7	4	6	9	5	8
6	X	8	X	7	X	7	X	X	6	X
7	6	4	6	7	6	X	5	7	6	7
X	4	3	4	6	5	3	6	X	4	4
6	X	4	7	7	6	8	X	5	X	7
2	10	7	9	7	6	6	3	8	5	7
5	7	X	X	7	6	4	6	7	6	9
4	5	2	10	X	7	3	7	6	5	6

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

Solution value: 6

6 Moves: down 2 -> down 6 -> up 7 -> down 9 -> right 9 -> right 1

Brandon Smith
Matthew Freeman

Puzzle Visualizer: Default

Puzzle

2	8	8	9	5	10	5	10	8	10	1
3	1	5	8	3	10	4	6	2	5	7
3	8	5	8	3	1	7	4	8	3	3
1	5	10	10	9	8	7	8	5	5	1
10	8	3	7	2	1	1	2	3	3	8
10	10	3	6	3	2	1	2	8	4	9
7	7	7	4	6	8	4	8	3	10	5
1	6	2	2	3	3	8	7	7	3	4
9	8	4	1	2	9	5	8	3	2	7
9	8	1	4	1	8	5	2	7	4	9
3	1	1	2	1	7	7	6	6	9	G

Puzzle Move Cost

0	5	1	X	7	X	X	7	8	3	2
7	6	4	4	6	8	X	5	X	5	3
1	7	6	2	6	X	8	7	X	8	X
X	X	X	X	X	5	4	7	X	X	X
8	X	3	X	5	4	5	6	X	7	9
2	4	5	6	7	5	6	6	10	7	3
6	X	5	X	6	8	7	7	X	6	7
5	6	4	5	5	6	9	6	7	8	X
6	4	2	4	5	10	3	6	9	5	4
X	5	5	5	7	7	9	6	X	6	X
5	4	5	3	6	4	5	8	X	4	X

Solution Visualizer

Default | Hill Climbing - Basic | Hill Climbing - Random Restarts | Hill Climbing - Random Walk | Simulated Annealing | Population Based

