# Algorithm Performance Evaluation

The various algorithms that we implemented were evaluated against the following metrics:

1. Execution time
2. Solution quality

The performances were then visualized in a bar chart and table respectively.

## Execution Time

This metric calculates the time taken in seconds for an algorithm to execute. This measurement when the same type of problem is passed into the various algorithms, for example, getting the best path to any base station from node 43 will be passed in to all the algorithms. The time library was imported to capture execution start time and end time and this the execution time calculated as follows:

execution time = execution stop time – execution start time

This information is recorded against the algorithm name in a dictionary. After all the executions, the dictionary is used to build a bar chart which shows execution time in seconds as in the figure below.

A graph with a bar

Description automatically generated with medium confidence

## Solution Quality

The best paths from a sensor to a base station may vary for various algorithms. These paths may have varying absolute costs or similar absolute costs. This metric analyses the best paths in terms of the absolute cost and ranks them such that those that have a high absolute cost are ranked higher. But this metric being done once may not reflect the consistent generation of quality solutions. To take care of this, we set the algorithm to run several problems and for each problem rank them. At the end, ranks for each of the problems are ranked to come up with an aggregated rank of the best solution which are represented in a table as shown below.

A table with numbers and lines

Description automatically generated with medium confidence

In the figure above, Dijkstra and simulated annealing got the best solutions and tied in terms of their performance ranking.

## Evaluation

After running several tests, we had the following results for our algorithms:

1. Execution time metric:
2. Dijkstra’s Algorithm (< 1 second)
3. Simulated Annealing Algorithm (1-5 seconds)
4. Discrete Genetic Algorithm (20 – 60seconds)
5. Ant’s Colony Optimization Algorithm (200-250 seconds)

The indicated times ranges are where most execution times lie for round of algorithm execution. They vary based on the kind of randomly generated solutions in the cases of the stochastic algorithms like Simulated Annealing and Discrete Genetic Algorithm. These times also vary based on the \*\*kwargs provided e.g. the population size in DGA or the number of ants in ACO. These performances were run against the default parameters which had been pre-tested to ensure optimal output.

Based on the results, Dijkstra’s Algorithm is the best algorithm for this problem in terms of time since executes in the shortest time consistently.

1. Solution quality metric:
2. Dijkstra’s Algorithm
3. Simulated Annealing
4. Discrete Genetic Algorithm
5. Ant Colony Optimization

The ranks were done based on 4 trials for each algorithm, and for each trial, the best path from a particular node was tried for each algorithm. There were times when DGA emerged the best because it is dependent on the randomly generated initial paths and the permutation with is more of a trial and error. The above ranks were derived from several executions and the performance consolidated to the above.

Based on the results, Dijkstra’s Algorithm performed best for this problem because it consistently gave good results. The upside of Dijkstra’s algorithm is the logic of evaluating the next best path in the trail. This made the algorithm always get to the base station from a node in a particular route even when run several times. This consistency makes it a reliable algorithm for good solutions alongside the quick execution time as discussed above.

## Tradeoff between time and quality

These rankings were based on several tests but again they vary based on the \*\*kwargs provided. At the end, for stochastic algorithms, there is a tradeoff between time and quality. For example, if you initialize many ants in the ACO algorithm, more time will be used to explore the search trail and therefore to get the best path and you might get better solutions.