ISSN No. 0976-5697

Volume 8, No. 4, May 2017 (Special Issue)

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

A Review of Ant Colony System Algorithm and its Models

Gurpreet Singh[1]
Assistant Professor,Deptt. Of CSE [1]
Sant Baba Bhag Singh University[1]
Khiala, Jalandhar, India[1]
gurujaswal@gmail.com[1]

Prof. (Dr.) Jatinder Singh Bal[2]
Professor,Deptt. Of CSE[2]
Sant Baba Bhag Singh University[2]
Khiala, Jalandhar, India[2]
bal_jatinder@rediffmail.com[2]

Abstract **-Ant Colony Optimization (ACO) is meta-heuristic algorithm inspired from nature to solve many combinatorial optimization problems such as Travelling Salesman Problem (TSP). ACO algorithms for datagram networks was given by DiCaro &M.Dorigo, in year 1996. One direction that researchers have gone to pursue this is to study the behaviour of ants for their techniques to find the shortest path between two points ACO is a meta-heuristic approach for solving hard combinatorial optimization problems.In this paper, the Ant Colony Optimization Technique has been applied in different network models with different number of nodes and structure to find the shortest path with optimum throughput. The performance measure taken here is shortest path as well as time taken by the data from source to destination.**

KEYWORDS:-Ant colony optimization, Combinatorial optimization, metaheuristics optimization, MAX–MIN Ant System, quadratic assignment problem.

I. INTRODUCTION

The routing protocols play a very important role in calculating, choosing and selecting the relevant path for transferring the data from the source to the destination efficiently. There are already many accepted routing algorithms [2] to find the shortest path and also to increase the throughput of the network.

Combinatorial optimization problems are intriguing because they are often easy to state but very difficult to solve. Many of the problems arising in applications areNP-hard, that is, it is strongly believed that they cannot be solved to optimalitywithin polynomials bounded computation time. Hence, to practically solve largeinstances one often has to use approximate methods which return near-optimal solutionsin a relatively short time. Algorithms of this type are loosely called heuristics. They often use some problem-specific knowledge to either build or improve solutions. A metaheuristic is a set of algorithmicconcepts that can be used to define heuristic methods applicable to a wide set of different problems. The use of metaheuristics has significantly increased the ability of finding very high quality solutions to hard, practically relevant combinatorial optimization problems in a reasonable time [14].

A. COMBINATORIAL OPTIMIZATION

Combinatorial optimization problems involve finding values for discrete variables such that the optimal solution with respect to a given objective function is found. Many optimization problems of practical and theoretical importance are of combinatorial nature [15].A combinatorial optimization problem is either maximization or a minimizationproblem which has associated a set of problem instances. The term problem refers to the general question to be answered, usually having several parameters or variables with unspecified values. The term instance refers to a problem with specified values for all the parameters.

More formally, an instance of a combinatorial optimization problem ∏ is a triple (S, F, Ω), where S is the set of candidate solutions, f is the objective function which assigns an objective function value f(s) to each candidate solution s € S, and Ω is a set of constraints. The solution isbelonging to the set~SJS of candidate solutions that satisfy the constraints Ω are called feasible solutions. The goal is to find a globally optimal feasible solution s*. For minimization problems this consists in finding a solution s*€ ~ S with minimum cost, that is, a solution such that f(s*)≥f(s) for all s€~ S; Note that in the followingwe focus on minimization problems and that the obvious adaptations have to be made if one considers maximization problems[1].

B. NATURAL BEHAVIOUR OF ANTS

Real ants are capable of finding the shortest path from a food source to their nest. While walking ants deposit pheromone on the ground and follow pheromone previously deposited by other ants, the essential trait of ACO algorithms is the combination of a priori information about the structure of a promising solution with a posteriori information about the structure of previously obtained good solutions [10]. To find a shortest path, a moving ants lay some pheromone on the ground, so an ant encountering a previously trail can detect it and decide with high probability to follow it. As a result, the collective behaviour that emerges is a form of a positive feedback loop where the probability with which an ant chooses a path increases with the number of ants that previously choose the same path.

a) Real ants follow a path between nest and food source.
b) An obstacle appears on the path: Ants choose whether to turn left or right with equal probability.
c) Pheromone is deposited more quickly on the shorter path.

d) All ants have chosen the shorter path.

