IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-3, Issue-3 E-ISSN: 2347-2693

Relative Investigation of Ant Colony Optimization and Genetic Algorithm based Solution to Travelling Salesperson Problem

Ashwani Chandel[1], Vikram Jeet Singh[2*]

[1,2]*Department of Computer Science, Himachal Pradesh University, India*

www.ijcseonline.org

Received: Feb/24/2015 Revised: Mar/06/2015 Accepted: Mar/22/2015 Published: Mar/31/2015

***Abstract*—** Travelling salesperson problem is a nondeterministic polynomial hard problem in combinatorial optimization studied in Operations Research and theoretical computer science. To solve this problem, we used two popular meta-heuristics techniques-Ant Colony Optimization and Genetic Algorithm. Both techniques are applied to solve a TSP with same dataset. We then compare them. For Ant Colony Optimization, we studied the effect of some parameters (number of ants, evaporation and number of iterations) on the produced results. On the other hand, we studied chromosome population, crossover probability and mutation probability parameters that effect Genetic Algorithm results.

***Keywords*—** Ant; Colony; Genetic; Algorithm; Travelling; Salesperson

I. INTRODUCTION

Travelling salesperson problem is described as: there are cities and given distances between and a travelling salesperson has to visit all of them, but he does not want to spend time on travelling more than required. We need to find the sequence of cities to minimize the travelled distance. The problem was first formulated as a mathematical problem in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, a large number of heuristics and exact methods are known, so that instances with tens of thousands of cities can also be solved. TSP is represented in numerous transportation and logistics applications such as: Arranging routes for school buses to pick up children in a school district; Delivering meals to home-bound people; Scheduling stacker cranes in a warehouse; Planning truck routes to pick up parcel post and many others; Planning logistics and the manufacture of microchips; or, A classic example of the TSP is the scheduling of a machine to drill holes in a circuit board [2].

The objective of this study is to compare Ant Colony Optimization (ACO) and Genetic algorithm (GA) on TSP.

II. ANT COLONY OPTIMIZATION

Ant Colony Optimization (ACO) is one of the most popular meta-heuristics used for combinatorial optimization in which an optimal solution is sought over a discrete search space. A well-known example is the TSP [2], where the search-space of candidate solutions grows more than exponentially as the size of the problem increases, which makes an exhaustive search for optimal solution infeasible. It was introduced by Marco Dorigo in the early 1990's [3], [4], [5]. Since then, several improvements have been devised by Gambardella et al [15] and Stützle et al [1]. ACO algorithm is based on a computational paradigm inspired by real ant colonies and the way they function. The underlying idea is to use several constructive computational agents or simulating real ants [7]. Ant's behaviour is governed by the goal of colony survival rather than being focused on the survival of individuals. The behaviour that provided inspiration for ACO is the ants' foraging behaviour (Figure 1), and in particular, how ants can find shortest paths between food sources and their nest. When searching for food, ants initially explore the area surrounding their nest in a random manner. While moving, ants leave a chemical pheromone trail on the ground.

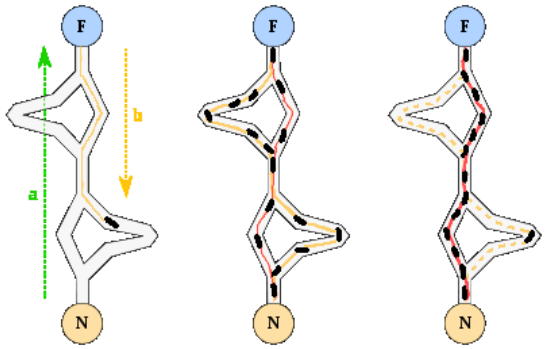


Figure 1: Ants Use Pheromone as Indirect Communication to Build Best Tour

Ants can smell pheromone. When choosing their way, they tend to choose, in probability, paths marked by strong pheromone concentrations. As soon as an ant finds a food source, it evaluates the quantity and the quality of the food and carries some of it back to the nest. During the return trip, the quantity of pheromone that an ant leaves on the ground may depend on the quantity and quality of the. Pheromone trails guide other ants to the food. The indirect communication between ants via pheromone trails enables

Corresponding Author: *Vikram,* vikramjeetsingh@live.in

them to find shortest paths between their nest and food source.

