ISSN: 2321-7782 (Online)
Volume 3, Issue 6, June 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Ant Colony Optimization using Routing Information Algorithm in Matlab

Mohit Malik[1]
Student
Department of CSE
Sat Kabir Institute of Tech. and Mgmt.
Bahadurgarh, Haryana - India

Kirti Bhatia[2]
Assistant Professor
Department of CSE
Sat Kabir Institute of Tech. and Mgmt.
Bahadurgarh, Haryana - India

Suryakiran Suhag[3]
Assistant Professor
Department of CSE
Sat Kabir Institute of Tech. and Mgmt.
Bahadurgarh, Haryana - India

Abstract: ***One of the most important phases of ACO algorithms is the construction phase during which an ant builds a partial solution and develops a state transition strategy. There have been a number of studies on the state transition strategy. However, most of the research studies look at how to improve pheromone updates rather than at how the ant itself makes a decision to move from a current position to the next position.***

The aim of this research is to develop a novel state transition strategy for Ant Colony Optimization algorithms that can improve the overall performance of the algorithms using routing information Protocol (RIP). The new proposed algorithm is called Cognitive Ant Colony Optimization and uses a new concept of decision-making taken from cognitive behaviour theory in routing selection protocol.

An ACO algorithm designed to help solve the routing problem in telecommunications networks. Network routing refers to the activities necessary to guide information in its travel from source to destination nodes. It is an important because it has a strong influence on the overall network performance and difficult because networks' characteristics, such as traffic load and network topology, may vary stochastically and in a time-varying way. It is in particular these characteristics of the problem, in addition to the physical distributedness of the overall problem on a real network, that make ACO algorithms a particularly promising method for its solution. In fact, the ACO processing paradigm is a good match for the distributed and non stationary (in topology and traffic patterns) nature of the problem, presents a high level of redundancy and fault tolerance, and can handle multiple objectives and constraints in a flexible way.

Keywords: Routing Information protocol (RIP), Ant Colony Optimization Algorithm (ACO), Traffic Pattern, Fault Tolerance.

I. INTRODUCTION

Communications networks can be classified as either circuit-switched or packet switched. The typical example of a circuit-switched network is the telephone network, in which a virtual or physical circuit is set up at the communication start and remains the same for the communication duration. Differently, in packet-switched networks, also called data networks, each data packet can, in principle, follow a different route, and no fixed virtual circuits are established. In this case the typical examples are local area computer networks and the Internet. Arguably, the main function of a data network, on which we focus in this work, is to assure the efficient distribution of information among its users. This can be achieved through the exploitation of an adequate network control system. One of the most important components of such a system, in conjunction with the admission, flow, and congestion control components, is routing (Walrand & Varaiya, 1996). Routing refers to the distributed

activity of building and using routing tables. The routing table is a common component of all routing algorithms: it holds the information used by the algorithm to make the local forwarding decisions. The type of information it contains and the way this information is used and updated strongly depends on the algorithm's characteristics. One routing table is maintained by each node in the network: it tells the node's incoming data packets which among the outgoing links to use to continue their travel toward their destination node. One of the most distinctive aspects of the network routing problem is the non-stationarity of the problem's characteristics. In particular, the characteristics of traffic over a network change all the time, and in some important cases (e.g., the Internet) the traffic can fluctuate in ways difficult to predict. Additionally, the nodes and links of a network can suddenly go out of service, and new nodes and links can be added at any moment.

Therefore, network routing is very different from the NP-hard problems we encountered in previous chapters. In fact, although in some simplified situations it is possible to reduce the routing problem to a standard combinatorial optimization problem, in realistic settings the dynamics of the traffic, and therefore of the costs associated with network links, is such that it might even be impossible to give a formal definition of what an optimal solution is.

Although AntSim is not the only ACO algorithm developed for routing problems, and not even the historically first one, we focus on it because it is the sole algorithm to have reached, at least at the experimental/simulation level at which it was tested, state-of-the-art performance. We give a detailed description of Antsim's data structures and control procedures, and a brief overview of the results obtained using a network simulation environment.

II. Broad Classification of Routing Algorithm

Routing algorithms can be broadly classified as centralized versus distributed and as static versus adaptive.

