Nature Inspired Computation Project

Oliver Cieplinski

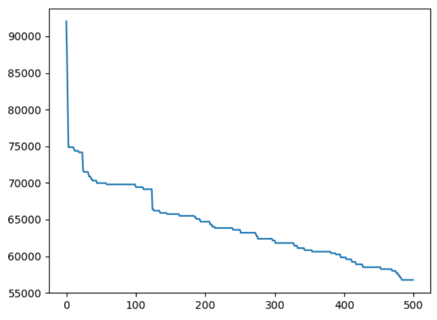
December 2022

1 Algorithm

2 Implementation and Experimentation

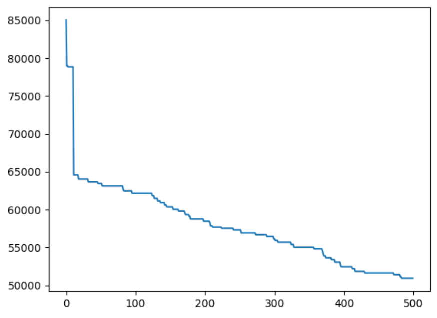
2.1 Experiment 1: m = *100* and e = *0.90*

This trial is initialised with 100 paths and an evaporation rate of 0.9.



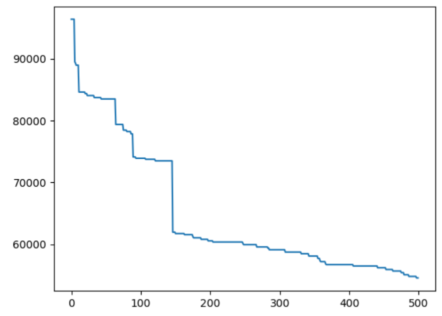
2.2 Experiment 2: m = *100* and e = *0.50*

This trial is initialised with 100 paths and an evaporation rate of 0.5.



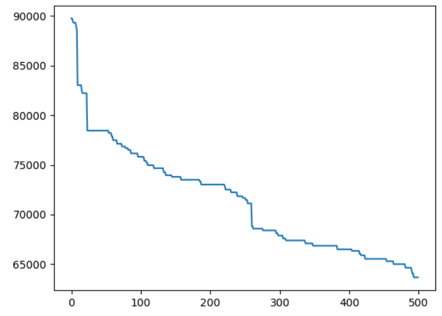
2.3 Experiment 3: *m = 10* and *e = 0.90*

This trial is initialised with 10 paths and an evaporation rate of 0.9.



2.4 Experiment 3: *m = 10* and *e = 0.50*

This trial is initialised with 10 paths and an evaporation rate of 0.5.



3 Analysis

3.1 Question 1: *Which combination of parameters produces the best results?*

The parameters which yielded the best result was experiment was m = 100 and e = 0.50. This is shown by the graphs, and the.

3.2 Question 2: *What do you think is the reason for your findings in Question 1?*

I think the reason that experiment 2 yielded the best outcome is because the evaporation rate is moderate, meaning there is more adaptation that is plausible. Compared to higher evaporation rates that lead to there being a higher likelihood that the ants will stick to a certain path, as the pheromones are stronger for more iterations. I also found through experimenting with larger values of m, that it can lead to higher levels of precision. I believe the reason that the evaporation rate of 0.9 didn't perform as well is because it allows for suboptimal paths to be discovered, and leads to faster convergence which stunts further exploration. I experimented further with higher values of m such as 200, and although the time complexity suffered each iteration yielded far better results when using a moderate evaporation rate, allowing for current routes to be discarded for potentially better ones.

3.3 Question 3: *How do each of the parameter settings influence the performance of the algorithm?*

When increasing the value of e, the performance of the algorithm increase up to a point. Testing with an evaporation rate of 1, I found that no evaporation leads to there being bias towards potential inefficient routes, as the pheromone intensity will stay the same, leading to ants following the same route and throttling performance. High evaporation rates allow for rapid adapatation to the routes, which can be useful when experimenting with lower node counts, however it quickly becomes inefficient with higher node counts and thus more potential paths. I found in general that when tweaking the amount of ant paths, I found that constantly increasing the value will mean that the solutions are found quicker and the algorithm is streamlined faster, depending on evaporation rates. High evaporation rate paired with high ant path counts can mean that early pheromone dense routes can be biased, leading to the program going off course. It also comes at a cost of time complexity, as it slows down the algorithm a lot to test all these fitness evaluations. As seen in the graphs, lower evaporation rates with high ant path counts lead to steady convergence onto the optimal solution, whereas high evaporation rates lead to sporadic results.

3.4 Question 4: *Can you think of a local heuristic function to add?*

A local heuristic function I would add is the a* algorithm. This would allow for the shortest path between nodes to be found, which could be implemented into the function which biases an ant to go towards. I believe this would be beneficial as it would create higher levels of accuracy without sacrificing a large amount of randomness. This would need to be balanced by the evaporation rate, as a search function may possibly lead to quicker pheromone biases upon certain paths.

3.5 Question 5: ***Can you think if any variation for this algorithm to improve your results? Explain your answer.***

A variation of this algorithm could be to allow for variable rates of pheromone evaporation upon high usage paths. Using this, it could allow for potential paths to be discovered that the ants have not come across, while also being robust enough to allow for optimum paths to be continually used. This should be implemented using not just random numbers, but also using functions to calculate how popular the paths are, and then allowing for other paths to be explored. This could also be explored in unused paths, however this could prove to create more disorder rather than increasing efficacy.

3.6 Question 6: ***Do you think of any any other nature inspired algorithms that might have provided better results? Explain your answer***

I believe that an evolutionary algorithm could provide better results in this case. Evolutionary algorithms would allow for multiple generations to be tested, and using mutators and crossover operators it would allow for the optimal fitness to be converged upon while remaining stochastic. An evolutionary algorithm that is trained to lower fitness would mean that population sizes could be tweaked to better fit for the size of the graph, which ant colony optimisation struggles with compared to other algorithms. Simulated annealing could also work in this way, however it would struggle with large amounts of data and possibly allow for convergence on local minimums.