Puzzle Solution

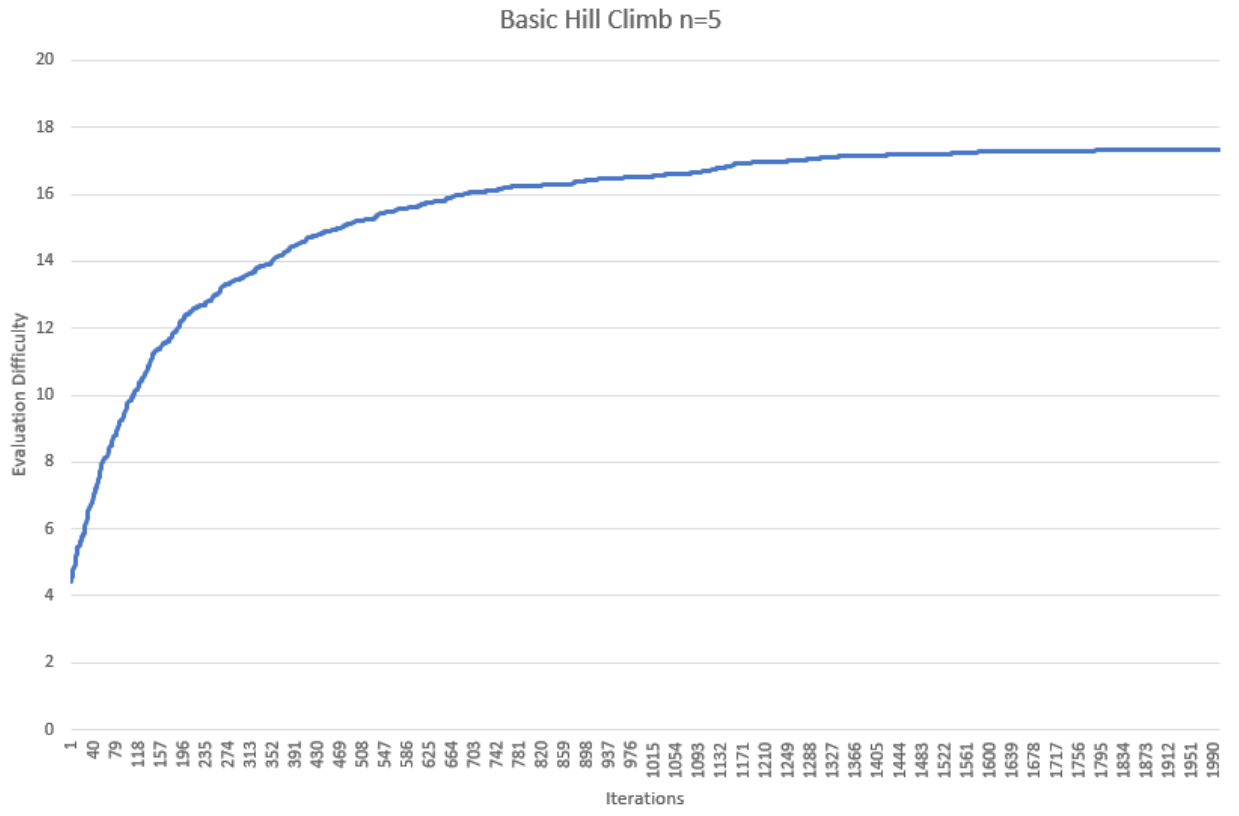
Solution value: -28

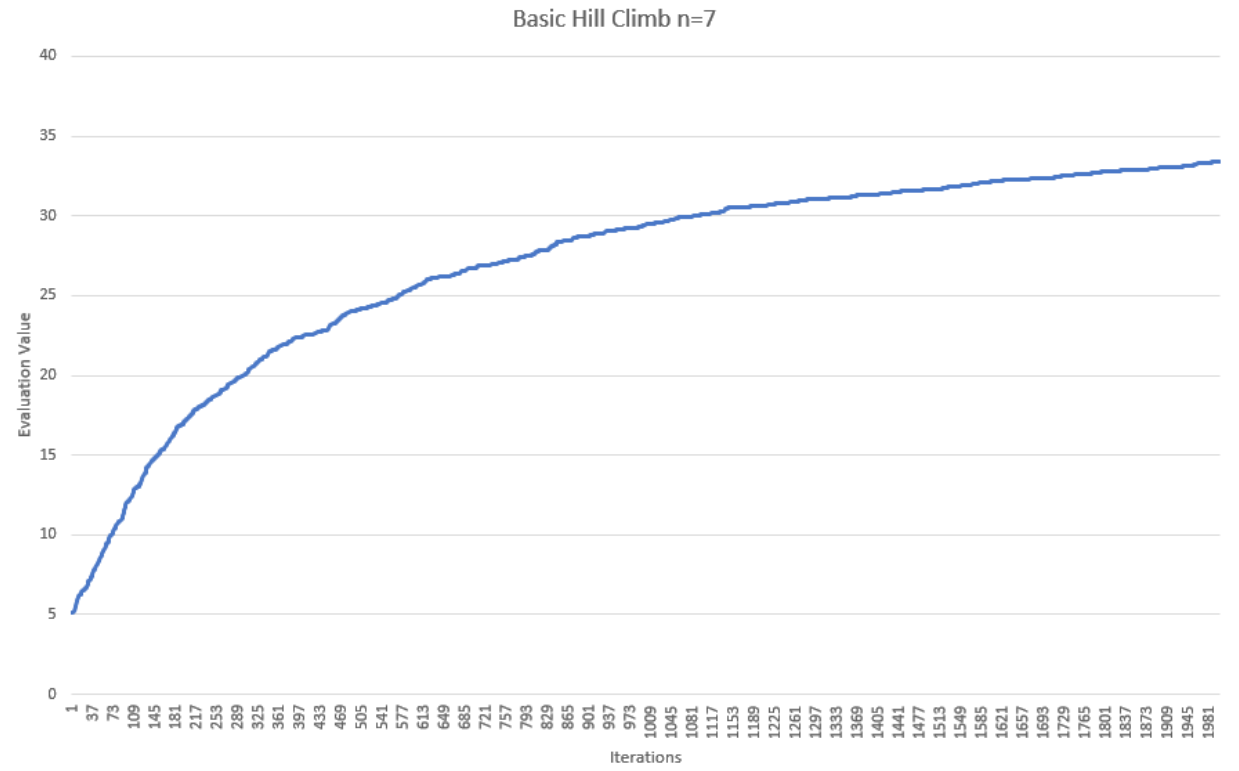
No valid path found

Task 3: Hill Climb Basic

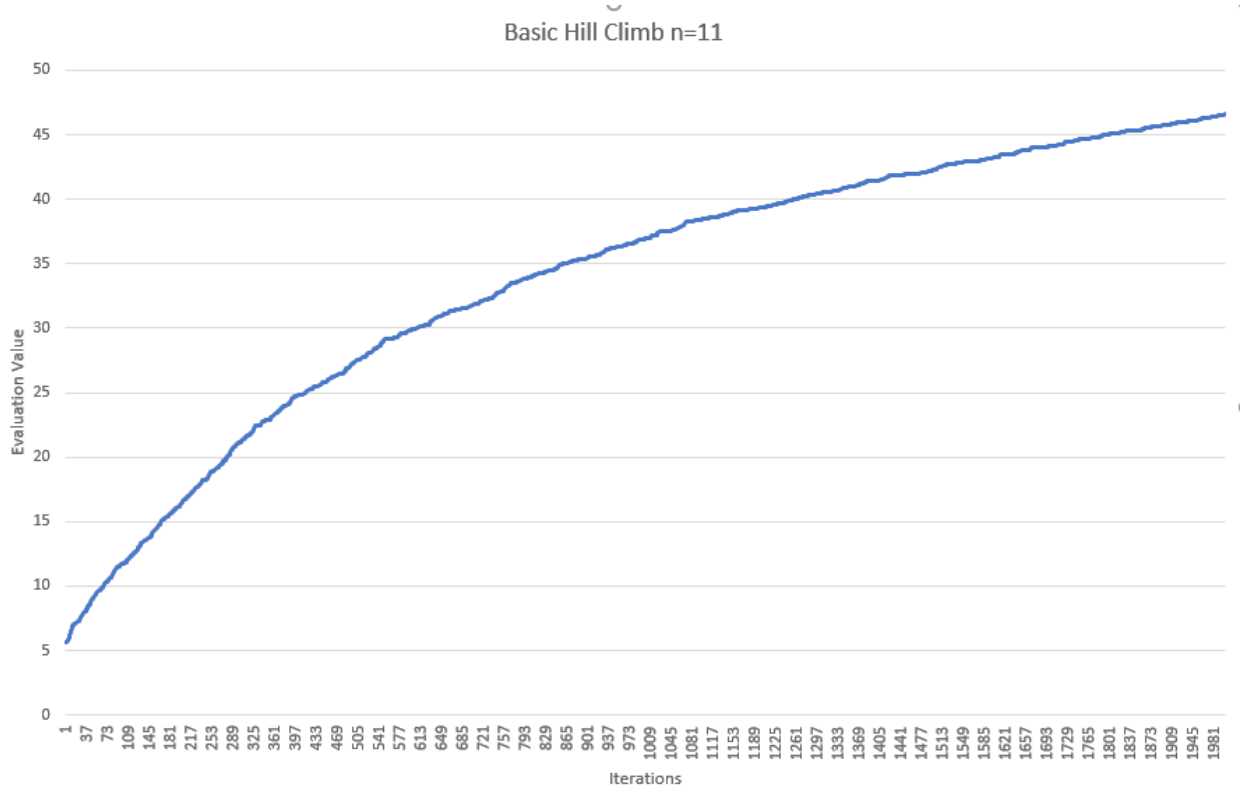
With basic hill climbing I used an average data over a total of 2400 runs, each 2000 iterations each. I used 40 random grids, and performed basic hill climbing 60 times per grid. I used the output at each iteration and averaged all the values, then plotted it. With n=5 it took about 2 minutes, n=7 took about 3, n=9 took about 5 minutes, and n=11 took about 7 minutes to complete.

