ANT COLONY OPTIMIZATION TO SOLVE TRAVELLING SALESMAN PROBLEM

A BACHELOR'S MINI PROJECT

Submitted in partial fulfillment

of the requirements for the completion of the 7th semester of the

UNDERGRADUATE PROGRAM

in

INFORMATION TECHNOLOGY

Submitted by

Rajdeep Singh(IIT2011137)

Ishan Adhaulia(IIT2011133)

Vinay Pratap(IIT2011111)

Ashish Kumar(IIT2011169)

Abhishek Kumar(IIT2011038)

Under the Guidance of

Dr. Sonali Agarwal

Associate Professor

IIIT - ALLAHABAD

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY,

ALLAHABAD – 211012

CANDIDATES' DECLARATION

We hereby declare that the work presented in this project report entitled **"Ant Colony Optimization Algorithm to solve the Travelling Salesman Problem"**, being submitted as a part of Mid Semester Project Evaluation at **Indian Institute Of Information Technology, Allahabad**, is an authenticated record of our original work under the guidance and supervision of **Dr. Sonali Agarwal.**

Date: September 25th, 2014

Place: Allahabad

Rajdeep Singh(IIT2011137)

Ishan Adhaulia(IIT2011133)

Vinay Pratap(IIT2011111)

Ashish Kumar(IIT2011169)

Abhishek Kumar(IIT2011038)

CERTIFICATE

This is to certify that the statement made by the candidates' is correct to the best of my knowledge and belief. The project entitled **"Ant Colony Optimization Algorithm to solve the Travelling Salesman Problem"** is a record of candidates' work carried out by them under my guidance and supervision.

Dr. Sonali Agarwal Date:

Associate Professor,

IIIT Allahabad.

ACKNOWLEDGEMENT

We wish to express our sincere gratitude to **Dr. Sonali Agarwal**, for providing us an opportunity to do our project work on **“Ant Colony Optimization Algorithm to solve the Travelling Salesman Problem”**. Her keen, vital encouragement, superb guidance, and constant support are the motive force behind this project work. We are very thankful to all the technical and non-technical staffs of the college for their assistance and co-operation. Last but not the least we wish to avail ourselves of this opportunity, express a sense of gratitude and love to our friends and our beloved parents for their manual support, strength, help and for everything.

CONTENTS

INTRODUCTION

Ant algorithms are a recently developed, population-based approach which has been successfully applied to several *NP*-hard combinatorial optimization problems such as Traveling Salesman Problem (TSP), Job-shop Scheduling Problem (JSP), Vehicle Routing Problem (VRP), Quadratic Assignment Problem (QAP), etc.
ACO algorithm models the behavior of real ant colonies in establishing the shortest path between food sources and nests. Ants can communicate with one another through chemicals called pheromones in their immediate environment. The ants release pheromone on the ground while walking from their nest to food and then go back to the nest. The ants move according to the amount of pheromones, the richer the pheromone trail on a path is, the more likely it would be followed by other ants. So a shorter path has a higher amount of pheromone in probability, ants will tend to choose a shorter path. Through this mechanism, ants will eventually find the shortest path.

Travelling Salesman Problem –
The traveling salesman problem (TSP), is this: given a finite number of cities along with the cost of travel between each pair of them, find the cheapest way of visiting all the cities and returning to your starting point. (Here, we consider just the symmetric TSP, where traveling from city X to city Y costs the same as traveling from Y to X.) In other words, the data consist of integer weights assigned to the edges of a finite complete graph; the objective is to find a hamiltonian cycle (that is, a cycle passing through all the vertices) of the minimum total weight.