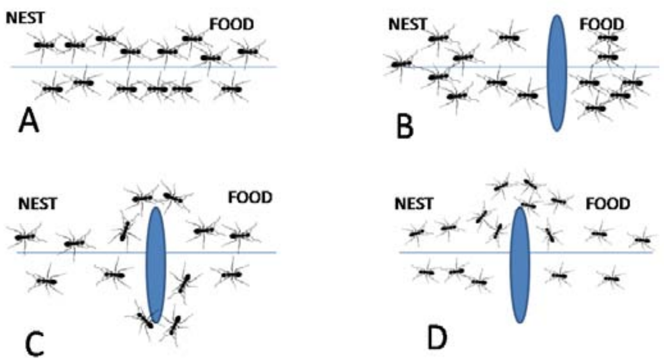

Figure. 1. Behaviour of ants

II. ANT COLONY OPTIMIZATION

Ant colony optimization is a metaheuristic in which a colony of artificial ants cooperates in finding good solutions to difficult discrete optimization problems. Cooperation is a key design component of ACO algorithms: The choice is to allocate the computational resources to a set of relatively simple agents (artificial ants) that communicate indirectly by stigmergy, that is, by indirect communication mediated by the environment. Good solutions are an emergent property of the agents' cooperative interaction [1]. ACO algorithms can be used to solve both static and dynamic combinatorial optimization problems. Static problems are those in which the characteristics of the problem are given once and for all when the problem is defined, and do not change while the problem is being solved [2].

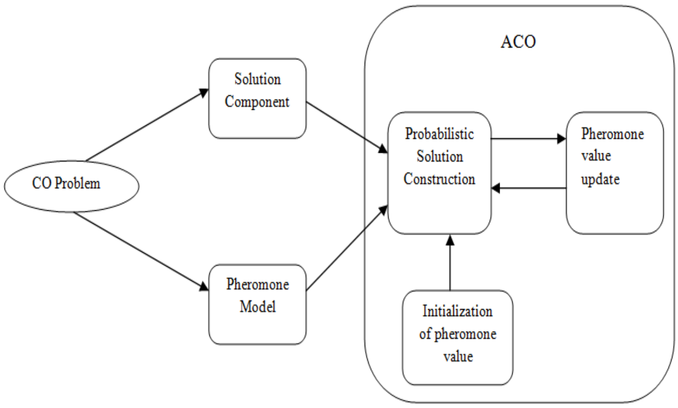

Figure. 2. The working of the ACO metaheuristic.

An ACO algorithm can be imagined as the interplay of three procedures: Construct-Ants-Solutions, Update Pheromones, and Daemon Actions [8].

a) *ConstructAntsSolutions*manages a colony of ants that concurrently and asynchronously visit adjacent states of the considered problem by moving through neighbour nodes of the problem's construction graph GC. They move by applying a stochastic local decision policy that makes use of pheromone trails and heuristic information [4]. 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 Update Pheromones procedure to decide how much pheromone to deposit [6].

b) *Update Pheromones* 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/connections 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 favouring the exploration of new areas of the search space.

c) *Daemon Actions* procedure is used to implement centralized actions which cannot be performed by single ants. Examples of daemon actions are the activation of a local optimization procedure, or the collection of global information that can be used to decide whether it is useful or not to deposit additional pheromone to bias the search process from a nonlocal perspective. As a practical example, the daemon can observe the path found by each ant in the colony and select one or a few ants (e.g., those that built the best solutions in the algorithm iteration) which are then allowed to deposit additional pheromone on the components/connections they used.

Algorithm:

```
Procedure ACO Metaheuristic
    Schedule-Activities
        Construct-Ants-Solutions
        Update-Pheromones
        Daemon-Actions % optional
    End-Schedule-Activities
End-procedure
```

The ACO metaheuristic is described in pseudo-code. The main procedure of the ACO metaheuristic manages the scheduling of the three above-discussed components of ACO algorithms via the Schedule Activities construct: (1) management of the ants' activity, (2) pheromone updating, and (3) daemon actions. The ScheduleActivities construct does not specify how these three activities are scheduled and synchronized. In other words, it does not say whether they should be executed in a completely parallel and independent way, or if some kind of synchronization among them is necessary. The designer is therefore free to specify the way these three procedures should interact, taking into account the characteristics of the considered problem.

III. Max-Min Ant System

MAX–MIN Ant System (MMAS) introduces four main modifications with respect to AS. The main modifications introduced by MMAS with respect to AS are the following [9].

a) It strongly exploits the best tours found: only either the iteration-best ant, that is, the ant that produced the best tour in the current iteration, or the best-so-far ant is allowed to deposit pheromone. Unfortunately, such a strategy may lead to a stagnation situation in which all the ants follow the same tour, because of the excessive growth of pheromone trails on arcs of a good, although suboptimal, tour.

b) To counteract this effect, a second modification introduced byMMAS is that it limits the possible range of pheromone trail values to the interval $[r_{min,}f_{max}]$.

c) Third, the pheromone trails are initialized to the upper pheromone trail limit, which, together with a small pheromone evaporation rate, increases the exploration of tours at the start of the search.

d) Finally, in MMAS, pheromone trails are reinitialized each time the system approaches stagnation or when no improved tour has been generated for a certain number of consecutiveiterations [13].