Brandon Smith
Matthew Freeman





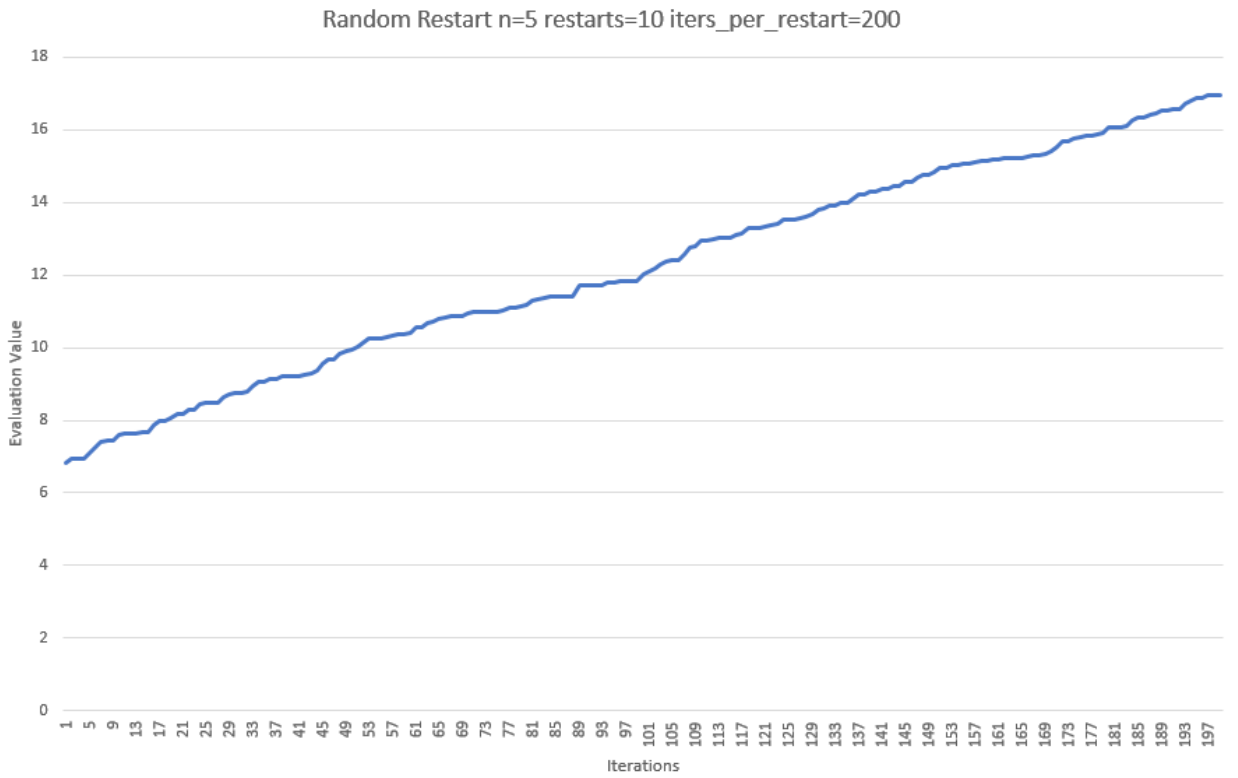
Brandon Smith
Matthew Freeman

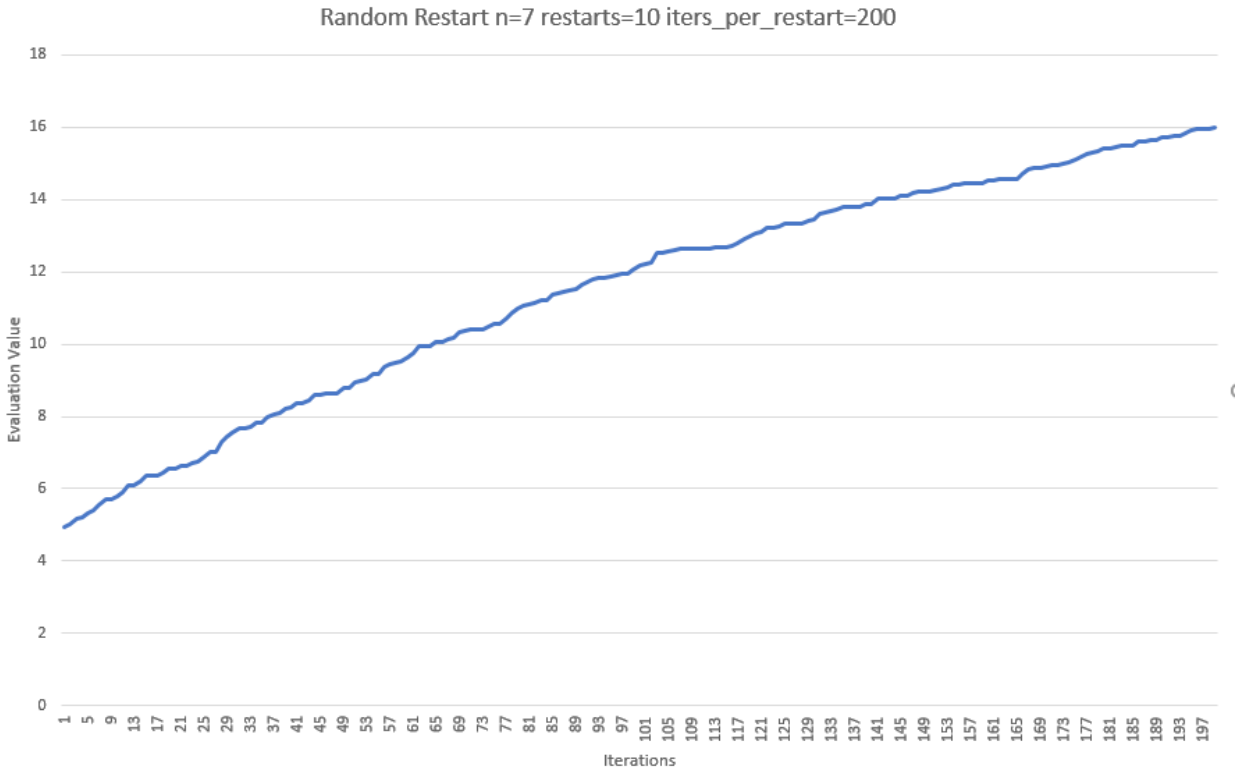


Task 4: Random Restart

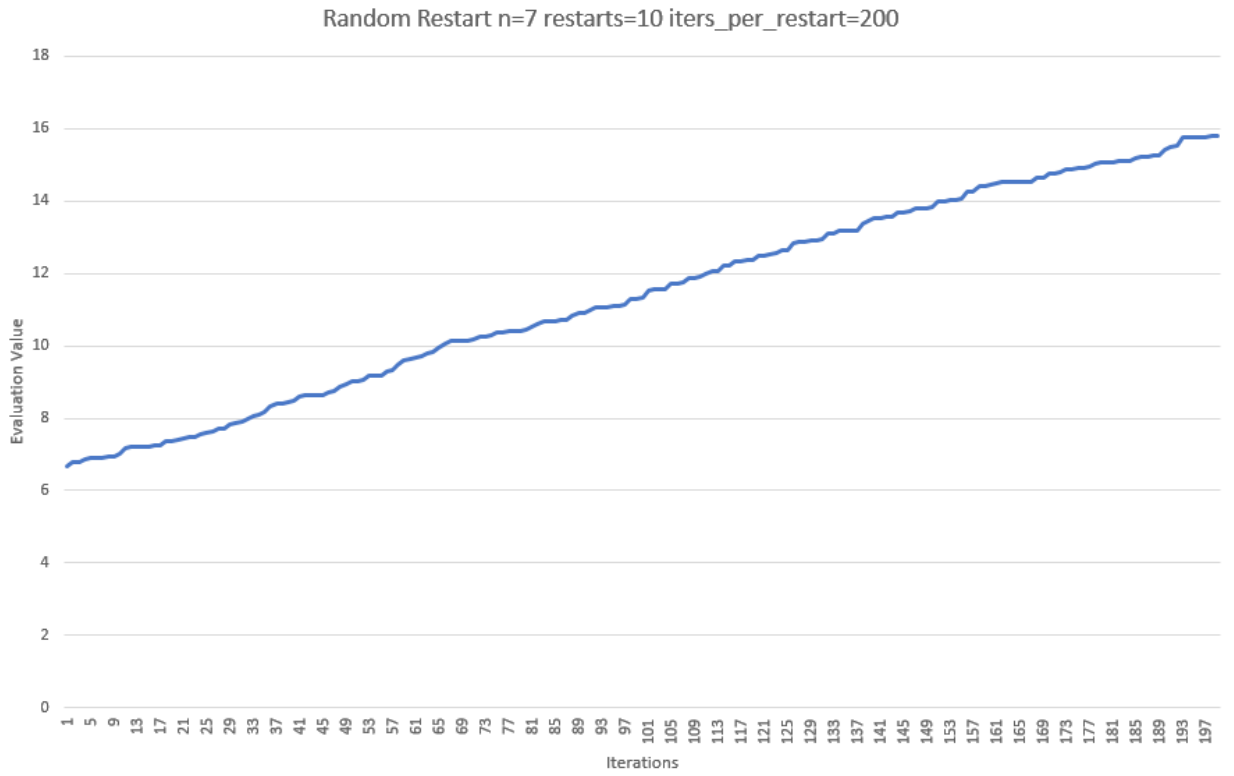
With random restart I used an average data over a total of 2400 runs, each 2000 iterations each. I used 40 random grids, and performed basic hill climbing 60 times per grid. To get 2000 iterations I made the number of restarts 10, and the number of iterations per restart 200. I found this most optimal to increase the best found at higher internal iterations. I used the output at each iteration and averaged all the values, then plotted it. With n=5 it took about 2 minutes, n=7 took about 3, n=9 took about 5 minutes, and n=11 took about 7 minutes to complete. Random restart seems worse at a fixed iteration count compared to basic, but I believe at higher computation time this method could come out on top. One reason for the lower overall is because we don't get to explore higher iterations, and when we are done with the current set we discard the puzzle, if its not the best and move to a completely random puzzle which has a high probability of being worse than the maximum.

