Implementation of Ant Colony Optimization: Travelling Salesman Problem

Shivali Moudgil
moudgil.shivali@gmail.com

Monika Pandey
a.monika.pandey@gmail.com

Abstract- In this paper we describe anartificial ant colony capable of solving the Travelling Salesman Problem(TSP). Ants of the artificial colony are capable to generate successively shorter feasible tours by using information accumulated in the form of pheromone trail deposited on the edges of the TSP graph. Computer simulations demonstrate that the artificial ant colony is capable of generating good solutions to both symmetric and asymmetric instances of the TSP.

I. INTRODUCTION

The ant colony optimization(ACO) is a heuristic algorithm that is inspired by the food foraging behavior of ants.The idea underline the ACO algorithm is to mimic the ant's foraging behavior with "simulated ants" moving around a graph searching for the optimal solution.As ants forage, they deposit a trail of slowly evaporating pheromone.all foraging ants use the pheromone as a guide regardless of whether the pheromone is deposited by itself or other ants. Pheromone accumulate as multiple ants travel through the same path. The pheromone on the trail evaporates as well. If an ant reaches the food first and returns to the nest before others, it return trail's pheromone stronger than other trails on which ants have not found food or have longer distances from the food source because the **return** trail has been travelled twice. This high volume of pheromone attracts other ants to follow this trail. The more the trail is travelled the stronger the pheromone content will be. The level of pheromone on other less travelled trails will decrease since fewer ants travel those trail and the pheromone evaporates. Eventually, the trail with highest level of pheromone is travelled by most of foraging ants will be the shortest trail and nest [8]. Consider the Fig.1.

We will show how a similar process can be put to work in a simulated world inhabited by artificial ants that try to solve the TSP.

The Travelling Salesman problem can be described as follows*: Given a complete graph with weights on the edges(arcs), find a hamiltonian cycle in graph of minimum total weight.*

II. ANT COLONY SYSTEM

Ant Colony System(ACS) meteheuristics is a particular class of ants algorithms. The ACS algorithm has been introduced by Dorigo and Gambardella [5] to improve the performance of Ant System [4], that allowed to find good solutions within a reasonable time for great problems. The ACS is based on three modifications of *Ant System*:a different transition rule, a different pheromone trail updating rule and the use of local and global pheromone updating rule, to favor exploration. An pseudo code for ACS algorithm is follows:

begin
Set parameters, initialize pheromone trails
Loop
Each ant is positioned on a starting node
Loop
Each ant applies a state transition rule to incrementally build a solution and a local pheromone updating rule
Until all ants have built a complete solution A global pheromone updating rule is applied Until End_condition
end

III. TECHNICAL IMPLEMENTATION of ACO USING ARTIFICIAL ANTS

In this work an artificial ant is an agent which moves from city to city on a TSP graph. It chooses the city to move to using a probabilistic function both of trail accumulate on edges and of a heuristic value, which was chosen here to be a function of the edges length. Artificial ants probabilistically prefer cities that are connected by edges with a lot of pheromone trail and which are closely-by. Initially, m artificial ants are placed on randomly selected cities. At each time step they move to new cities and modify the pheromone trail on the edges used-this is termed *local trail updating*. When all the ants have completed a tour the ant that made the shortest tour modifies the edges belonging to its tour-termed *global trail updating*-by adding an amount of pheromone trail that is inversely proportional to the tour length.