A. *ACO Parameters*

ACO algorithm is meta-heuristic that optimizes a problem iteratively trying to improve a candidate solution with regard to a given measure of quality. In general, meta-heuristic doesn't guarantee an optimal solution. The most asked question is: what is the best result can we obtain in less iteration with minimum cost and time? Or when to terminate? This require a good estimation for parameters used with ACO algorithm, like pheromone trail decay coefficient (p), pheromone amount, number of ants (M), maximum number of iterations.

B. *ACO representation of TSP*

A travelling salesperson is required to pass through a number of cities, each city is visited once and he needs to find the shortest closed path tour that link all cities. Thus, we have undirected graph consists of V nodes or cities linked by undirected E edges G = (V, E) the edge weights represent distances between cities. As shown in Figure 2, the search space S consists of all tours in G. The objective function value f(s) of a tour $s \in S$ is defined as the sum of the edge-weights of the edges that are in S.

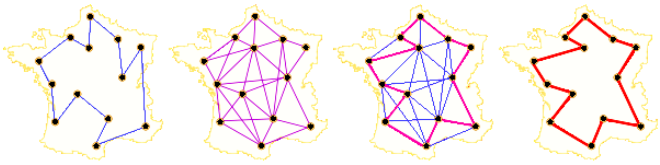


Figure 2: Undirected Graph showing Four Stages of ACO to Reach Shortest Closed Path

TSP can be modelled in different ways as a discrete optimization problem. Concerning the Ant system approach, the edges of the given TSP graph can be considered solution components, ant introduce a pheromone value Ti, j for the edge ei, j. The general algorithm is based on a set of ants, each making one of the possible tours or round-trips along the cities. Each tour considered as one solution s of search space S, and the sum of the edges-weights is the objective function f(s). Now we search for the best tour s at which we have smallest f(s). The following steps describe how each ant constructs a solution s:

- Each ant chose randomly one city as start node.
- The ant starts building the tour by moving from one city to another unvisited city.
- The traversed edge is chose by probability P(ei,j).
- The traversed edge is added to the solution being constructed.
- When all cites are visited the ant move to the start node.
- Having completed its journey, the ant deposits more pheromones on all edges it traversed. Deposited pheromones are: $Ti, j \leftarrow T_{i,j}+1/f(s)$
- After each iteration, trails of pheromones evaporate is done.

The previous steps is used to construct one tour, these steps can be repeated more and more to obtain the optimum solution. In each tour, the more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen.

III. GENETIC ALGORITHM

Genetic algorithms as part of evolutionary computing technique, are inspired by Darwin's theory. Solution to a problem solved by genetic algorithms is evolved. I. Rechenberg introduced the idea of evolutionary computing in the 1960s in his work "Evolution strategies" (Evolutions strategy in original). Other researchers then developed his idea. Genetic Algorithms (GAs) were invented by John Holland and developed by him and his students and colleagues [8]. In 1992 John Koza used genetic algorithm to evolve programs to perform certain tasks [8]. He called his method as genetic programming. LISP programs were used, because programs in this language can be expressed in the form of a "parse tree", which is the object GA works upon.
Genetic algorithm is started with a set of solutions (chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are selected to form new solutions (off springs) are selected according to their fitness, the more suitable they are the more chances they have to reproduce. This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied. It is well known that problem solving can be often expressed as looking for the extreme of a function and GA tries to find the minimum/maximum of the function. Following is the broad outline of a basic Genetic Algorithm:

1- [Start] Generate random population of n chromosomes (suitable solutions for the problem).
2- [Fitness] Evaluate the fitness f(x) of each chromosome x in the population.
3- [New population] Create a new population by repeating the following steps until the new population is complete.
   - [Selection] Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
   - [Crossover] with a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.
   - [Mutation] with a mutation probability mutate new offspring at each locus (position in chromosome)
   - [Accepting] place new offspring in a new population.