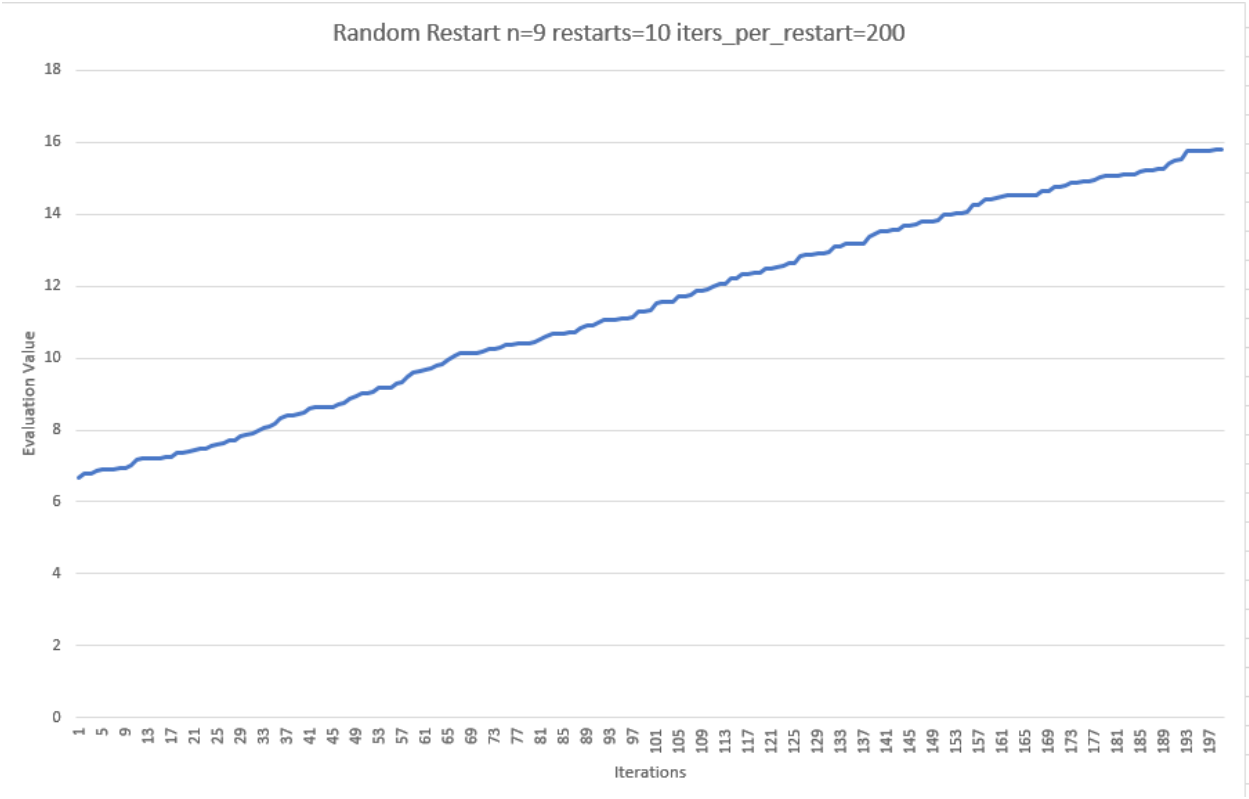
Brandon Smith
Matthew Freeman



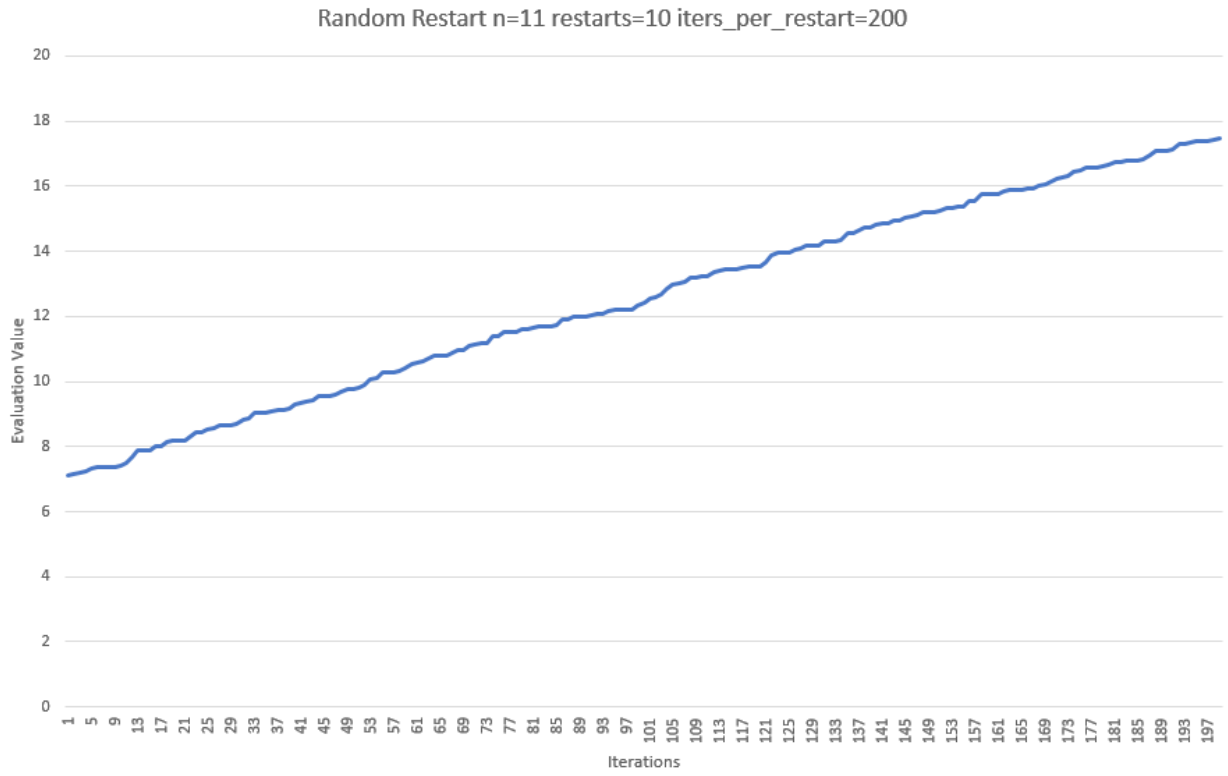


Brandon Smith
Matthew Freeman





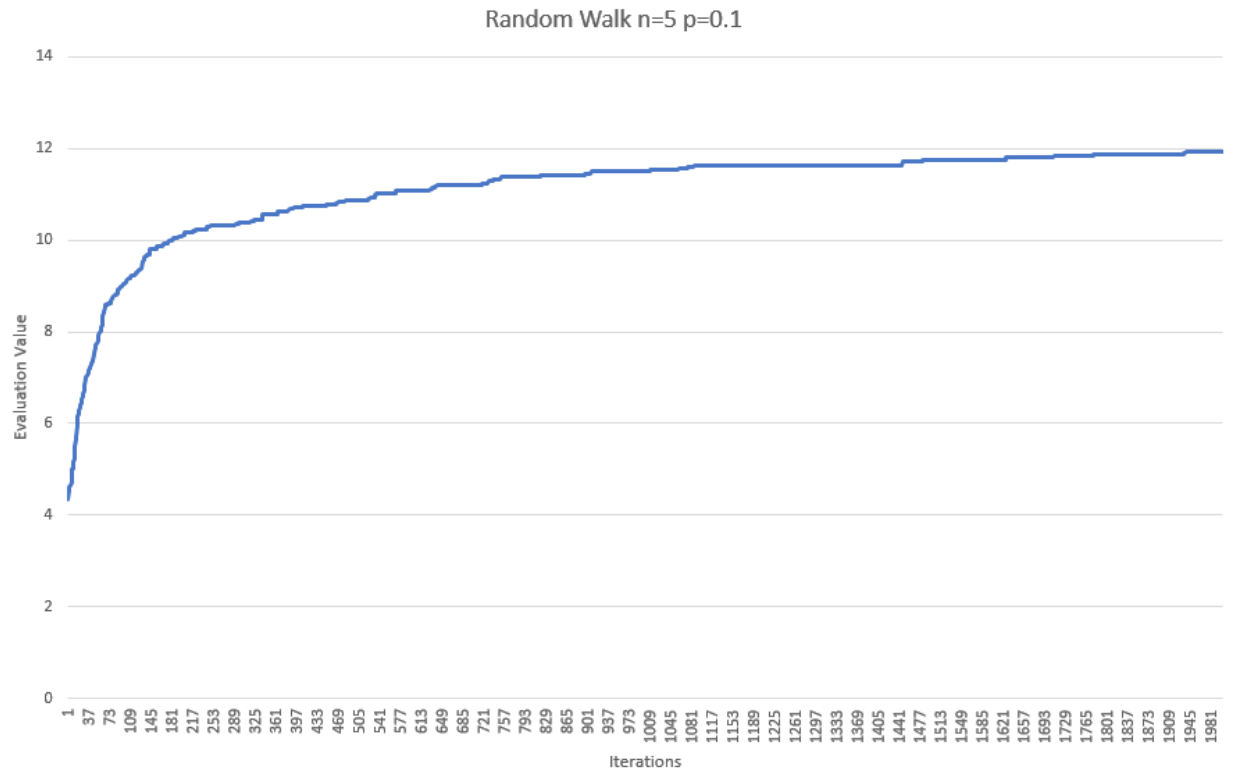
Brandon Smith
Matthew Freeman

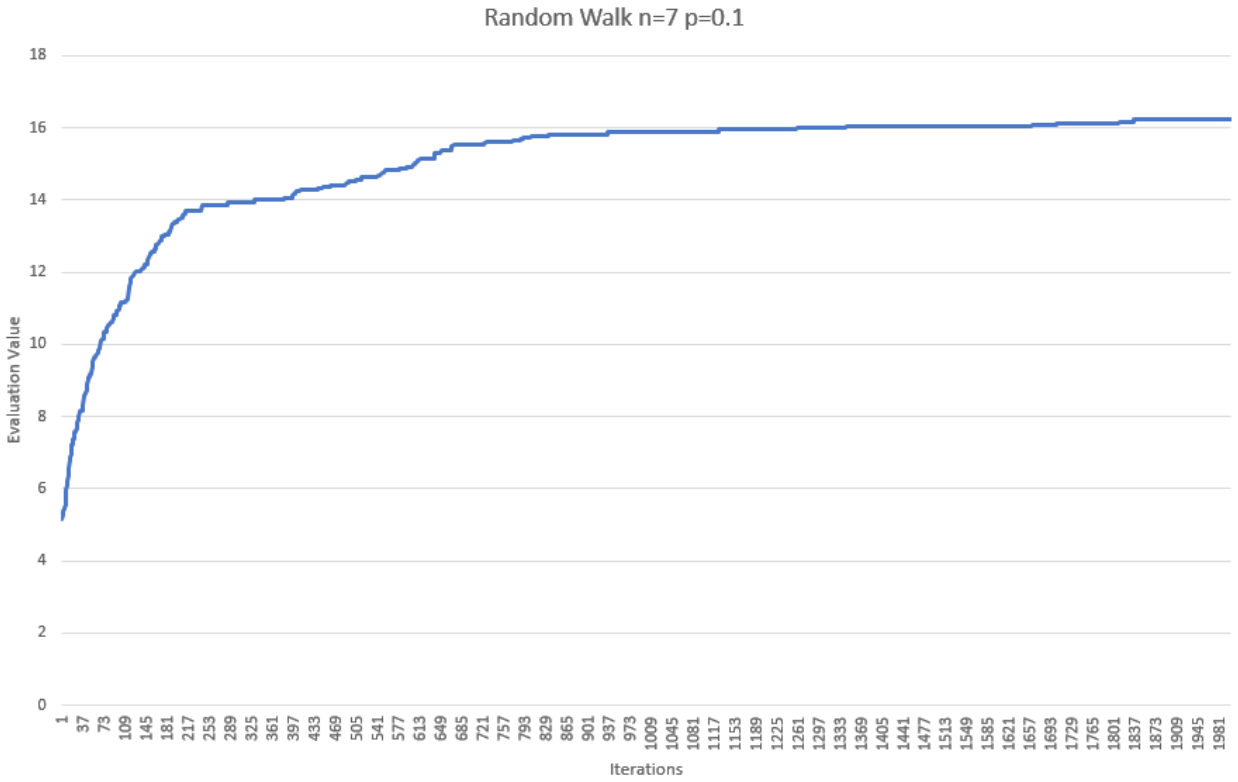


Task 5: Random Walk

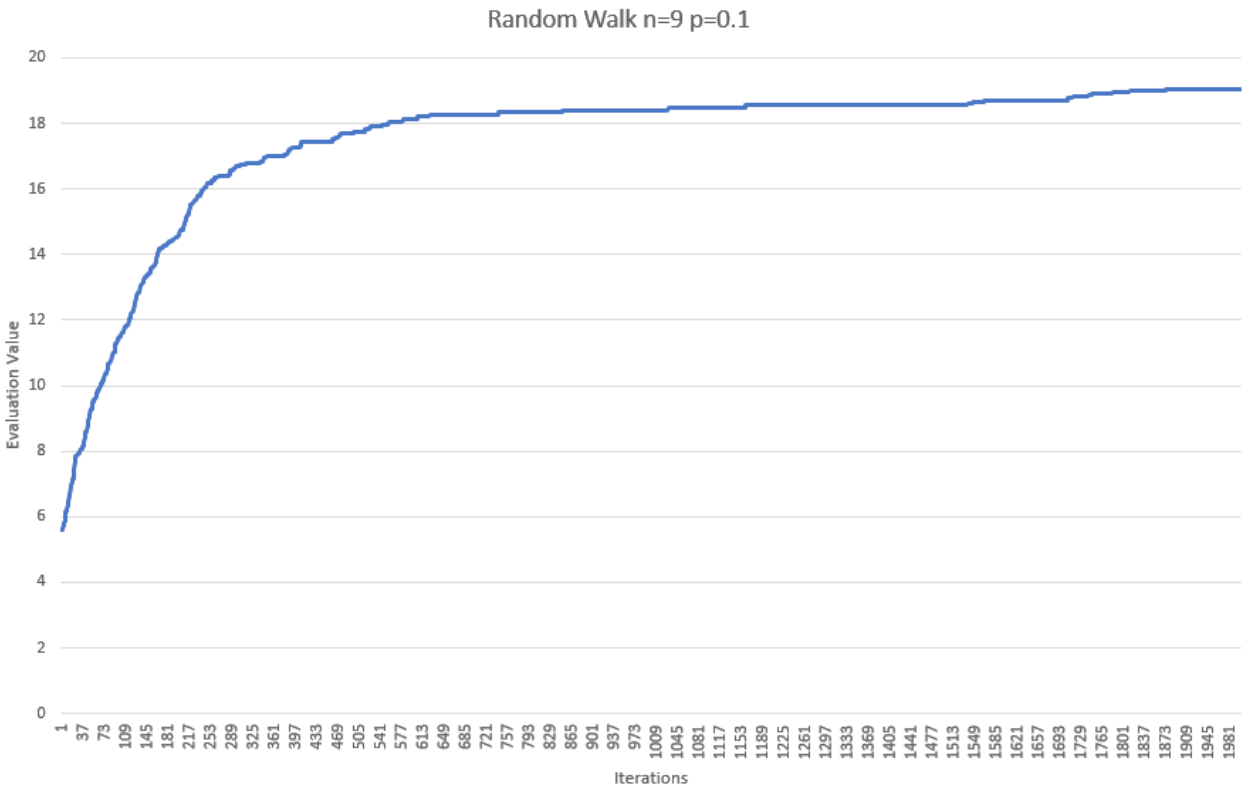
With basic hill climbing I used an average data over a total of 2400 runs, each 2000 iterations each. I used 40 random grids, and performed basic hill climbing 60 times per grid. I used the output at each iteration and averaged all the values, then plotted it. I found that when p=0.1 it produced the best results. When I made the value higher the average started to decrease because I suspect that we are accepting downhill slopes too much and creating a bad trend. With n=5 it took about 2 minutes, n=7 took about 3, n=9 took about 5 minutes, and n=11 took about 7 minutes to complete. Random walk seems better than random restart, but worse than basic hill climbing. I believe that random walk can help find more maxima over a long period of time, however my compute time was limited and could only do 2000 iterations. Random walk helps when we are stuck at a local max, but more may exist. I choose a high probability to show that the random walk can have an adverse affect on the average. A value of 0.1 may seem low, but we got half of what we expected with pure hill climbing. A value of 0.001 produced very similar results to pure hill climbing which is why I decided to show when the probability is 0.1

