IJCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Volume-3, Issue-9 E-ISSN: 2347-2693

Priority Mechanism for ant Colony Optimization in Network Routing

Farhaan Jalia[1*] and Aruna Gawde[2]

[1*,2]*Department of Computer Engineering,*
Dwarkadas J. Sanghvi College of Engineering, India

www.ijcseonline.org

Received: Sep /02/2015 Revised: Sep/10/2015 Accepted: Sep/24/2015 Published: Sep/30/ 2015

***Abstract*—**Congestion, packet loss and increased response-time due to network traffic are common problems in most networks. This results in lowered network efficiency and poor Quality of Service (QoS). A number of routing protocols have been developed to deal with network traffic. The goal of every network routing protocol is to direct the traffic from source to destination maximizing the network performance. The Ant Colony Optimization (ACO) based routing protocol is efficient when used to dynamically route network traffic. Currently, there are many variations of the ACO algorithm in the domain of network routing. Past work has been done by researchers to improve the performance of the algorithm. In this paper we first study and analyze the existing work in this field and weigh the pros and cons of the different modifications and variations of the algorithm. We then propose a modification to the ACO algorithm in order to improve the quality of service offered by a network by routing packets according to their priority. Packets that belong to time-sensitive services like VOIP will be given higher priority and routed differently from low priority packets like FTP. By doing so the proposed algorithm will improve the success rate of the high priority packets while still maintaining high overall throughput of the network by dropping low priority packets that form loops. We then implement this algorithm on NS2 network simulator. The algorithm is then tested to see how it dynamically adapts to network changes. We then conduct tests to calculate the success rate and throughput that is offered by the algorithm and compare the results to those of other ACO algorithms. Our results indicate that the proposed algorithm improves the overall performance of the network by striking a balance between throughput and success rate thanks to the priority mechanism.

***Keywords*—** Computer Networks; Routing; QoS; Ant Colony Optimization; Swarm Intelligence

I. INTRODUCTION

ACO is based on the behavior of real ants in finding the shortest path from a source to the destination where food is available. The basic idea behind ACO algorithms for routing is the acquisition of routing information through the sampling of paths using small control packets called ants. The ants are generated concurrently and independently at the nodes, with the task to test a path from a source node to an assigned destination node. The ant collects information about the quality of the path - end-to-end delay, number of hops - and uses this on its way back from the destination to the source to update the routing information at the intermediate nodes and the source. The next set of ants or data packets can now learn from the pheromone deposit, ie, feedback left by these initial ants. The main characteristic of ACO is that after each iteration, the pheromone values are updated by all the ants that have reached the destination successfully. In ACO, the next node is selected dynamically, with the probability to choose the shortest path more. Thus, when the number of packets increases, packets may follow alternate paths to control congestion.

It has been found that the current ACO algorithms have a trade off between throughput and success rate. ACO algorithms exhibiting a high throughput have a low success rate and those with a high success rate have low throughput. Both parameters hold immense importance in evaluating network performance.

II. EXISTING SYSTEM

The optimal performance of a network is extremely important for any network. Routing of packets in a network does not only involve finding the shortest path from the source to the destination but finding the best path from the source to the destination. Avoiding congestion, increasing throughput, greater success rate, avoiding packet loss, reducing delay, are just some of the factors to be considered to evaluate network performance.

Ant Colony Optimization based Routing Algorithm is an emerging algorithm which has proved to be extremely effective.

ACO Algorithm is based on the behavior exhibited by real ants while finding the shortest path to the food. It has been observed that ants deposit a certain amount of pheromone in its path while travelling from their nest to the food. Again while returning, the ants are subjected to follow the same path marked by the pheromone deposit and again deposit the pheromone in its path. In this way, the ants following the shorter path are expected to return earlier and hence increase the amount of pheromone deposit in its path at a faster rate than the ant following a longer path.

The ants increasingly begin to use the paths with greater deposits of pheromone at the same time and the longer path with lesser deposits of pheromone are soon lost due to evaporation of the pheromone. Thus, all ants starting their journey can learn from the information left by the previously visiting ants and are guided to follow a shorter path directed by the pheromone deposit.