Published By:
International Journal of Engineering Research in Current Trends

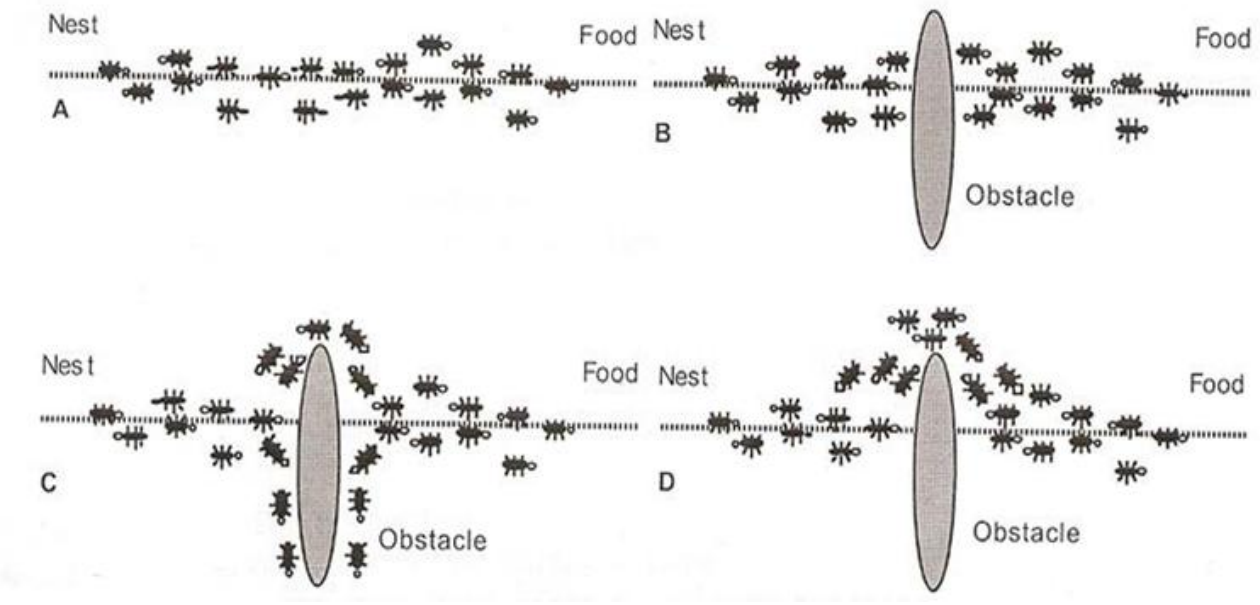

Fig.1. (A) Real ants follow a path between nest and food source.
(C) Pheromone is deposited more quickly on the shorter path.

Fig.2. (B) An obstacle appears on the path: Ants choose whether to turn left or right with equal probability.
(D) All ants have chosen the shorter path.

These are three ideas from natural ant behavior that we have transferred to our artificial ant colony; (i) the preference for paths with a high pheromone level, (ii) the higher rate of growth of the amount of pheromone on shorter paths, (iii) the trail mediated communication among ants .
Artificial ants were also given a few capabilities which do not have a natural counterpart, but which have been observe to be well suited to the TSP application: artificial ants can determine how far away cities are, and they are endowed with a working memory Mk use to tour , and is updated after each time step by adding the new visited city.

There are many different ways to translate the above principles into a computational system apt to solve the TSP. In our ant colony system (ACS) an artificial ant K in city r chooses the city s to move to among those which do not belong to its working memory Mk by applying the following probabilistic formula:

s= argmax otherwise $\{[\tau(r,u)] * [\eta(r,u)]\hat{}\beta\}$ $\leq$
s=S.

$if\ q \quad q0$

W here $\tau(r,u)$ is the amount of pheromone trail on edge (r,u) $\eta(r,u)$ is a heuristic function which was chosen to be the inverse of the distance between cities (r,u), β is a parameter which weighs the relative importance of pheromone trail and of closeness, q is a value chosen randomly with uniform probability in[0,1], q0 $\left(0 \leq q0 \leq 1\right)$ is a parameter, and S is a random variable selected according to the following probability distributed ,
which favors edges which are shorter and have a higher level of pheromone trail;

Pk(r,s)= $[\tau(r \quad ,s)]*[\eta(r,u)]^{\beta} / \sum[\tau(r,u)] * [\eta(r,u)]\hat{}\beta$

Otherwise,
Pk(r,s)=0.

Where Pk(r,s) is the probability with which ant K chooses to move from city r and city s.
Pheromone trail is changed both locally and globally.

Global updating is intended to reward edges belonging to shorter tours. Once artificial ants have completed their tours, the best ant deposits pheromone on visited edges: that is, on those edges that belong to its tour (the other edges remain unchanged.)-The amount of pheromone deposited on each visited edge (r,s) by the best ant is $\Delta\phi(r,s)$ inversely proportional to the length of the tour : *The shorter the tour the greater the amount the pheromone deposited on edges*. This manner of depositing pheromone is intended to evaluate the property of differential pheromone trail accumulation, which in the case of real ants was due to the interplay between length of path and continuity of time. The global trail updating formula is

$\phi(r,s) < -(1-\alpha).\varphi(r,s) + \alpha.\Delta\varphi(r,s),$ where $\Delta\varphi(r,s) = (shortest\ tour)\hat{}-1.$

Global trail updating is similar to a reinforcement learning scheme in which better solution get a higher reinforcement.