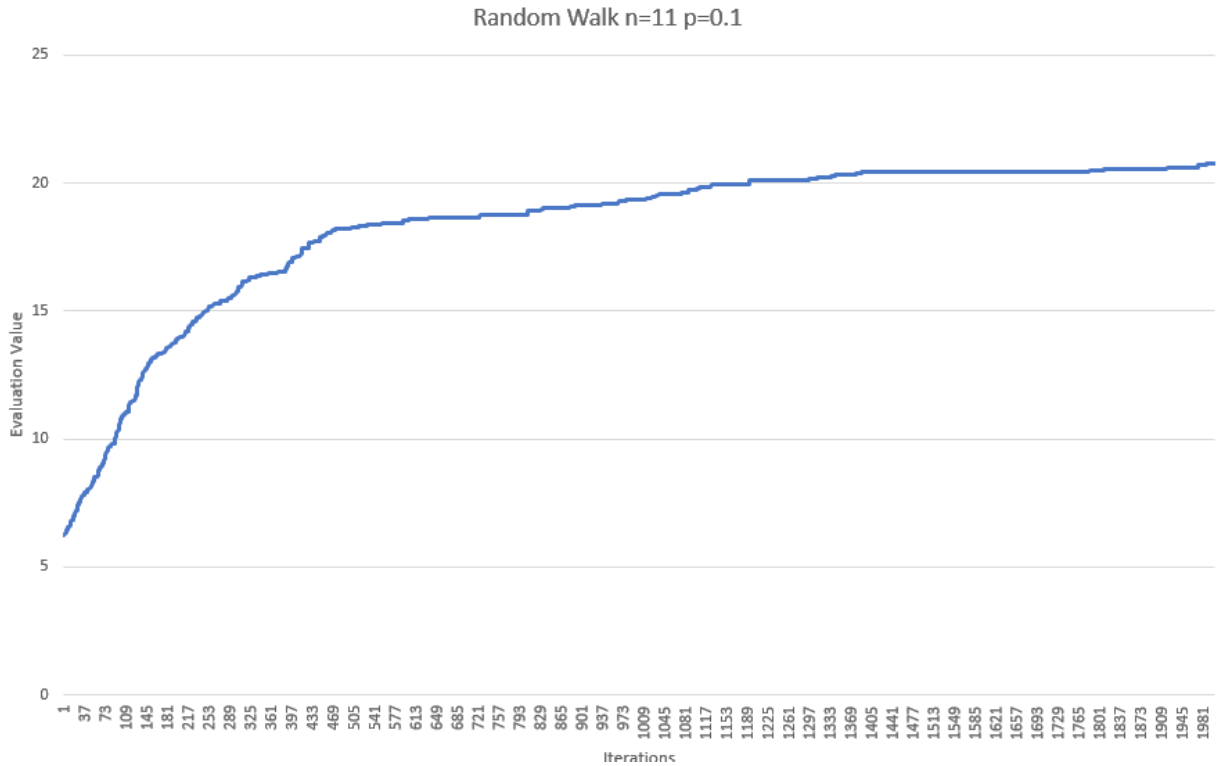
Brandon Smith
Matthew Freeman





Brandon Smith
Matthew Freeman

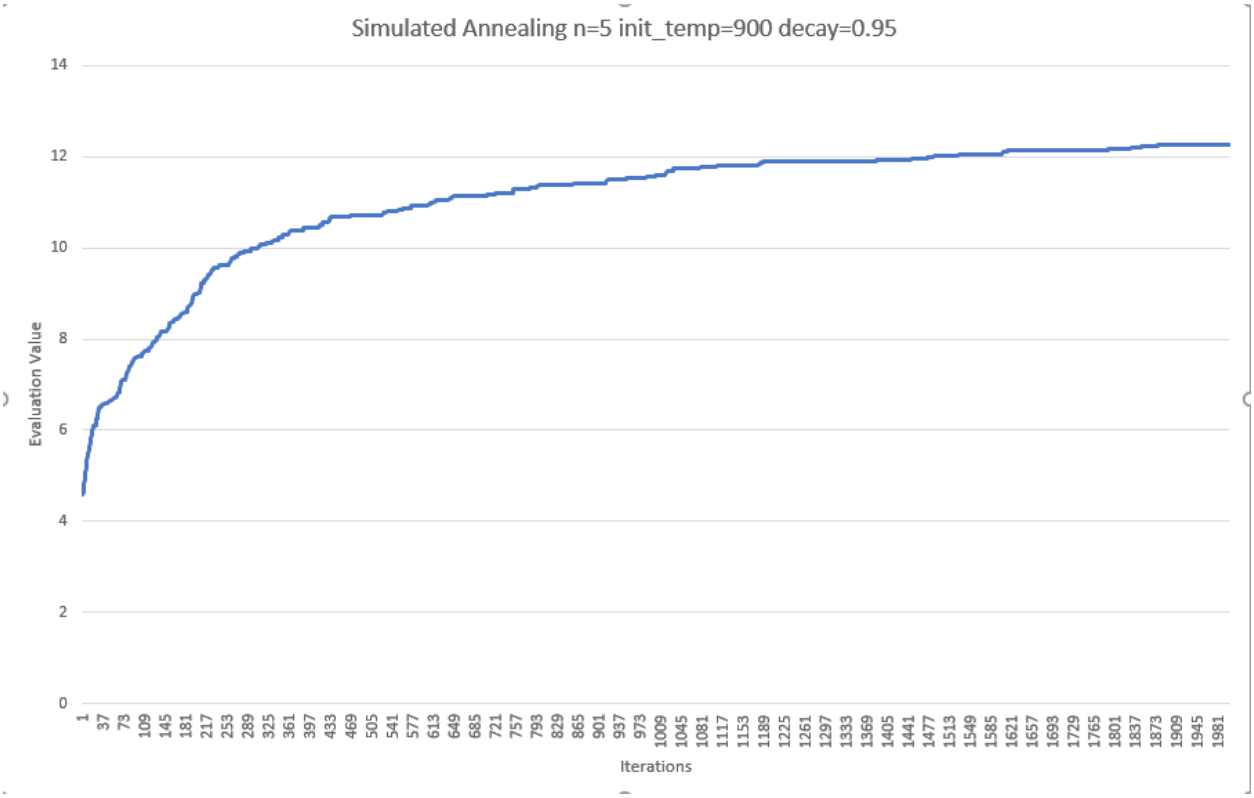




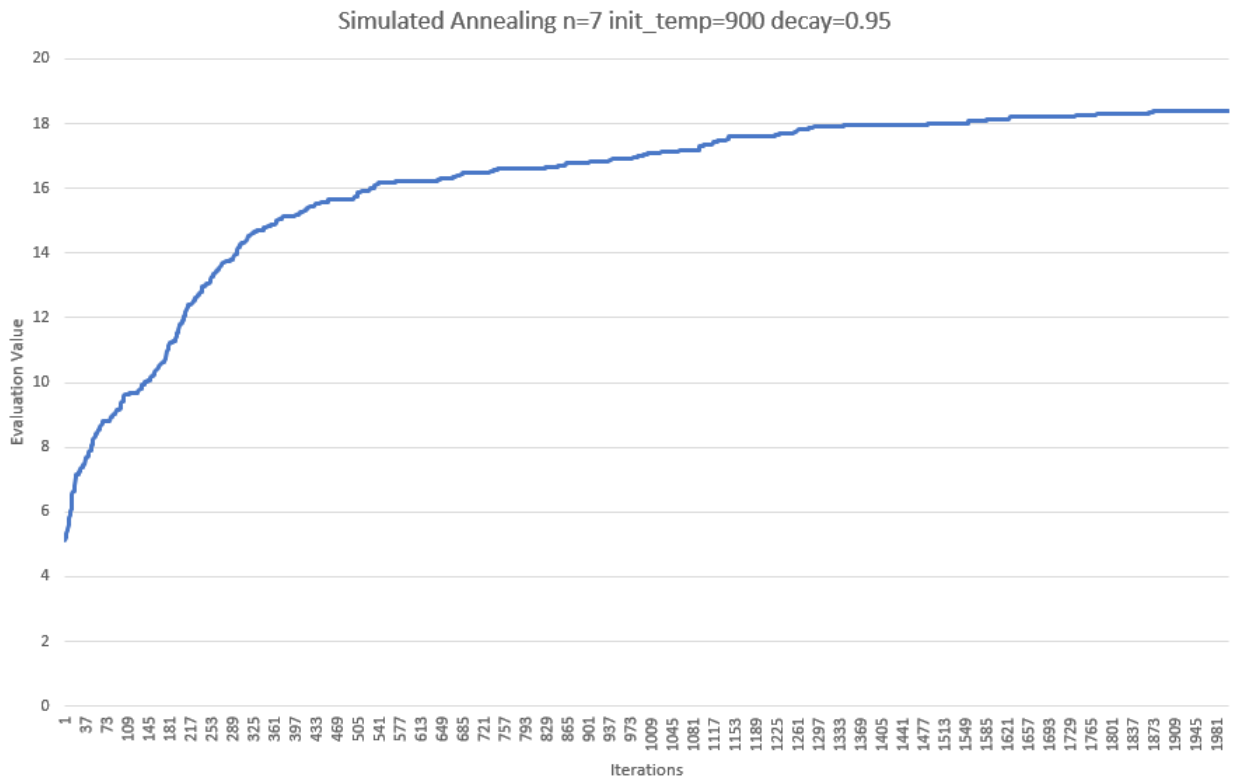
Brandon Smith
Matthew Freeman

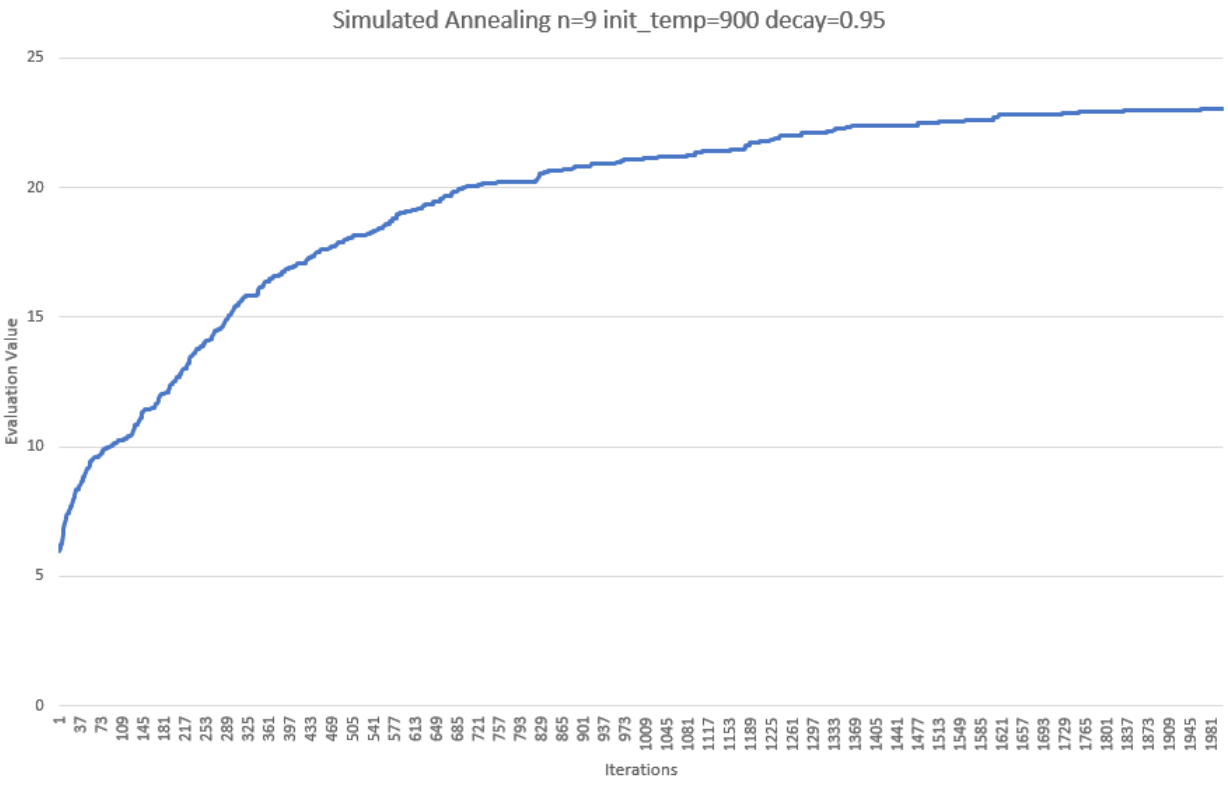
Task 6: Simulated Annealing

With basic hill climbing I used an average data over a total of 2400 runs, each 2000 iterations each. I used 40 random grids, and performed basic hill climbing 60 times per grid. I used the output at each iteration and averaged all the values, then plotted it. I used an initial temperature of about 900 and decay as 0.95. If the value of initial_temp is too low the probability that we accept a downhill trend is very low and we may only find the local maxima. This will be similar to just pure hill climbing. With the decay value at a high 0.95 we ensure that the temperature does not decay too fast. I found these to be the optimal values. With n=5 it took about 2 minutes, n=7 took about 3, n=9 took about 5 minutes, and n=11 took about 7 minutes to complete. Compared to the previous three methods of search this seems more efficient than random restart, and random walk, but not the basic hill climbing. I believe that the changing acceptance probability enables quickly finding more local maxima and when we are at high iterations we keep on the current local maxima thus optimizing which point we end on.