The ACO algorithm is explained in [1] as follows:
A number of artificial ants (packets) are simulated from a source to the destination. The forward ants select the next node randomly for the first time taking the information from the routing table and the ants who are successful in reaching the destination update the pheromone deposit at the edges visited by them by an amount (C/L), where 'L' is the total path length of the ant and C a constant value that is adjusted according to the experimental conditions to the optimum value. The next set of the ants can now learn from the pheromone deposit feedback left by the previously visited successful ants and will be guided to follow the shortest path. The probability of selecting a node j from node i is given by

$$p_{ij} = \frac{\tau_{ij}^{\alpha} . \eta_{ij}^{\beta}}{\sum \tau_{ij}^{\alpha} . \eta_{ij}^{\beta}}$$

if a link exists between nodes i and j or

$p_{ij} = 0$, if there is no link between nodes i and j

where, p_{ij} is the probability of selecting a node i from node j,

τ_{ij} is the pheromone associated with the path joining node i and node j,

$\eta_{ij} =(1/ d_{ij})$, where d_{ij} is the distance between the nodes i and j and

α and β are parameters that controls the relative importance of the pheromone versus the heuristic information η_{ij}.

After each iteration, the pheromone values are updated by all the number of ants (packets) that have reached to the destination successfully and found a solution in the iteration itself. The pheromone value τ_{ij} while traveling from node i to node j is updated as follows:

$$\tau_{ij} = (1 - \rho). \tau_{ij} + \sum_{i=1}^{m} \tau_{ij}^{k}$$

where ρ is the evaporation rate, m is the total number of successful ants (packets) and $\Delta\tau_{ij}$ is the quantity of pheromone laid on edge (i,j) by packet k.

ACO Algorithm can be implemented with 3 variations [1]:

a. **ACO1**: In this method, the ant packets are not allowed to visit a node that it has already visited before, i.e., the ant packets are not allowed to form loops. If a packet reaches a state that it has no other way except to form a loop, the packet is discarded.

b. **ACO2**: In this method, the ant packets are allowed to form loops and visit an already visited node. However, they cannot visit the node last visited by it. In this method, to prevent a packet from going into an infinite loop, if the packet has not reached the destination after a certain interval of time, it should be marked as unsuccessful.

c. **ACO3**: ACO2 is modified with the restriction that the ant packet will not visit the last n nodes already visited by it. A Tabu list[3] is maintained to keep a list of the last n nodes visited.

The ACO algorithm has been successfully applied to discrete optimization problems such as the Travelling Salesperson Problem, Graph Colouring, Scheduling etc. ACO algorithm when applied to the Network Routing Problem also, outperformed the traditional algorithms such as Open Shortest Path First (OSPF), Shortest Path First (SPF), Bellman-Ford (BF), Q-routing and PQ routing. AntNet was found to perform better, both in terms of throughput and distribution of packet delay. It also showed a robust behaviour under the different traffic conditions and the ability to reach a stable behaviour very quickly. [5]

In spite of the many advantages of ACO routing algorithm, it was found to have some drawbacks based on its different implementations [1]:
ACO1 was found to have a high throughput but a low success rate since a large number of packets are discarded and hence, do not reach the destination.

On the other hand, ACO2 and ACO3 have a high success rate since the packets are allowed to form loops and thus, have a higher probability to reach the destination. But they have a low throughput due to the cycles formed by the packets.

This is summarised in Table 1.

TABLE I
COMPARISON OF THE ALGORITHMS

ACO 1	Highest throughput	Low success rate
ACO2	Low throughput	High success rate
ACO 3	Low throughput	High success rate

III. MODIFICATIONS TO IMPROVE PERFORMANCE OF ACO

Various modifications have been suggested to the ACO algorithm over time to improve its performance. Some of them are enlisted below [2].

A. *Non-linear Ant Launching:*

In many ACO schemes, it was proposed that ants should be launched at constant intervals. However, in such a scheme it may occur that a small number of ants are not enough to obtain a solution whereas the performance of the system degrades if the number of ants launched is too large.

A more appropriate approach would be to increase the number of ants being launched as the network becomes more and more congested. This facilitates the discovery of alternative uncongested routes when experiencing heavy traffic conditions. Similarly, under low traffic conditions, fewer ants should be launched allowing more stable routing tables.

Thus, in this modification, ants are to be launched at a rate proportional to the current traffic load of the network. The ants are launched based on the queue size of the nodes, which reflect the degree of traffic experienced. To prevent a node from launching tee few ants, a maximum time interval between two consecutive ants is to be adopted. Similarly, to prevent an excessive number of ants from being launched, a minimum interval threshold is also set.