Solutions to the Travelling Salesman Problem –

1. Brute Force
   Make a list of all possible Hamilton circuits
   Calculate the weight of each Hamilton circuit by adding up the weights of its edges.
   Choose the Hamilton circuit with the smallest total weight.
   The Brute-Force Algorithm is guaranteed to find a solution, but, the algorithm is inefficient, since it has to look at all (N - 1)! Hamilton circuits, and this can take a long time.
2. Branch and Bound
   The branch and bound algorithm firstly seeks a solution of the assignment problem.
   Initially a minimum cost is set and at each level after updating the cost of moving to next node, if the current cost becomes greater than the minimum cost till now the whole path in the search space is pruned.
3. Our approach – Ant Colony Optimization
   Implementing the Travelling Salesman Problem using ACO meta heuristics.
   In this approach a colony of (artificial)ants are used to traverse the graph and find the shortest possible route to travel all the cities as described above in the introduction.

PROBLEM SPECIFICATION

Travelling Salesman Problem using Ant Colony Optimization.

For a given weighted graph G = (N, E) and M number of Ants, where N is the set of n cities and E is the set of edges fully connecting all cities. Each edge (i, j) ϵ E is assigned a cost d_{ij}, which is the distance between cities i and j.

It is an NP-hard problem (A problem is NP-hard if solving it in polynomial time would make it possible to solve all problems in class NP in polynomial time).
Approaches to solve such problems:

1. Devising algorithms for finding exact solutions (they will work reasonably fast only for small problem sizes).
2. Devising "suboptimal" or heuristic algorithms, i.e., algorithms that deliver either seemingly or probably good solutions, but which could not be proved to be optimal.

Apply Ant Colony Optimization algorithm to solve the above problem.
At every node in the graph, the decision making probability is based on the amount of pheromone deposited on the outgoing edges at that node. The edge with more amount of pheromone will have more probability of being selected.
Amount of pheromone on an edge is increased every time an ant passes through it, and keeps on evaporating as time passes.

SCOPE

Ant colony optimization algorithms have been applied to many combinatorial optimization problems, ranging from Assignment problems such as Graph coloring problem and University course Timetabling problem to Routing problems such as Travelling Salesman problem and Vehicle routing problem.

The essence of the Traveling Salesman Problem is evident within many practical applications in real life. From a mail delivery person trying to figure out the most optimal route that will cover all of his/her daily stops, to a network architect trying to design the most efficient ring topology that will connect hundreds of computers.
The fundamental goal is to find the optimal tour. That is, to determine an order in which each location should be visited such that each location is visited only once, and the total distance traveled, or cost incurred, is minimal.

LITERATURE SURVEY

Travelling Salesman Problem –

The TSP is a NP-hard combinatorial optimization problem which has attracted a very significant amount of research by Johnson & McGeoch, 1997; Lawler et al., 1985; and Reinelt, 1994.
The TSP has played a central role in Ant Colony Optimization, because it was the application problem chosen when proposing the first ACO algorithm called Ant System (Dorigo, 1992; Dorigo, Maniezzo, & Colorni, 1991b, 1996) and it was used as a test problem for almost all ACO algorithms proposed later.

The Travelling Salesman Problem(TSP) is stated as, given a complete graph, G, with a set of vertices, V, a set of edges, E, and a cost, cij, associated with each edge in E. The value cij is the cost incurred when traversing from vertex i ∈ V to vertex j ∈ V. Given this information, a solution to the TSP must return the cheapest Hamiltonian cycle of G.
A Hamiltonian cycle is a cycle that visits each node in a graph exactly once.

NP –Hard Problem -

NP is the set of decision problems with the following property: If the answer is YES, then there is a proof of this fact that can be checked in polynomial time. Intuitively, NP is the set of decision problems where we can verify a YES answer quickly if we have the solution in front of us.