Local updating is intended to avoid a very strong edge being chosen by all ants: Every time an edge is chosen by an ant its amount of pheromone is changed by applying the local trail updating formula:

$\tau(r,s) < -(1-\alpha).\tau(r,s) + \alpha.\tau 0,$

where $\tau 0$is a parameter. Local trail updating is also motivated by trail evaporation in real ants.

It is interesting to note that ACS employs a novel type of exploration strategy. First, there is the stochastic component of formula (1): here the exploration of new path is based towards strong and high trail edges. (Formula (1),which we call pseudo-random-proportional action choice rule, is strongly reminiscent of the pseudo-random action choice rule often used in reinforcement learning). Second, local trail updating tends to encourage exploration since each path taken has its pheromone value reduced by the local updating formula.

IV. RESULT

We applied ACS to the symmetric and asymmetric TSP listed in Tables 1,2.

Using the test problems listed in Tables 1,2 the performance of ACS was compared with the performance of other naturally inspired global optimization methods: simulated annealing(SA),neural nets(NNs), here represented by the elastic net (EN) and by the self organizing map (SOM), evolutionary computation (EC) ,here represented by the genetic algorithm (GA) and by evolutionary programming (EP), and the combination of simulated annealing and genetic algorithms (AG); moreover we compared it with the farthest insertion (FI) heuristic. Numerical experiments were executed with ACS and FI, whereas the performance figures for the other algorithms were taken from the literature. The ACS parameters were set to the following values: m=10, q0=0.9, $\beta = 2, \alpha = 0.1, \tau 0 = (n.Lnn)\hat{}-1$, where Lnn is the tour length produced by the nearest neighbor heuristic, and n is the number of cities.

We also ran ACS on some bigger problems to study its behavior for increasing problem dimensions. For these runs we implemented a slightly modified version of ACS which incorporates a more advanced data structure known as candidate list, a data structure normally used when trying to solve big TSP problems. A candidate list is a list of preferred cities to be visited ; it is a static data structure which contains, for a given city I, the cl closest cities. In practice, an ant in ACS with candidate list first chooses the city to move to among those belonging to the candidate list. Only if none of the cities in the candidate list can be visited then it considers the rest of the cities.

Table 1. Comparison of ACS with other nature inspired algorithms on random instances of the symmetric TSP. Comparisons on average tour length obtained on five 50-city problems.SA=simulated annealing, EN=elastic net, SOM=self organizing map, FI=farthest insertion . Results on SA, EN, and SOM are from (Durbin and Willshaw, 1989; Potvin, 1993). FI results are averaged over 15 trials starting from different initial cities. ACS was run for 1,250 iterations using m=20 ants and the results are averaged over 15 trials. The best average tour length for each problem is in boldface.

Problem name	ACS	SA	EN	SOM	FI
City set 1	5.86	5.88	5.98	6.06	6.03
City set 2	6.05	6.01	6.03	6.25	6.28
City set 3	5.57	5.65	5.70	5.83	5.85
City set 4	5.70	5.81	5.86	5.87	5.96
City set 5	6.17	6.33	6.49	6.70	6.71

Table 2. Comparison of ACS with other nature-inspired algorithms on random instances of the symmetric TSP. Comparison on the shortest tour length obtained by SA+3-opt=best tour length found by simulated annealing and many distinct runs of 3-opt, SOM+= best tour length found by SOM over 4000 different runs (by processing the cities in various orders), F1, F1+3-opt=best tour length found by farthest insertion locally optimized by 3-opt, and ACS with and without local optimization by 3-opt. The 3-opt heuristics used the result of ACS and FI as starting configuration for local optimization. Results on SA+3-opt and SOM+ are from (Durbin and Willshaw, 1987;Potwin, 1993). ACS was run for 1,250 iterations using m=20 ants and the best tour length was obtained out of 15 trials. The best tour length for each problem is in boldface.