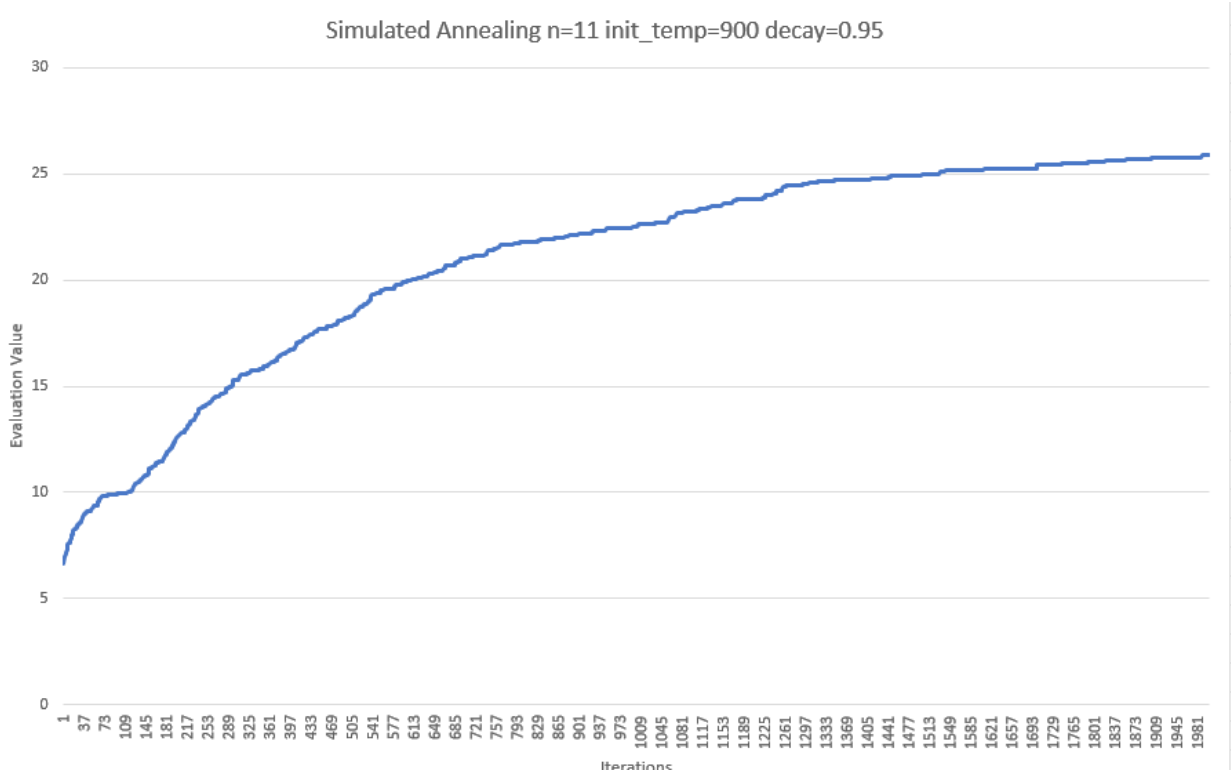


Brandon Smith
Matthew Freeman





Brandon Smith
Matthew Freeman



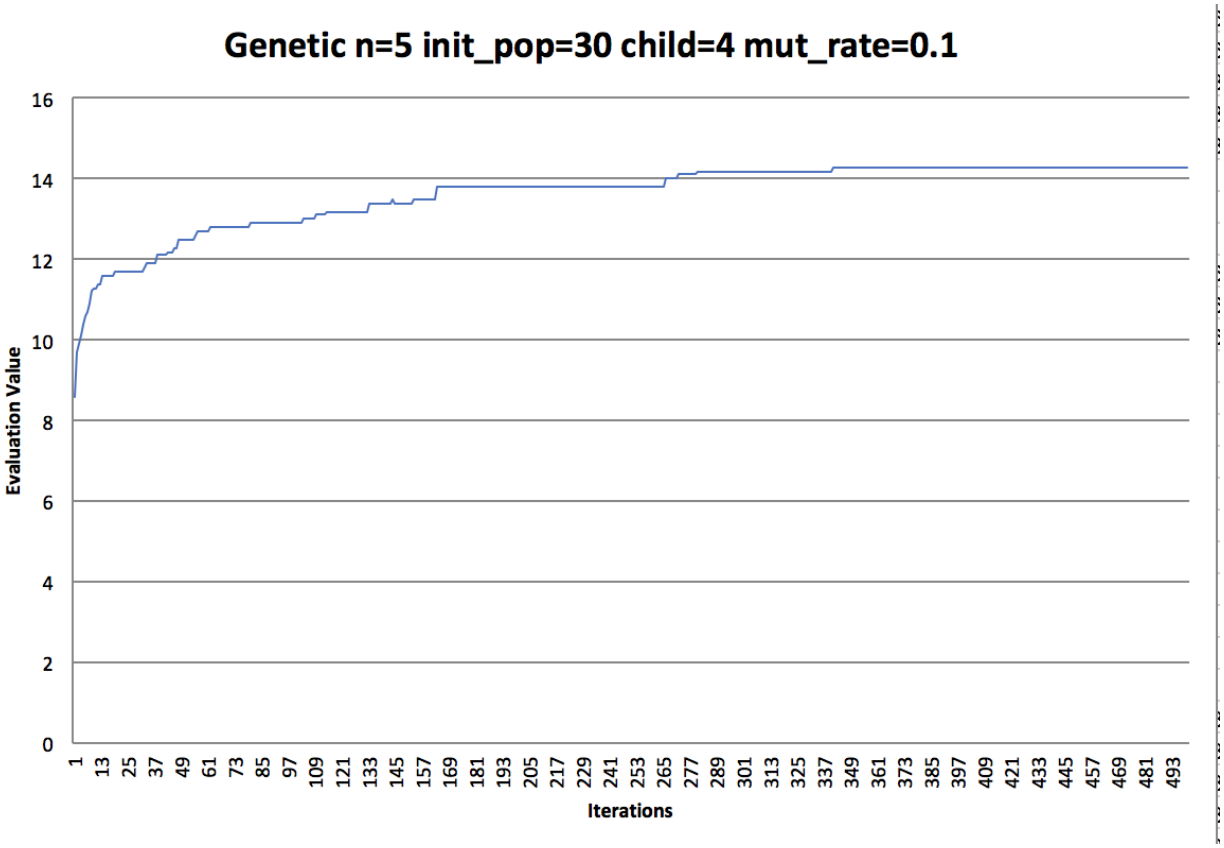
Task 7: Genetic Algorithm

I implemented a genetic based population search method for this task. I followed a pretty standard approach to this method. Each iteration is a generation. Beginning this algorithm we define an initial population, which will be the maximum amount allowed in the population. The initial population will be all randomly generated and then sorted by maximum fitness value. The fitness value is the evaluation value for each puzzle. After the evaluation value is determined and all puzzles sorted we go through all the population and cross-over them to create 2 new children. The cross-over section is a randomly chosen chunk that is switched between the parents, and the result are the new children. However there is another parameter which is number of children(pairs). So 2 parents can have multiple pairs of children. After we cross every parents we have more children than the population can hold, so we only choose the top x children, where x is defined as the number we specified as the initial population. Now these children will be used as input for the next generation. During cross-over we can also specify a mutation rate. The mutation rate has a probability of altering 1 gene when crossing over.