A problem π is NP-hard if a polynomial-time algorithm for π would imply a polynomial-time algorithm for every problem in NP. In other words:

$$\boxed{\pi \text{ is NP-hard} \Leftrightarrow \text{If } \pi \text{ can be solved in polynomial time, then } P = NP}$$

Combinatorial Optimization –

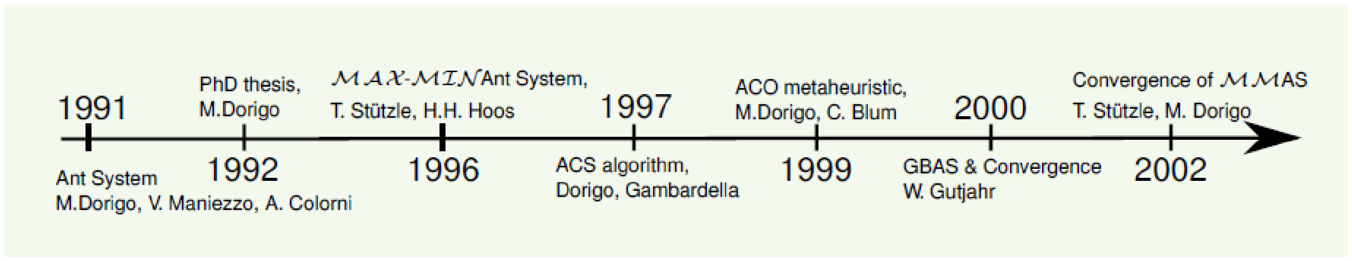
Combinatorial optimization is a topic that consists of finding an optimal object from a finite set of objects. In many such problems, exhaustive search is not feasible. It operates on the domain of those optimization problems, in which the set of feasible solutions is discrete, and in which the goal is to find the best solution. Some common problems involving combinatorial optimization are the traveling salesman problem ("TSP") and the minimum spanning tree problem ("MST").

Swarm Intelligence –

Swarm intelligence is a computational intelligence technique to solve complex real-world problems. It involves the study of collective behaviour of individuals in a population who interact locally with one another and with their environment in a decentralised control system.
Example – Ant Colony Optimization

<u>Ant Colony Optimization</u> –

Was developed by Marco Dorigo (Italy) in his PhD thesis in 1992.



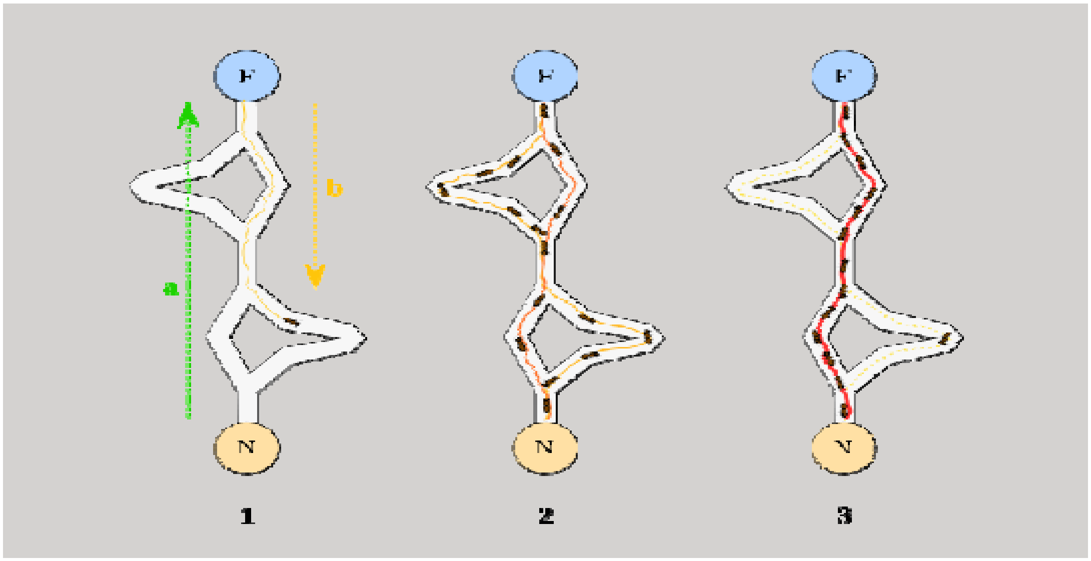
Technique for solving problems which can be expressed as finding good paths through graphs.
Each ant tries to find a route between its nest and a food source.
The behavior of each ant in nature –

1. Wander randomly at first, laying down a pheromone trail.
2. If food is found, return to the nest laying down a pheromone trail.
3. If pheromone is found, with some increased probability follow the pheromone trail.
4. Once back at the nest, go out again in search of food.