4- [Replace] use new generated population for a further run of algorithm
5- [Test] if the end condition is satisfied, stops, and returns the best solution in current population.
6- [Loop] Go to step 2

There are many things that can be implemented differently in various problems. First question to be answered is how to create chromosomes and what types of encoding to choose; next question is how to select parents (in hope that the better parents will produce better offspring). Making a new population only by new offspring can cause loss of best chromosomes from the last population, so a method called Elitism is used. At least one best solution is copied without changes to a new population, so the best solution found can survive to the end of the run. Basic parameters of GA are: Crossover Probability, Mutation Probability and Population size. Crossover and mutation are most important parts of a GA. The performance is influenced mainly by these two operators.

A. GA Parameters

Crossover probability suggests how often crossover is to be performed. If there is no crossover, offspring is an exact copy of parents. If there is a crossover, offspring is made from parts of parent's chromosome. If crossover probability is 100% then all offspring is made by crossover. If it is 0 %, whole new generation is made from exact copies of chromosomes from old population (but this does not mean that the new generation is the same). Crossover is made in hope that new chromosomes will have good parts than old chromosomes and they may be better. However, it is good to leave some parts of the population to survive to the next generation.

Mutation probability, on the contrary, shows how often parts of chromosome will be mutated. If there is no mutation, offspring is taken after crossover without any change. If mutation is performed, part of chromosome is changed. If mutation probability is 100%, whole chromosome is changed, if it is 0%, nothing is changed. Mutation is made to prevent falling GA into local extreme, but it should not occur very often, because in this case GA will in fact change to random search.

Population size means how many chromosomes are in population (in one generation). If there are too few chromosomes, GA has a few possibilities to perform crossover and only a small part of search space is explored. On the other hand, if there are too many chromosomes, GA slows down. Research shows that after some limit (depends mainly on encoding and the problem) it is not useful to increase population size, as it does not make solving the problem faster.

B. GA Representation of TSP

Since chromosomes are selected from the population to be parents to crossover. The problem is how to select these chromosomes. According to Darwin's evolution theory the best ones should survive and create new offspring. There are many methods how to select the best chromosomes, for example, there is Roulette wheel selection, Boltzmann selection, Tournament selection, Rank selection, Steady state selection, or some others. Encoding of chromosomes is one of the problems, when we are starting to solve problem with GA. Encoding depends greatly on the problem, but there are many encoding style like as: Binary encoding, Permutation encoding.

First, we need to decide how to represent a route of the salesperson. The most natural way of representing a route is the path representation. Each city is given an alphabetic or numerical name, the route through the cities is represented as a chromosome, and appropriate genetic operators are used to create new routes. For TSP, we use Permutation encoding for ordering the problem. Every chromosome is a string of numbers, which represents the numbers in a sequence. Permutation encoding is only useful for ordering problems. Even for these problems for some types of crossover and mutation corrections must be made to leave the chromosome constant (have real sequence in it). Chromosomes denote the order of cities, in which salesperson shall visit them.

Crossover and mutation operators depend on type of encoding and also on the problem. For our encoding and problem, we use single point crossover; as shown in Figure 3, the permutation is copied from the first parent until we reach this point, then the second parent is scanned and if the number is not yet in the offspring it is added. There are more ways to produce the rest after crossover point (*).

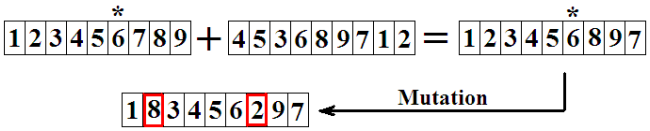
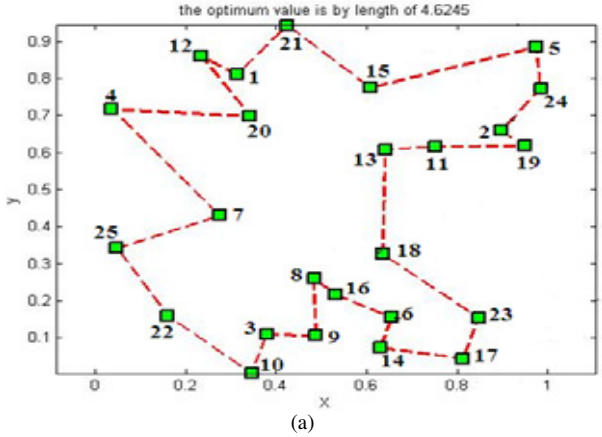