A. Centralized Routing Algorithm

In centralized algorithms, a main controller is responsible for updating all the node's routing tables and for making every routing decision. Centralized algorithms can be used only in particular cases and for small networks. In general, the delays necessary to gather information about the network status and to broadcast the decisions and the updates make them infeasible in practice. Moreover, centralized systems are not fault-tolerant: if the main controller does not work properly, the entire network is affected.

B. Distributed Routing Protocol

In contrast, in distributed routing, the computation of paths is shared among the network nodes, which exchange the necessary information. The distributed paradigm is currently used in the great majority of networks.

In static routing, the path taken by a data packet is determined only on the basis of its source and destination, without regard to the current network traffic. The path chosen is usually the minimum cost path according to some cost criterion, and can be changed only to account for faulty links or nodes. Adaptive routing is, in principle, more attractive, because it can adapt the routing policy to time and spatially varying traffic conditions. As a drawback, adaptive algorithms can cause oscillations in the selection of paths. This can cause circular paths, as well as large fluctuations in measured performance (Bertsekas & Gallager, 1992). Another interesting way of looking at routing algorithms is from an optimization perspective. In this case the main choice is between optimal routing and shortest path routing.

C. Optimal Routing Protocol & Shortest-path

Optimal routing has a network-wide perspective and its goal is to optimize a function of all individual link flows (usually this function is a sum of link costs assigned on the basis of average packet delays) (Bertsekas & Gallager, 1992).

Shortest-path routing has a source-destination pair perspective: there is no global cost function to optimize. Its objective is to determine the shortest path (minimum cost) between two nodes, where the link costs are computed (statically or adaptively)

according to some statistical description of the traffic flow crossing the links. Considering the different content stored in each routing table, shortest-path algorithms can be further subdivided into two classes called distance-vector and link-state (Steenstrup, 1995).

D. Distance-vector algorithms

Distance-vector algorithms make use of routing tables consisting of a set of triples of the form (destination, estimated distance, and next hop), defined for all the destinations in the network and for all the neighbor nodes of the considered switch. In this case, the required topologic information is represented by the list of identifiers of the reachable nodes. The average per node memory occupation is in the order of O (φ.n) where φ is the average connectivity degree (i.e., the average number of neighbor nodes considered over all the nodes) and n is the number of nodes in the network. The algorithm works in an iterative, asynchronous, and distributed way. The information that every node sends to its neighbors is the list of its last estimates of the distances (intended as costs) from itself to all the other nodes in the network. After receiving this information from a neighbor node j, the receiving node i update its table of distance estimates overwriting the entry corresponding to node j with the received values. Routing decisions at node i are made choosing as the next hop node the one satisfying the expression is

$$\arg\min_{j \in \mathcal{N}_i} \{d_{ij} + D_j\},$$

Where d_{ij} is the assigned cost to the link connecting node i with its neighbor j and Dj is the estimated shortest distance from node j to the destination. It can be shown that this algorithm converges in finite time to the shortest paths with respect to the used metric if no link cost changes after a given time (Bellman, 1958; Ford & Fulkerson, 1962; Bertsekas & Gallager, 1992); this algorithm is also known as distributed Bellman-Ford.

E. Link-state Algorithm

Link-state algorithms make use of routing tables containing much more information than that used in distance-vector algorithms. In fact, at the core of link-state algorithms there is a distributed and replicated database. This database is essentially a dynamic map of the whole network, describing the details of all its components and their current interconnections. Using this database as input, each node calculates its best paths using an appropriate algorithm such as Dijkstra's (Dijkstra, 1959), and then uses knowledge about these best paths to build the routing tables. The memory requirement for each node in this case is Oðn2Þ. In the most common form of link state algorithm, each node acts autonomously, broadcasting information about its link costs and states and computing shortest paths from itself to all the destinations on the basis of its local link cost estimates and of the estimates received from other nodes. Each routing information packet is broadcast to all the neighbor nodes which in turn send the packet to their neighbors, and so on. A distributed flooding mechanism (Bertsekas & Gallager, 1992) supervises this information transmission, trying to minimize the number of retransmissions. It should be clear to the reader, from what was said in chapter 1, that ACO algorithms can easily be adapted to solve routing problems following the shortest-path/ distance-vector paradigm.