However, pheromones evaporate over time, such that unless they are reinforced by more ants, the pheromones will disappear.

Description –

1. The first ant wanders randomly until it finds the food source (F), then it returns to the nest (N), laying a pheromone trail.
2. Other ants follow one of the paths at random, also laying pheromone trails. Since the ants on the shortest path lay pheromone trails faster, this path gets reinforced with more pheromone, making it more appealing to future ants.



3. The ants become increasingly likely to follow the shortest path since it is constantly reinforced with a larger amount of pheromones. The pheromone trails of the longer paths evaporate.

Pheromones –
A pheromone is a chemical an animal produces which changes the behavior of another animal of the same species (animals include insects). Some describe pheromones as behavior-altering agents. Many people do not know that pheromones trigger other behaviors in the animal of the same species.

Meta-heuristic –

Heuristic – Pertains to the process of gaining knowledge or some desired result by intelligent guesswork rather than by following some pre established formula. The term seems to have two usages:

1. Describing an approach to learning by trying without necessarily having an organized hypothesis or way of proving that the results proved or disproved the hypothesis. That is, "trial-by-error" learning.
2. Pertaining to the use of the general knowledge gained by experience, sometimes expressed as "using a rule-of-thumb."

A meta-heuristic is a heuristic method for solving a very general class of computational problems in the hope of obtaining a more efficient or more robust procedure. Meta-heuristics are generally applied to problems for which there is no satisfactory problem-specific algorithm or heuristic; or when it is not practical to implement such a method. Most commonly used meta-heuristics are targeted to combinatorial optimization problems.

PROPOSED APPROACH

According to Marco Dorego and Thomas Stutzle,

Ant Colony optimization is a metaheuristic in which a colony of artificial ants cooperate in finding good solutions to difficult discrete optimization problems, TSP being one of them.
Although the ACO metaheuristic can be applied to any interesting combinatorial optimization problems, the real issue is how to map the considered problem to a representation that can be used by the artificial ants to build solutions.

Informally, an ACO algorithm can be imagined as the interplay of three procedures:

1. ConstructAntsSolutions
2. UpdatePheromones

ConstructAntsSolutions –
Manages a colony of ants that concurrently and asynchronously visit adjacent states of the considered problem by moving through neighbour nodes of the problems construction graph. They move by applying a local decision policy that makes use of pheromone trails and heuristic information. In this way, ants incrementally build solutions to the optimization problem. Once an ant has built a solution, or while the solution is being built, the ant evaluates the (partial) solution that will be used by the UpdatePheromones procedure to decide how much pheromone to deposit.

UpdatePheromones -
Is the process by which the pheromone trails are modified. The trails value can either increase, as ants deposit pheromone on the components or connections they use, or decrease, due to pheromone evaporation .From a practical point of view, the deposit of new pheromone increases the probability that those components that were either used by many ants or that were used by at least one ant and which produced a very good solution will be used again by future ants. Differently, pheromone evaporation implements a useful form of forgetting: it avoids a too rapid convergence of the algorithm toward a suboptimal region, therefore favoring the exploration of new areas of the search space.

<u>OUTLINE OF ACO METAHEURISTIC</u> -

```
procedure ACO Metaheuristic
        ScheduleActivities
                ConstructAntsSolutions
                UpdatePheromones
        end-ScheduleActivities
end-procedure
```

SOLUTION TO THE ABOVE PROBLEM –

Transformation of parameters –

NATURE	**PROGRAM**
Natural habitat	Graph (nodes and edges)
Nest and food	Nodes in the graph: start and destination
Ants	Agents, our artificial ants
Visibility	The reciprocal of distance, η
Pheromones	Artificial pheromones, τ
Foraging behavior	Random walk through graph (guided by pheromones)

Parameters -

Construction graph:

The construction graph is identical to the problem graph. The set of nodes correspond to the cities and the connections correspond to the set of edges, and each connection has a weight which corresponds to the distance d_{ij} between nodes i and j. The states of the problem are the set of all possible partial tours.