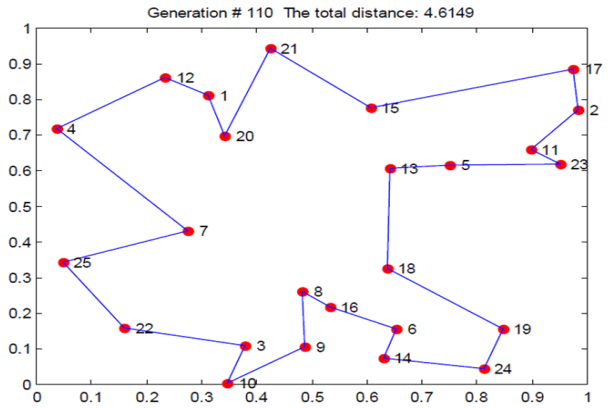
Figure 3: Crossover and mutation for permutation encoding

IV. COMPARISON

Both techniques (GA and ACO) are used to solve TSP with high acceptable performance. As in Figure 4, we can see the best tour and distance between 25 cities for the same data by using both GA and ACO.



(a)



(b)

Figure 4: The best tour of 25 cities using both (a) ACO and (b) GA [8]

For ACO, the result is obtained by using 2500 iterations, 250 ants for each iteration and 0.9 as an evaporation coefficient to produce the best distance of 4.6245. While for GA, the best distance is 4.6149 by using crossover and mutation probability as 0.75 and 0.009 respectively after 110 iterations with 200 chromosomes. Advantage of GA is the small spent time against the large time required by ACO [8].

V. CONCLUSION

From the study of both the algorithms, we conclude that it is difficult to select the best parameter for ACO, but we can observe the dependency of the number of iterations on both the evaporation coefficient p and the number of ants M. that if p=0 that have no evaporation, the algorithm does not converge. But when p is large enough (p=0.9), the algorithm often converged to suboptimal solutions for complex problem. It is necessary to evaluate that how these parameters have an effect on best number of iterations and evaporations coefficient. Also for GA, we need to select the best value for chromosome population, crossover, and mutation probabilities. But still, GA is better suited than ACO for solving a TSP. The results can be used for working upon the design and architecture of various systems and applications as in [9], [10], [11], [12].

REFERENCES

[1] M.Dorigo and T.Stutze, "Research Paper on Ant Colony Optimization", MIT Press, Cambridge (2004).
[2] E. Lawer and J. Rooney, "Research Paper on The Travelling Salesman Problem", John Wiley & Sons, New York (1985).
[3] M. Dorigo, "PhD thesis Optimization, Learning and Natural Algorithms", Politecnico di Milano, Italy (1992).
[4] M. Dorigo and A. Colorni, "Research Paper on Ant System: Optimization by a Cooperating Agents", IEEE Trans Syst Man Cabernet Part B, p. 29-41 (1996).
[5] M. Dorigo and A. Colorni, "Technical Report on a Positive Feedback Strategy", Politecnico di Milano, Italy (1991).
[6] M. Dorigo and L.M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Travelling Salesman Problem", IEEE Transactions on Evolutionary Computation, Vol 1 (1997).
[7] S. Camazine and J.L.Deneubourg,"Research Paper on Self-Organization in Biological Systems", Princeton University Press, Princeton (2001).
[8] J.L. Deneubourg and S. Goss, "The self-organization exploratory pattern of the Argentine ant", J Insect Behavior, pages 59-68 (1990).
[9] Vikram Jeet Singh and Ashwani Chandel, "Evolving E-Governance through Cloud Computing based environment", International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), Vol 3 Issue 4.
[10] Ashwani Chandel and Manu Sood, "Searching and Optimization Techniques in Artificial Intelligence: A Comparative Study and Complexity Analysis", International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), Vol 3 Issue 3 (2014).
[11] Vikram Jeet Singh, Vikram Kumar and Kishori Lal Bansal, "Research on Application of Perceived QoS Guarantee through Infrastructure specific Traffic Parameter Optimization", International Journal of Computer Network and Information Security (IJCNIS), Issue 3, MECS Publisher-Hong Kong (2014).
[12] Ashwani Chandel and Vikram Jeet Singh, "Research on the Design Architecture & Services over a State Wide Area Network: A case of Himachal Pradesh", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Volume 4, Issue 2 (2015)