III. The Communication Network Model

In this chapter, we focus on irregular topology packet-switched data networks with an IP-like network layer (in the ISO-OSI terminology (Tanenbaum, 1996)) and a very simple transport layer. In particular, we focus on wide area networks (WANs), of which the Internet is a noteworthy instance. In WANs, hierarchical organization schemes are adopted. Roughly speaking, sub networks are seen as single host nodes connected to interface nodes called gateways. Gateways perform fairly sophisticated network layer tasks, including routing. Groups of gateways, connected by an arbitrary topology, define logical areas. Inside each area, all the gateways are at the same hierarchical level and ''flat'' routing is performed among them. Areas communicate only by means of area border gateways. In this way, the computational complexity of the routing problem, as seen by each gateway, is much reduced, at the cost of an increase in the complexity of the design and management of the routing protocols.

The instances of the communication networks that we consider in the following can be mapped on directed weighted graphs with n processing/forwarding nodes. All the links between pairs of nodes are viewed as bit pipes characterized by a bandwidth (bit/s) and a transmission delay (s). Every node is of type store-and-forward and has a buffer space where the incoming and outgoing packets are stored. This buffer is a shared resource among all the queues attached to every incoming and outgoing link of the node. All the traveling packets are subdivided into two classes: data and routing packets. Additionally, there are two priority levels in queues. Usually, data packets are served in the low-priority queues, while routing packets are served in the high-priority queues. The workload is defined in terms of applications whose arrival rate is given by a probabilistic model. By application (or session, or connection in the following), we mean a process sending data packets from an origin node to a destination node. The number of packets to send, their sizes, and the intervals between them are assigned according to some defined stochastic process. We do not make any distinction among nodes, which act at the same time as hosts (session endpoints) and gateways/routers (forwarding elements). The adopted workload model incorporates

In order to run experiments with AntNet, a complete network simulator was developed in C++ by Gianni Di Caro (Di Caro, 2003; Di Caro & Dorigo, 1998c). It is a discrete event simulator using as its main data structure an event list, which holds the next future events. The simulation time is a continuous variable and is set by the currently scheduled event. The aim of the simulator is to closely mirror the essential features of the concurrent and distributed behavior of a generic communication network without sacrificing efficiency and flexibility in code development in matlab.

IV. THE ANTNET ALGORITHM

AntSim, the routing algorithm is a direct extension of the Simple Ant Colony Optimization algorithm. As will become clear in the following, AntSim is even closer to the real ants' behavior that inspired the development of the ACO metaheuristic than the ACO algorithms for NP-hard problems. Informally, the AntSim algorithm and its main characteristics can be summarized as follows.

- At regular intervals, and concurrently with the data traffic, from each network node artificial ants are asynchronously launched toward destination nodes selected according to the traffic distribution
- Artificial ants act concurrently and independently, and communicate in an indirect way (through the pheromones they read and write locally on the nodes.
- Each artificial ant searches for a minimum cost path joining its source and destination node.
- Each artificial ant moves step by step toward its destination node. At each intermediate node a greedy stochastic policy is applied to choose the next node to move to. The policy makes use of (1) node-local artificial pheromones, (2) node-local problem-dependent heuristic information, and (3) the ant's memory.
- While moving, the artificial ants collect information about the time length, the congestion status, and the node identifiers of the followed path.
- Once they have arrived at the destination, the artificial ants go back to their source nodes by moving along the same path as before but in the opposite direction.
- During this backward travel, node-local models of the network status and the pheromones stored on each visited node are modified by the artificial ants as a function of the path they followed and of its goodness.
- Once they have returned to their source node, the artificial ants are deleted from the system.

A. AntSim: The Algorithm

AntNet is conveniently described in terms of two sets of artificial ants, called in the following forward and backward ants. Ants in each set possess the same structure, but they are differently situated in the environment; that is, they can sense different inputs and they can produce different, independent outputs. Ants communicate in an indirect way, according to the stigmergy paradigm, through the information they concurrently read and write on the network nodes they visit.

The AntSim algorithm, whose high-level description in pseudo-code is given in matlab, can be described as being composed of two main phases: solution construction and data structures update.