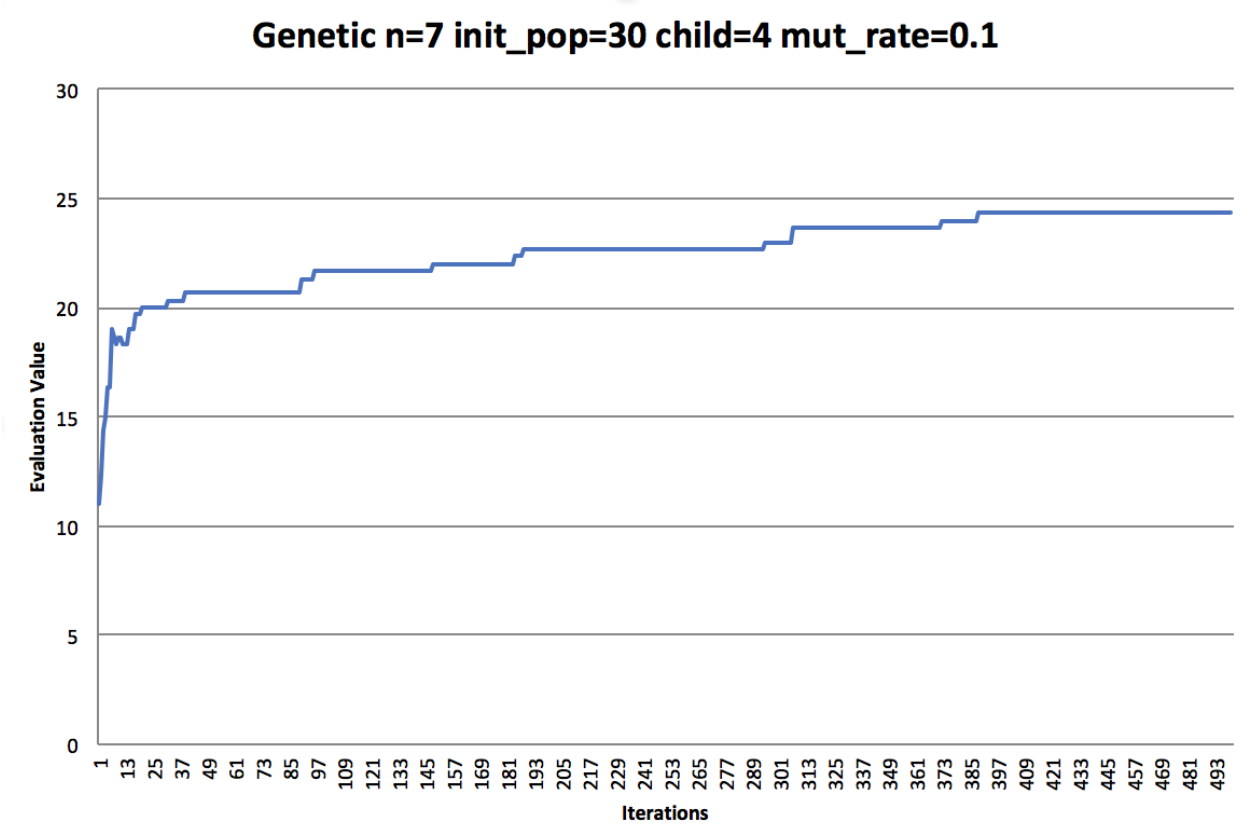
During my analysis of choosing different values for the variables, I came across these optimal values. The initial population is 30, children pairs is 4, and the mutation rate is 0.1. All over 300 generations. Changing the initial population had the greatest effect on the best values. This is most likely because we have a bigger gene pool. The drawback is however the computation time is very adversely affected with a higher number of initial population. To keep a fair comparison to the other methods of searching I limited it to 30. The number of children pairs has the same performance impact, with minimal affect. Anything higher than 4 had virtually no affect

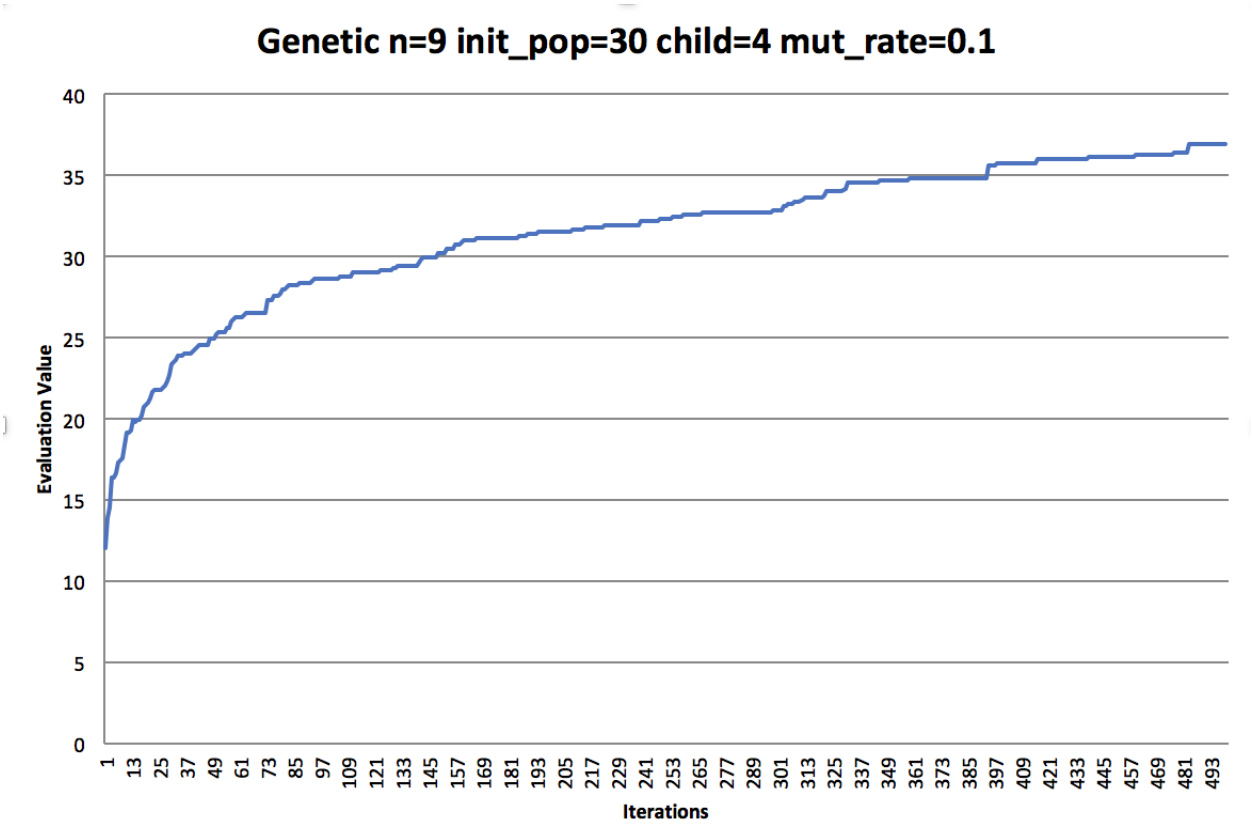
on the average value. Mutation rate surprisingly had a large effect on the average values. If we had a large mutation rate the average was lower. This Is probably similar to the random walk where we are going towards a downhill slope, however in this case we do not know if it's downhill.

For the generated charts I used the above configuration to match the previous 4 methods of searching. They each took the same amount of time compared to their local search counter parts eg. 2 3 5 7 mins. Surprisingly I get a little better results than the non basic hill climbing methods. I ran each grid size 50 times.



Brandon Smith
Matthew Freeman





Brandon Smith
Matthew Freeman