A. *Update of Pheromone Trails*

After all ants have constructed a tour, pheromones are updated by applying evaporation as in Ant system followed by the deposit of new pheromone as follows:

$r_{ij} \leftarrow r_{ij+\Delta} r_{ij}^{best}$,

Where $\Delta r_{ij}^{best=}1/c^{best}$, the ant which is allowed to add pheromone may be either the best-so-far, in which case $\Delta r_{ij}^{best}=1/c^{bs}$, or the iteration-best, in which case $\Delta r_{ij}^{best}=1/c^{ib,}$ where C^{ib} is the length of the iteration-best tour. General, in MMAS implementations both the iteration-best and the best-so-far update rules are used, in an alternate way[9].

B. *Pheromone Trail Limits*

In MMAS, lower and upper limit $r_{min}$and r_{max}on the possible pheromone values on any arc are imposed in order to avoid search stagnation. In particular, the imposed pheromone trail limits have the effect of limiting the probability p_{ij} of selecting a city j when an ant is in city i to the interval $[p_{min},p_{max}]$, with $0<p_{min}\le p_{ij}\le p_{max}\le 1$. Only when an ant k has just one single possible choice for the next city, that is $|N_i^k|=1$, we have $p_{min}=p_{max}=1$.

C. Pheromone Trail Initialization and Re-initialization

At the start of the algorithm, the initial pheromone trails are set to an estimate of the upper pheromone trail limit. This way of initializing the pheromone trails, in combination with a small pheromone evaporation parameter, causes a slow increase in the relative difference in the pheromone trail levels, so that the initial search phase of MMAS is very explorative. As a further means of increasing the exploration of paths that have only a small probability of being chosen, in MMAS pheromone trails are occasionally reinitialized. Pheromone trail re-initialization is typically triggered when the algorithm approaches the stagnation behaviour (as measured by some statistics on the pheromone trails) or if for a given number of algorithm iterations no improved tour is found [9].

IV. Ant Colony for Quadratic assignment problem

The quadratic assignment problem (QAP) is the problem of assigning *n* facilities to *n* locations so that the assignment cost is minimized, where the cost is defined by a quadratic function. The QAP is considered one of the hardest CO problems, and can be solved to optimality only for small instances [12]. The QAP can best be described as the problem of assigning a set of facilities to a set of locations with given distances between the locations and given flows between the Facilities. The goal is to place the facilities on locations in such a way that the sum of the products between flows and distances is minimized. More formally, given n facilities and n locations, two n×n matrices $A=[a_{ij}]$ and $B=[b_{rs}]$, where a_{ij} is the distance between locations i and j and b_{rs} is the flow between facilities r and s, and the objective function [9].

$$f(\prod) = \sum_{i,j=1}^{n} \text{bija}_{\prod i \prod j}$$

Where$\prod_I$ gives the location of facility i in the current solution $\prod$ € (n), then the goal in the QAP is to find an assignment of facilities to locations that minimizes the objective function. The term $\text{bija}_{\prod i \prod j}$describes the cost contribution of simultaneously assigning facility i to location $\prod_i$ and facility j to location$\prod$j.

In the following, we describe the application of AS (AS-QAP), MMAS (MMASQAP), and ANTS (ANTS-QAP) to the QAP. Some of the main design choices, such as the definition of the construction graph and of the pheromone trails, are the same for the three algorithms [9].

a) *Construction graph*the set of components C comprises all facilities and locations. The connections L fully connect the components. A feasible solution is an assignment consisting of n pairs (i, j) between facilities and locations, with each facility and each location being used exactly once. There are no explicit costs assigned to the couplings.

b) *Constraints*The only constraint is that a feasible solution for the QAP assigns each facility to exactly one location and vice versa. This constraint can be easily enforced in the ants' walk by building only couplings between still unassigned facilities and locations.

c) *Pheromone trails*the pheromone trails r_{ij}refer to the desirability of assigning facility ito location j (or the other way round, the two choices being equivalent).

d) *Heuristic information* MMAS-QAP does not use any heuristic information, whereas AS-QAP and ANTS-QAP do.

A. Application to the quadratic assignment problem

The solutions (ants) found so far are then optimized using a local search method, update of the pheromone trail simulates the evaporation and takes into account the solutions produced in the search strategy. In some way, the pheromone matrix can be seen as shared memory holding the assignments of the best found solutions [14]. The different steps of this algorithm are following:-

a) *Solution construction*AS-QAP sorts the facilities in non-increasing order of flow potentials and at each construction step an ant k assigns the next, still unassigned, facility ito a free location j using the action choice rule of Ant system. In ANTS-QAP, the lower bound is computed once at the start of the algorithm. Along with the lower bound computation one gets the values of the dual variablescorresponding to the constraints when formulating the QAP as an integer programming. These values are used to define the order in which locations are assigned. The action choice rule is the same as that used by the ANTS algorithm.

b) *Pheromone update*the pheromones for all three algorithms are updated following the corresponding rules defined for each of these algorithms.

c) *Local search*all three ACO algorithms were combined with a 2-opt local search procedure for the QAP. MMAS-QAP was also tested with a local search procedure based on short runs of a tabu search algorithm.