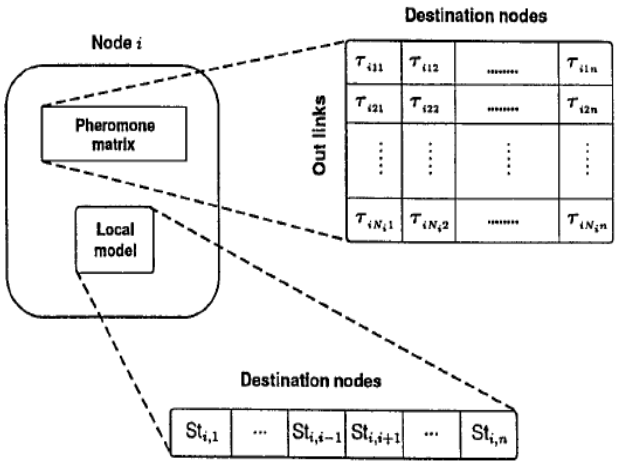


Figure 1: the artificial ants in AntNet for the case of a node i

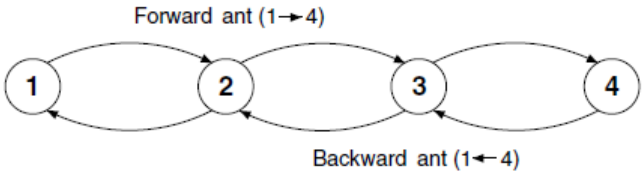


Figure 2: Antsim's ants update node structures

V. The Experimental Settings

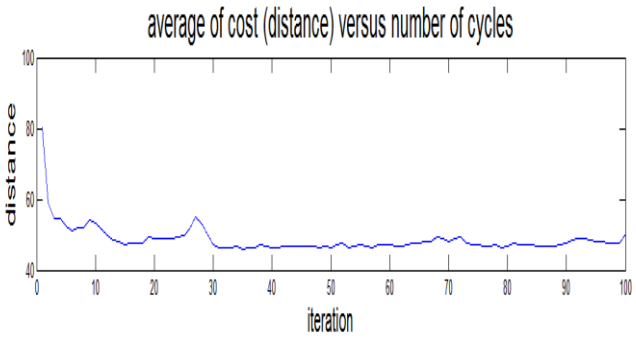
AntSim with some of the best-known routing algorithms. Note that, because the functioning of a data network is governed by many components which may interact in nonlinear and unpredictable ways, the choice of a meaningful test bed is not an easy task: the approach followed is to define a limited set of classes of tunable components. These are: the topology and the physical properties of the network, the traffic patterns, and the metrics chosen for performance evaluation, the competing routing algorithms chosen, and their parameter values. In the following, for each class the choices are explained.

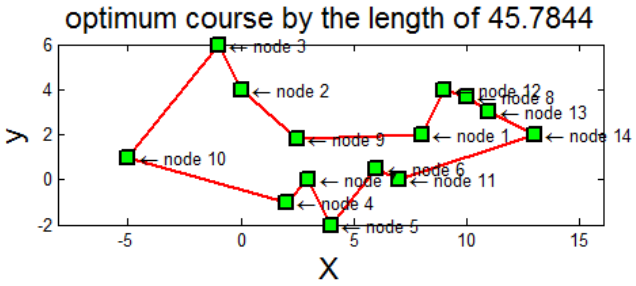
1. Topology and Physical Properties of the Net
2. Traffic Patterns
3. Metrics for Performance Evaluation

The performance of the algorithms was studied for increasing traffic load and for temporary saturation conditions. In the experiments reported here, the saturating input traffic, whose level is a function of the routing algorithm used, was determined using AntNet as routing algorithm.

All reported data are averaged over ten trials lasting 1000 virtual seconds of simulation time, which was found to be a time interval long enough to make effects due to transients negligible and to get enough statistical data to evaluate the behavior of the routing algorithm. Before being fed with data traffic, the algorithms are given 100 preliminary simulation seconds with no data

traffic to build initial routing. In this way, each algorithm builds the routing tables according to its own ‘‘vision’’ about minimum cost paths in relation to the physical characteristics of the network.





VI. Conclusion and Future Scope

In AntNet, the continual online adaptation of pheromone matrices (and therefore of the corresponding routing tables) is the emerging result of a collective learning process. In fact, each forward-backward ant pair is complex enough to find a good route and to adapt the pheromone matrices for a single-source destination path, but it cannot solve the global routing optimization problem. It is the interaction between the ants that determines the emergence of a global effective behavior from the point of view of network performance. Ants cooperate in their problem-solving activity by communicating in an indirect and noncoordinated asynchronous way. Each ant acts independently. Good routes are discovered by applying a policy that is a function of the information accessed through the network nodes visited, and the information collected about the route is eventually released on the same nodes. Therefore, communication among artificial ants is mediated in an explicit and implicit way by the ‘‘environment,’’ that is, by the node’s data structures and by the traffic patterns recursively generated by the data packets’ utilization of the routing.

