**ECM3412 (Nature-Inspired Computation) Coursework**

**Experimental Design**

In this coursework, I experimented on the performance of Ant Colony Optimisation by altering its parameters one-by-one, to find the optimal setup to the Travelling Salesman Problem.

The parameters I considered were:

* Colony Population (Number of Ants)
* Number of iterations
* Pheromone Evaporation Rate
* Algorithm Variation (Normal, Elitist and MMAS – Max-Min Ant System)
* Heuristic Functions

In this experiment, I have identified the main variables as follows:

* Performance (Global Minimum Cost): Dependent variable
* The parameters (Above): Independent variables
* Comparison algorithm: Control variable

For the comparison algorithm, I used a simple greedy search as a basis for expected performance. With this, any output from ant colony optimisation worse than the greedy search output is an indicator of bad performance.

I used grid search to tune each integer and float parameter, through the Program class. The process is as follows:

1. Runs the given algorithm a given number of times.
2. Calculates and outputs performance metrics from these algorithm runs (overall best path found and average path found).
3. Increases the parameter value by a step, determined by the difference between the bounds, divided by the given number of steps.
4. If this is the [number of steps + 1]th iteration, move onto the next parameter. Otherwise, go to step 1.

Every other parameter can be tuned by taking every possible value it can take.

I ran each experiment with the parameters in the code, meaning 16 parallel algorithms, and 10 experiment steps.

Additionally, each experiment was performed twice, once for the Burma file and once for the Brazil file. This is to account for different input files having different optimal parameters.

Furthermore, I obtained the optimal solutions for Burma and Brazil from the TSPLIB website[[1]](#endnote-1), which are *3323* and *25395* respectively. I will use these values to determine how close each algorithm run comes to optimality.

**Experiments**

Number of Ants

For the Burma file, the algorithm found 3323 as the best located path by any of the algorithm runs, regardless of the number of ants. The average quality of each solution, however, shows a trend of improving as the number of ants increases. This improvement appears to be of a reverse-exponential shape (e-x), meaning this improvement gives diminishing returns. Furthermore, increasing this parameter much further is impractical - execution time is directly proportional to the number of ants, and surpassing 1,000 ants with 10,000 iterations means over 107 pathfinding loops.

The algorithm had similar findings for the Brazil file regarding average cost, except the best cost also followed the same relationship. For Burma (a smaller solution space), increasing the ant population didn’t seem to give much benefit, but for Brazil (a larger one), it very much did. This suggests that increasing the number of ants can have a large impact on the optimality of the solutions found by Ant Colony Optimisation, and this impact is proportional to the size of the solution space.

Number of Iterations

The Brazil file had a major outlier at 1 iteration (fitness 59369), so it was removed from the graph to display the data points of 100-1000 iterations more clearly.

The results for the Burma file and Brazil file are quite conclusive – they suggest that an increase in the number of iterations results in a more optimal path found. This improvement is diminishing however, and it doesn’t appear to be beneficial to perform more than 1000 iterations for either of the two input files.

Algorithm Variation

For Burma, the best performance was from the standard implementation, but for Brazil, the best performance came from the elitist implementation. In both cases, MMAS performed the worst, however MMAS has adjustable parameters that need to be explored (below).

The main parameters that affected MMAS performance were the exponents used in the probability calculation. For the other algorithms, setting both to 1 was optimal, but for this algorithm, a value of 10 was needed for beta (the desirability exponent).

Pheromone Evaporation Rate

For the Brazil file, increasing pheromone evaporation rate led to better solutions up to a value of 0.45, at which point increasing it slowly led to worse solutions. The improvement in fitness as this value goes from 0.5 to 1 is by far the greatest, almost halving the cost of the best solution. Similar results appeared for the Burma file, which had an optimal value (3323) at ~0.4, and a low average solution there.

Heuristic Function

This experiment tests heuristic functions of the form Q/d, and so it includes another heuristic, 1/d.

For the Burma file, there appears to be no correlation between pheromone importance and fitness. On the other hand, for the Brazil file, pheromone importance is directly proportional to a lower fitness. Furthermore, the average best fitness found for the latter improves at a similar rate, meaning the results are consistent.

It makes the most sense to pick a value with near-optimal results, but which won’t cause excessive side effects when paired with other parameter changes. A value of 15-20 seems to fit these conditions best.

**Analysis**

*Question 1*

A high ant population, high number of iterations, and moderate pheromone evaporation rate produced the best results for both input files. It is likely that the impact of these parameters becomes more pronounced with larger input files, as they had a more significant impact on the Brazil file than the smaller Burma file. Additionally, the best performing algorithm variation is the elitist algorithm, and the best heuristic function appears to be Q/d where Q ≈ 20.

*Question 2*

The more ants in the population, the more paths are explored and evaluated every iteration. This means that the likelihood of finding a better path than the last iteration increases. Similarly, increasing the number of iterations increases the number of times the above occurs, resulting in a similar trend.

Having a moderate evaporation rate can be explained by elimination of both extremes. A low evaporation rate results in the algorithm falling into local minima, as the importance of pheromone is too low compared to that of the heuristic function. A high evaporation rate, on the other hand, results in ants not improving paths at all, as the importance of pheromone far outweighs that of the heuristic function.

The elitist algorithm performs the best, likely because it adds a larger incentive to continue exploring along the best possible path compared to the standard algorithm.

*Question 3*

*Execution Time*

There are two parameters which primarily control execution time: number of ants and number of iterations. These control how many ant traversals are performed per iteration, and how many iterations there are, respectively.

*Solution Quality*

The larger the ant population, number of iterations or numerator of the heuristic function (Q), the better the algorithms performed, with diminishing returns. The algorithm variation used causes a variation in the quality of the output, and can change the optimal values for each parameter, further altering the quality of solutions. Finally, the pheromone evaporation rate causes improved solution quality as it converges on its optimal value, and worse solution quality as it diverges from it.

*Question 4*

Another local heuristic function to add is a negative-exponential heuristic: . Like the heuristic function Q/d, this function gives a smaller value as d increases. However, the negative-exponential curve has a much slower incline for values approaching 0, which may prove useful in some situations.

*Question 5*

A parallelised Ant Colony Optimisation algorithm would allow for more efficient use of system resources, and therefore allow further convergence, especially for the Brazil file. This variation would a good inclusion if I were to continue this project, as it would allow for further analysis of the convergence limits of TSPLIB files, potentially allowing experimentation with larger search spaces.

*Question 6*

An alternative nature-inspired algorithm that might have produced better results is Particle Swarm Optimisation. The fact that Particle Swarm Optimisation has fewer parameters to tune[[2]](#endnote-2) means it would have produced near-optimal results after less experimentation. Therefore, in another experiment, choosing this algorithm instead may lead to more time available to be spent tuning the remaining parameters, or further optimising the algorithm.

1. TSPLIB Website, 13/12/2023, Universität Heidelberg. Website: <http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/> [↑](#endnote-ref-1)
2. Wikipedia, 13/12/2023. Website: <https://en.wikipedia.org/wiki/Particle_swarm_optimization> [↑](#endnote-ref-2)