Travelling Salesperson – Ant Colony Optimization

Sudarshan Rao Mundkur

Computer Science Department, Purdue University

Fort Wayne, IN, 46805

[munds01@pfw.edu](mailto:munds01@pfw.edu)

**Problem Statement**: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?

Given this, there are two important rules to keep in mind:

* Each city needs to be visited exactly one time.
* We must return to the starting city, so our total distance needs to be calculated accordingly.

**Ant Colony** optimization is a probabilistic technique for approximating the global optimum of a given function.

Algorithm:

Randomly place ants at the cities

For each ant:

* 1. Choose a not yet visited city until a tour is completed
  2. optimize the tour
  3. Update pheromone

Evaporate Pheromone

**Working of the program:**

|  |  |
| --- | --- |
| Function Name | Functionality |
| Main() | Function used to initialize all parameters, create Symmetric Distance Matrix and Create a random route. Call runAcoTsp() function which returns minimum distances and path for each iterations.  Plot graph for number of iterations versus minimum distance calculated in each iteration. |
| runAcoTsp(space, iterations = 100, colony = 50, alpha = 1.0, beta = 1.0) | Creates the visibility matrix, which is (1/distance matrix).  Creates pheromone matrix and initializes it to zero.  Call method initializeAnts() and moveAnts().  Evaporate pheromones.  Calculate distance for the routes returned by each ants and return minimum distance and path for each iteration. |
| initializeAnts(space, colony) | Randomly initialize each ant to a city position as starting point for the ant. |
| moveAnts | Move the ant from its starting positions to the next city position based on the probability which is calculated using the visibility matrix and pheromone matrix. |
| calcDistance(path , space) | Calculates the total distance of the tour taken by the ant. |

**Observation / Test-Case:**

* Test case - 1

Please enter number of cities: 100

Please enter number of iterations: 100

Please enter number of ants: 15

alpha value: 1

beta value: 1

evaporation rate: 0.5

Chart, histogram

Description automatically generated

* Test case - 2

Please enter number of cities: 110

Please enter number of iterations: 150

Please enter number of ants: 20

alpha value: 1

beta value: 1

evaporation rate: 0.5

Chart, histogram

Description automatically generated

* Test case – 3

Please enter number of cities: 105

Please enter number of iterations: 200

Please enter number of ants: 30

alpha value: 1

beta value: 2

evaporation rate: 0.2

Chart, histogram

Description automatically generated

* Test case – 4

Please enter number of cities: 100

Please enter number of iterations: 250

Please enter number of ants: 35

Alpha value: 2

Beta value: 1

Evaporation rate: 0.4

Chart, histogram

Description automatically generated

* Test case - 5

Please enter number of cities: 100

Please enter number of iterations: 300

Please enter number of ants: 40

Alpha value: 1

Beta value: 1

Evaporation rate: 0.8

Chart, histogram

Description automatically generated

* Test case – 6

Please enter number of cities: 100

Please enter number of iterations: 350

Please enter number of ants: 50

Alpha value: 2

Beta value: 2

Evaporation rate: 0.6

Chart, histogram

Description automatically generated

**Observation:**

After observing the above output results, factors such as number of ants, evaporation rate and number of iterations control the convergence pattern. The Ant colony optimization algorithm takes longer time to execute compared to Genetic algorithm and Simulated Annealing. Increasing the number of ants gives a better, shorter distance but increases the execution time exponentially. Ant colony optimization takes longer time for each iteration, but optimal solution is found in a smaller number of iterations.

**Conclusion**:

Genetic algorithm is fast, easy to implement and cost efficient in terms of computational resources.

Ant colony optimization finds the shortest path, but it takes a long time in execution compared to other algorithms. It gives better results and is better suited when the number of cities is more.

Simulated annealing takes the least amount of time to execute when compared amongst the three algorithms and gives acceptable optimal solution.

**Which method gives better results?**

Ant colony optimization method gives the shortest distance solution amongst the three algorithms.

**Which method is faster?**

Simulated annealing method gives an optimal solution in the least amount of time (less than 1 second) compared to the other two algorithms.

**Note:**

The above conclusions are made solely based on the program execution on my system configuration, programming language used, programming methods and system computation power. The results may vary if the algorithms are run on other systems with different configurations and environment setup.