Problem name	ACS	ACS + 3-opt	SA + 3-opt	SOM +	FI	FI + 3-opt
City set 1	5.84	5.84	5.84	5.84	5.89	5.85
City set 2	6.00	6.00	5.99	6.00	6.02	5.99
City set 3	5.57	5.57	5.57	5.58	5.57	5.57
City set 4	5.70	5.70	5.70	5.60	5.76	5.70
City set 5	6.17	6.17	6.17	6.19	6.50	6.40

V. CONCLUSIONS

The key to the application of the ACS to a new problem is to identify an appropriate representation for the problem (to be represented as the graph searched by many artificial ants), and an appropriate heuristic that defines the distance between any two nodes of the graph. Then the probabilistic interaction among the artificial ants mediated by the pheromone trail deposited on the graph edges will generate good, and often optimal, problem solutions.

There are many ways in which ACS can be improved so that the number of tours needed to reach a comparable performance level can diminish, making its application to larger problem instances feasible. First, a local optimization heuristic like 2-opt, 3-opt or Lin-Kernighan(Lin and Kernighan,1973) can be embedded in the ACS algorithm (this is a standard approach to improve the efficiency of general purpose algorithms like EC, SA, NNs, as discussed in (Johnson and McGeoch, in press)). In the experiments presented in this article, local optimization was just used to improve on the best results produced by the various algorithms. On the contrary, each ant could be taken to its local optimum before global trail updating is performed. Second, the algorithm is amenable to efficient parallelization, which could greatly im[prove the performance for finding good solutions, especially for high- dimensional problems. The most immediate parallelization of ACS can be achieved by distributing ants on different processors: the same TSP is then solved on each processor by a smaller number of ants, and the best tour found is exchanged asynchronously among processors. A preliminary implementation (Bolondi and Bondanza,1993) of a similar scheme (Dorigo, Maniezzo and Colorni, 1996) on a net of transputers has shown that it can make the complexity of the algorithm largely independent of the number of ants. Third, the method is open to further improvements such as the introduction of specialized families of ants , tighter connections with reinforcement learning methods (Gambardella and Dorigo,1995; Dorigo and Gambardella, 1996) , and the introduction of more specialized heuristic functions to direct the search.

ACKNOWLEDGEMENT

Writing of this paper happens to be one of our achievement in this phase of life. We extend our gratitude to our role model teachers and excellent academicians *Mr.Praveen Kaithal and Mr.Ashish Sen* for all their contributions and academic guidance in making this paper more informative and of a respectable standard.

We would also like to thank all our friends and classmates who encouraged us and provided a very conductive environment to make the most of our time.

Lastly, we want to thank our college for organizing this prestigious paper presentation event.

REFERENCES

[1] Balas, E., Ceria, S. and Cornuejoles, G.,1993, A lift -and-project cutting plane algorithm for mixed 0 -1 programs, mathematical programming, 58, 295-324.

[2] Durbin, R. and Willshaw, D., 1987, An analogue approach to the travelling salesman problem using an elastic net method. Nature, 326, 689-691.

[3] Dorigo, M., Maniezzo, v. and Colorni, A., 1996, The Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on systems, man, and cybernetic-Part B, 26, 29-41.

[4] M. Dorigo, Optimization , Learning and Natural Algorithms (in Italian), Ph.D thesis, Dipartmento. di Elettronica., Politecnico di Milano, Italy, (1992), 1-140.

[5] M. Dorigo, L.M. Gambardella, Ant Colony System : A cooperative learning approach to the Travelling Salesman Problem, IEEE Trans. Evol. Comp., 1, (1997), 53-66.

[6] Marco Dorigo, Luca Maria Gambardella on Ant colonies for the travelling Salesman problem. Available: http://www.idsia.ch/~luca/acs- bio97.pdf

[7] Andreea Vescan and Camelia-M.Pintea on Ant Colony Component-based system for Travelling Salesman Problem. Available:http://www.m- hikari.com/ams...2007/.../pinteaAMS25 -28-2007.pdf

[8] Bonabeau, E., F. Henaux, (1998). Routing in telecommunications networks with ant- like agents 1437:60.