Constraints:

The only constraint in the TSP is that all cities have to be visited and that each city is visited at most once. This constraint is enforced if an ant at each construction step chooses the next city only among those it has not visited yet.

Pheromone trails and heuristic information:

The pheromone trails t_{ij} in the TSP refer to the desirability of visiting city j directly after i. The heuristic information h_{ij} is typically inversely proportional to the distance between cities i and j, a straight-forward choice being h_{ij}.

Solution construction:

Each ant is initially put on a randomly chosen start city and at each step iteratively adds one still unvisited city to its partial tour. The solution construction terminates once all cities have been visited.

Applying the above procedure to solve TSP:

<u>INITIALISATION</u>:

1. Global constants alpha, beta, rho and Q are declared and initialized.
```
alpha = 3.      // influence of pheromone in direction
beta  = 2.      // influence of adjacent node distance
rho   = 0.01    // pheromone decrease factor
Q     = 2.0     // pheromone increase factor
```

2. First a graph of n cities is created with random edge lengths in the range 1.0 to 8.0 such that the graph is connected, symmetric and undirected.
   The graph is created as a 2-D double array dist[][] where dist[i][j] denotes the distance to move from city i to city j.

3. Artificial ants are created which traverse the graph asynchronously and independently.
   To simulate artificial ants a 2-D integer array ants[][] is created where ants[i] is an array of int values that represent the trail or path, from an initial city through all other cities for the ant with index i.
   Here the first index indicates the ant number.
   The initialization method allocates a row for the trail for each ant picking a random start

4. A 2-D double array pheromones[][] is created and initialized to 0.01.

<u>UPDATING ANTS</u>:

Here the ant array is updated such that each ant is reinitialized by constructing a new (better) trail based on the pheromone and distance information.
Updating basically determines the next city.
For determining the next city, an array called "tau" is constructed.
tau value = (pheromone[i][j] ^ alpha) * (1 / d[i][j]) ^ beta)
Larger values of pheromone increase tau.
Larger distances of edge decrease tau.
Now find the probability of the all the cities and select the one with the maximum.

<u>UPDATING PHEROMONES</u>:

Pheromone value is decreased, simulating evaporation and increased, simulating the deposit of pheromones by ants already visited the trail.

pheromones[i][j] = pheromones[i][j] + (Q / d[i][j]) + (rho * pheromones[i][j]).

PSEUDO CODE

```
main() {
        numCities, numAnts
        Creation and Initialization of graph matrix
        Initialization of the ant’s trails
        Creation and Initialization of pheromone matrix

        BestTrail(…)                        //Finds the best trail among the trails of all the ants
        Bestlength = Length(…)              //Length of the best trail

        Initialize an arbitrary tmax

        for ( time = 0 to tmax ) {
                UpdateAnts(…)               //Starts from a random ant, and builds the trail based on
                                            the amount of pheromone deposited on the outgoing
                                            paths
                UpdatePheromones(…)         //Updates pheromone value for each edge present in the
                                            trail
                BestTrail(…)

                currbestlength = Length(…)

                If (currbestlength < bestlength)
                        Update bestlength
                        Store the corresponding trail
        }
}
```

Output: Best Length and the corresponding trail denoting the order in which cities are to be travelled.

