Title

Ishwarya Anand

# Travelling Salesman Problem

Travelling Salesman Problem (TSP): Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point. Note the difference between Hamiltonian Cycle and TSP. The Hamiltoninan cycle problem is to find if there exist a tour that visits every city exactly once. Here we know that Hamiltonian Tour exists (because the graph is complete) and in fact many such tours exist, the problem is to find a minimum weight Hamiltonian Cycle.



Example of TSP(“Travelling Salesman Problem | Set 1 (Naive and Dynamic Programming) - GeeksforGeeks”, n.d.)

For example, consider the graph shown in figure. A TSP tour in the graph is 1-2-4-3-1. The cost of the tour is 10+25+30+15 which is 80.

The problem is a famous NP hard problem. There is no polynomial time know solution for this problem.

Following are different solutions for the traveling salesman problem.

## Recursive Brute Force Algorithm

* Consider city 1 as the starting and ending point.
* Generate all (n-1)! Permutations of cities.
* Calculate cost of every permutation and keep track of minimum cost permutation.
* Return the permutation with minimum cost

**Time Complexity: Θ(n!)**

## Genetic Algorithm

(“Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator”, n.d.)

* Step 1. Create an initial population of P chromosomes.
* Step 2. Evaluate the fitness of each chromosome.
* Step 3. Choose P/2 parents from the current population via proportional selection.
* Step 4. Randomly select two parents to create offspring using crossover operator.
* Step 5. Apply mutation operators for minor changes in the results.
* Step 6. Repeat Steps 4 and 5 until all parents are selected and mated.
* Step 7. Replace old population of chromosomes with new one.
* Step 8. Evaluate the fitness of each chromosome in the new population.
* Step 9. Terminate if the number of generations meets some upper bound; otherwise go to Step 3.

## Crossover Operator for TSP

Two routes are randomly generated from the tournament population and then participate in crossover to produce a new route. An example crossover of route1 and route2

* Route1 : [NY, SF, Houston, Chicago, Boston, Austin, Seattle, Denver, Dallas, LA]
* Route2 : [LA, Seattle, Austin, Boston, Denver, NY, Houston, Dallas, SF, Chicago]
* Intermediate crossover route : [NY, SF, Houston, Chicago, Boston, null, null, null, null]
* Final crossover route : [NY, SF, Houston, Chicago, Boston, LA, Seattle, Austin, Denver]

## Mutation Operator for TSP

This operator generates a random permutation of the elements present in one chromosome. An example route mutation

* Original route : [Boston, Denver, Los Angeles, Austin, New York, Seattle, Chicago, San Fransisco, Dallas, Houston]
* Mutated route : [Boston, Denver, New York, Austin, Los Angeles, Seattle, San Fransisco, Chicago, Dallas, Houston]

## Ant Colony Optimization

Ants can achieve impressive results as a group while searching for food. FInding food is an optimization problem(gathering food while minimizing the energy to do that). Stigmergy is where individual organisms interact with each other through modifications of their environment. It is a form of self organization that can produce complex structures and behaviour without the need for planning or direct communication. Ants find shortest path from colony to food source using Stigmergy. They use phermones(a chemical they produce) to mark a path for other ants to follow. When other ants come across a phermone trail, they follow it with a certain probability. This probability increases relative to the strength of the phermone trail. Phermone strength increases relative to the number of ants that have followed that trail and how recent this has happened. Ants don’t always follow the same phermone trail because some of them should always be searching for new food sources. Phermone trail decreases over time so that ones that no longer lead anywhere stop being attractive and are replaced by other trails. We exploit some elements of Stigmergy to solve Travelling Salesman Problem.

# Source Code

## Recursive Brute Force ALgorithm to solve Travelling Salesman Problem

## Route.java



## City.java

# References

n.d. <https://www.geeksforgeeks.org/travelling-salesman-problem-set-1/.> <https://www.geeksforgeeks.org/travelling-salesman-problem-set-1/.>

n.d. <https://www.hindawi.com/journals/cin/2017/7430125/.> <https://www.hindawi.com/journals/cin/2017/7430125/.>