Project 2: Local Search
CS 4200
Thomas Landry

Approach:
I was only able to implement the genetic search algorithm. I did so by using a one dimensional array as set out by the project parameters. The algorithm itself is extremely easy to put together and with testing I was able to solve for n = 100 in a relatively short amount of time. I capped the algorithm by max iteration instead of time just for simplicity's sake in the code. The code itself is written to be versatile for any problem that fits or can be redefined by a single dimensional array as it takes the heuristic as a lambda expression.

Data:

Genetic Algorithm for Problems of n = 25

Top x of Population Sampled	Rate of Mutation	Maximum Generation per solution	Children Created per Iteration	Success Rate	Average Time to Solution (nano)
10	30	5000	120	82%	330588187
30	30	5000	120	87.2%	545563797
30	30	4000	120	80.6%	447041690
30	40	5000	120	93.4%	542628651
30	50	5000	120	98.0%	415969681
40	50	5000	120	97.0%	495664912

Observations:
Finding the right balance of parameters for the genetic algorithm is likely the most time consuming part of implementing this search strategy. Making small adjustments to the number of population sampled from or the time the algorithm is allowed to run can make huge differences with success rate. I started with a 20 percent mutation rate but found that increasing it to 30 had a pretty large positive effect on the number of successful solutions. Further increasing the mutation rate had a massive impact on success rate and time to find a solution, emphasizing how important diversity is in finding a quick solution to the problem. However too much diversity in the children seems to increase the time to solution and slightly detract from the accuracy of the algorithm.

Conclusion:

Out of all the parameters, Mutation rate and the number of generations seem to be the biggest factors in how successful the algorithm is in finding a solution. The algorithm is an interesting take on the idea of guessing the answer and checking the result. I think incomplete algorithms like these are the most interesting things coming out of Artificial Intelligences because they really seems to highlight the ‘out of the box’ thinking that a lot of cutting edge CS is focussed on now a days.