V. CONCLUSION

We have successfully reviewed the ant colony for the optimization problem. It is clearly understood from this paper how ant colony originated, the basic concepts and the working of ant colony optimization. Ant colony optimization is one of the well-known optimization technique. In this paper, three variants of ACO for Network Routing is proposed and implemented on various standard Network Models. The results show that the type of the variations of the ACO that should be applied on the network obviously depends on the structure, size and the application. The key to the application of ACS to a new problem is to identify an appropriate representation for the problem (to be represented as a graph searched by many artificial ants), and an appropriate heuristic that defines the distance between any two nodes of the graph. Ant System can mainly be seen as a first study to demonstrate the viability of ACO algorithms to attack NP-hard combinatorial optimization problems, but itsperformance compared to other approaches is ratherpoor.

REFERENCES

[1] Pardeep Kaur,” Ant colony optimization: An optimization technique”, International Journal of Advance Foundation and Research in Science & Engineering (IJAFRSE) Volume 1, Special Issue, ICCICT 2015**.**

[2] Vittorio Maniezzo, Alberto Colorni Marco Dorigo "The Ant System: Optimization by a colony of cooperating agents," IEEE Transactions on Systems,Man, and Cybernetics–Part B, vol. Vol.26, pp. pp.1-13, 1996.

[3] Debasmita Mukherjee, SriyankarAcharyya,” Ant Colony Optimization Technique Applied in Network Routing Problem”, International Journal of Computer Applications (0975 - 8887) Volume 1 – No. 15, 2014.

[4] Khushbu Khatri,Vinit Kumar Gupta,” A Survey Paper on Solving TSP using Ant Colony Optimization on GPU”, An international journal of advanced computer technology, 3 (12), December-2014.

[5] Ms. Meenakshi R Patel, Ms. BabitaKubde,” A survey paper on Ant Colony Optimization Routing algorithm for selecting Multiple Feasible Paths for Packet Switched Networks**”** IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 2, No 3, March 2012.

[6] Vijayalaxmi,Dr.Chandrashekhara, S.Adiga,”AntColony, Optimization Techniques for network Routing Problem”, International Conference on Innovations in Engineering and Technology (ICIET'2013) Dec. 25-26, 2013.

[7] Ismail Musirin,MuhammadKhayat Idris.” Ant Colony Optimization (ACO) Technique in Economic Power Dispatch Problems”, International Multi Conference of Engineers and Computer Scientists 2008 Vol II IMECS 2008, 19-21 March, 2008.

[8] Zar Chi Su SuHlaing,May Aye Khine,” Solving Traveling Salesman Problem by Using Improved Ant Colony Optimization Algorithm”, International Journal of Information and Education Technology, Vol. 1, No. 5, December 2011.

[9] Dweepna Garg, Saurabh Shah,” ANT COLONY OPTIMIZATION FOR SOLVING TRAVELING SALESMAN PROBLEM” International Journal of Computer Science and System AnalysisVol. 5, No. 1, January-June 2011, pp. 23-29.

[10] BhanuPratap Singh, SohanGarg,”A Characteristics Study of Ant Colony Optimization Algorithms for Routing Problems”,International journal Of Advanced Research in Computer Science And Software Engineering,Volume 3, Issue 3, March 2013.

[11] E.-G. Talbi, O. Roux, C. Fonlupt,” Parallel Ant Colonies for the quadratic assignment problem”: International journal Of Advanced Research in Computer Science And Software Engineering, pp 441–449,2001.

[12] Guo Hong,” A Hybrid Ant Colony Algorithm for Quadratic Assignment Problem”,*the Open Electrical & Electronic Engineering Journal,* 2013, *7,* (Supple 1: M5) 51-54.

[13] M. Dorigo, "The ant system: optimization by a colony of cooperating agents," *IEEE Trans Syst. Man, Cybern,* vol. 26, pp. 1-13, 1996.

[14] L. Maria Gambardella, "Ant colonies for the quadratic assignment problem," *J. Oper. Res. Soc.*, vol. 50, pp. 167-176, 1999.

[15] L. Bianchi, L.M. Gambardella et M.Dorigo, *An ant colony optimization approach to the probabilistic traveling salesman problem*, PPSN-VII, Seventh International Conference on Parallel Problem Solving from Nature, Lecture Notes in Computer Science, Springer Verlag, Berlin, Allemagne, 2002.