TOOLS USED

System:

Linux Ubuntu
Version: 12.04 LTS

Language used:

C++
Version: 4.6.3
Libraries: Standard Template Library

ACTIVITY TIME CHART

TASK	STATUS	DURATION
Study of the Ant Colony Optimization	Completed	10th Aug. – 15th Aug.
Figuring out the problem on which to apply ACO	Completed	17th Aug. – 19th Aug.
Study of the Travelling Salesman Problem	Completed	20th Aug. – 21st Aug.
Implementation of TSP using Branch and Bound approach	Completed	22nd Aug. – 23rd Aug.
Identifying the Base Research Paper	Completed	23rd Aug. – 27th Aug.
Analysis of the Research Paper (Ant Colony Optimization: A New Meta-Heuristic by Marco Dorigo and Gianni Di Caro)	Completed	28th Aug. – 2nd Sept.
Implementation of TSP using the ACO meta-heuristic	Completed	5th Sept. – 19th Sept.
Preparation of the project report	Completed	20th Sept. – 23rd Sept.
Preparation of the project presentation	Completed	22nd Sept. – 24th Sept.
Comparative study between the results obtained from Branch and Bound approach and ACO	Under progress	25th Sept. – till date
Continuous discussion with the guide		

WORK DONE TILL MID-SEM

- Understood the concept of the Travelling Salesman Problem -

  Type of problem: NP – Hard.
  General solving technique: Brute force method, Branch and Bound approach.

- Implemented the Travelling Salesman Problem using Branch and Bound approach.

  Works better than the Brute force method (i.e., exhaustive search for the results obtained from all the possible combinations and comparing the results).

  Problems encountered with the above technique - well only for solving the problems with no more than 40-80 nodes. For the practical relevance, it is necessary to solve the larger-scale problems with the help of heuristics.

- Familiarized with the basic elements of Ant Colony Optimization (ACO) technique of solving combinatorial problems.

  Heuristic methods vary from exacts methods in that they give no guarantee to find the optimal solution to the given problem (so that solution is called suboptimal), but in many cases this is the solution of good quality and we can obtain it in acceptable time. Heuristic methods are usually focused on solving the special type of problems.

- Implemented the Travelling Salesman Problem using Ant Colony Optimization (ACO) meta-heuristics.

WORK TO BE DONE BY END-SEM

- Comparative Study of the performances of the Ant Colony Optimization (ACO) implementation and the Branch and Bound approach of solving the Travelling Salesman Problem on the basis of time complexity analysis.

- To improve the performance of the Ant Colony Optimization (ACO) implementation of the Travelling Salesman Problem.

How ?

The following algorithms help in increasing the performance of Ant Colony Optimization algorithm –

- Roulette Wheel Selection algorithm - is a genetic operator used in genetic algorithms for selecting potentially useful solutions for recombination.

- Fisher Yates algorithm - The Fisher Yates shuffle is quite efficient; indeed, its time and space complexity are optimal.

Looking for further ideas also.

REFERENCES

[1] Marco Dorigo and Thomas Stutzle, Ant Colony Optimization. Cambridge, MIT Press, 2004.

[2] Marco Dorigo and Gianni Di Caro, Ant Colony Optimization: A New Meta-Heuristic, IEEE Press.
[3] Dorigo M. & L.M. Gambardella (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*, 1, 1, in press.

[4] Wikipedia(http://en.wikipedia.org/wiki/Travelling_salesman_problem)
[5] Wikipedia(http://en.wikipedia.org/wiki/Ant_colony_optimization_algorithms)
[6] Wikipedia(http://en.wikipedia.org/wiki/Branch_and_bound)
[7] Wikipedia(http://en.wikipedia.org/wiki/Metaheuristic)
[8] Wikipedia(http://en.wikipedia.org/wiki/Combinatorial_optimization)
[9] Wikipedia(http://en.wikipedia.org/wiki/Fitness_proportionate_selection)
[10] Wikipedia(http://en.wikipedia.org/wiki/Fisher%E2%80%93Yates_shuffle)

[11] http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/AproxAlgor/TSP/tsp.htm
[12] http://www.scholarpedia.org/article/Ant_colony_optimization

COMMENTS AND SUGGESTIONS