ACO algorithms have been tested on a large number of academic problems. These include problems related to the traveling salesman, as well as assignment, scheduling, subset, and constraint satisfaction problems. For many of these, world-class performance has been achieved.

Today, several hundred papers have been written on the applications of ACO. It is a true metaheuristic, with dozens of application areas. While both the performance of ACO algorithms and our theoretical understanding of their working have significantly increased, as shown in previous chapters, there are several areas in which until now only preliminary steps have been taken and where much more research will have to be done.

One of these research areas is the extension of ACO algorithms to more complex optimization problems that include (1) dynamic problems, in which the instance data, such as objective function values, decision parameters, or constraints, may change while solving the problem; (2) stochastic problems, in which one has only probabilistic information about objective function value(s), decision variable values, or constraint boundaries, due to uncertainty, noise, approximation, or other factors; and (3) multiple objective problems, in which a multiple objective function evaluates competing criteria of solution quality.

Active research directions in ACO include also the effective parallelization of ACO algorithms and, on a more theoretical level, the understanding and characterization of the behavior of ACO algorithms while solving a problem.

References

1. Stutzle, T., & Dorigo, M. (2002). A short convergence proof for a class of ACO algorithms. IEEE Transactions on Evolutionary Computation, 6(4), 358–365.
2. Selvi, V., & Umarani, R. (2010). Comparative analysis of ant colony and particle swarm optimization techniques. International Journal of Computer Applications, 5(4).
3. Stutzle, T., & Linke, S. (2002). Experiments with variants of ant algorithms. Mathware and Soft Computing, 9(2–3), 193–207.
4. Lee, J. W., Lee, D. H., & Lee, J. J. (2011, May). Global path planning using improved ant colony optimization algorithm through bilateral cooperative exploration. Digital Ecosystems and Technologies Conference (DEST), 2011 Proceedings of the 5th (pp. 109-113). IEEE.
5. M. Christoudias, B. Georgescu, and P. Meer. Synergism in low level vision. In 16th International Conference on Pattern Recognition., Quebec City, Canada, volume IV, pages 150–155, 2002.
6. Randall, M., & Lewis, A. (2010, December). . Modifications and additions to ant colony optimisation to solve the set partitioning problem. e-Science Workshops, 2010 Sixth IEEE International Conference on (pp. 110-116). IEEE.
7. Van Ast, J. M., Babuška, R., & De Schutter, B. (2010). Ant colony learning algorithm for optimal control. Interactive Collaborative Information Systems, 155-182.
8. Socha, K., Knowles, J., & Sampels, M. (2002). A MAX-MIN Ant System for the university course timetabling problem. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms, vol. 2463 of Lecture Notes in Computer Science (pp. 1–13). Berlin, Springer-Verlag.
9. Merkle, D., Middendorf, M., & Schmeck, H. (2002). Ant colony optimization for resource-constrained project scheduling. IEEE Transactions on Evolutionary Computation, 6(4), 333–346.
10. Maniezzo, V., & Carbonaro, A. (2000). An ANTS heuristic for the frequency assignment problem. Future Generation Computer Systems, 16(8), 927–935.

AUTHOR(S) PROFILE

Mohit Malik, a P.G Scholar and pursuing M.Tech degree in Computer Science Engineering from the Sat Kabir Institute of Technology and Management (2013-15) from the Maharshi Dayanand University. He did the B.Tech degree in Computer Science Engineering from Matu Ram Institute of Engineering and Management in 2013 from the Maharshi Dayanand University Rohtak.

Kirti Bhatia, is currently working as an Assistant Professor in Department of Computer Science in Sat Kabir Institute of Technology and Management with a rich teaching experience. She is a Post Graduate and a Graduate in Computer Science Engineeering.

Suryakiran Suhag, is currently working as an Assistant Professor in Department of Computer Science in Sat Kabir Institute of Technology and Management. She is a Post Graduate in Computer Science Engineering from PDM College Bahadurgarh in 2012 and a Graduate in Computer Science Engineering from Shri Baba Mastnath College Rohtak in 2009.