B. *Verification Mechanism*

Usually according to the ACO algorithm packets are forwarded according to the entries in the pheromone table, and the condition of the link is not considered. Thus if a link with maximum pheromone value is currently unavailable, the packet would still be forwarded on it and be lost in the process.
The proposed algorithm adopts a verification mechanism which checks the state of a link before forwarding a packet to it. If the link is available, data packets are forwarded according to the highest entry in the pheromone table, while if the link is not available then the next best available link is chosen.
The verification function continuously checks the status of the links using both artificial ants and data packets. In this way ants are able to rapidly interact with sudden network changes caused by link failures as ants are not allowed to choose unavailable links and therefore alternative available links are enforced and made available for future communication. Therefore the proposed algorithm adapts immediately to rapid topology changes.

C. *Priority Scheduling*

The traditional ACO schemes do not distinguish between the different data types flowing over a network. This means that all data packets are routed in the same way, as these techniques are mainly designed to reduce congestion and improve network performance. However, modern communication networks have to support multimedia data flows, requiring a different level of QoS.

The proposed method is able to discriminate between the different data flows, therefore providing a better QoS. This was achieved by assigning higher priorities to delay intolerant information such as VoIP and Videoconferencing applications, while other content such as FTP is assigned a lower priority. The priority levels are used to determine the order in which packets are scheduled once they arrive at the destination node. A pre-emptive priority scheduling scheme was adopted.

The packets representing the artificial ants were set to a medium priority. In this way, ants would still be affected by the congestion at the nodes, penalizing ants choosing congested paths, while at the same time ants are still processed at an acceptable rate.

D. *Smart Initialization*

The pheromone tables adopted in ACO methods are generally initiated at random. This means that initially a setup time is required. The proposed algorithm eliminates this converging period through the use of a smart initialization process. To reduce the convergence period, a scheme that assigns an initial greater probability value to the neighbouring nodes, when such nodes are listed as destination is used.

E. *Fixed Sized Ants*

A previous ACO scheme enforced artificial ants to upgrade all the entries in the routing table corresponding to all the intermediate nodes visited by the ants. This enables the routing tables to adapt more rapidly to network variations. However, ants were required to remember all the addresses of the visited nodes and the time at which such visit occurred, causing the ants to grow proportionally to the number of nodes visited.
This dynamicity in the ant's size is undesirable, since varying size packets require a longer processing time than fixed sized packets.
The proposed algorithm defines fixed sized ants. This implies that the number of updates performed at a routing

table by a single ant would be limited to a predefined number. In this way, ants still retain the possibility of multiple updates, therefore being responsive to topology changes, whilst retaining fixed sized ants for faster processing.

IV. Proposed System

The main aim of this work is to improve the performance of the ACO Routing Algorithm. For this, we will be making use of the proposed modifications and combining it with the existing algorithm[4] to attain an optimum variation that enables it to perform at its best.

In ACO1, ACO2 and ACO3 mentioned above, all packets are assumed to have the same priority. Thus in ACO1, which has a low success rate, packets are discarded without taking into consideration its priority. However, this would serve as a major setback in networks which deal with packets from various applications. In such a network, if packets belonging to delay intolerant applications such as VoIP and Videoconferencing are discarded upon reaching a dead end (ACO1), or they are delayed due to the formation of a sizeable loop (ACO3), it will lead to performance degradation of that application. Thus, packets of such applications must be given a higher priority as compared to applications such as FTP.

In order to deal with this issue, we have implemented the Priority Scheduling Modification proposed for the ACO algorithm. The main was to implement a Priority Mechanism in combination with the ACO1 and ACO2 algorithms using NS2[6].

The packets being routed in the network will be assigned a priority based on their application. These packets will be scheduled depending on their priorities. In addition, if a packet reaches a dead end where no other possible path is available except forming a node, its priority should be checked. If it has a high priority, it may be allowed to form a loop in order to prevent it from being discarded. However, if the packet has a low priority, it can be discarded.

This modification will help keep the throughput as well as the success rate considerably high. Since high priority packets are not discarded on reaching a dead end, it increases the success rate. On the other hand, all packets are not allowed to form loops thus, the total time taken for the packets to reach the destination is less leading to an increase in throughput.

We have implemented this modification to the ACO algorithm and analysed its performance under various network conditions such as constant load, varying load, different network sizes etc. The focus is mainly on analysing the throughput and success rate of the algorithm through comparative studies with the original ACO algorithm.

A. *Algorithm*

The basic algorithm will be as follows:

a) At regular intervals of time forward ants are generated from a randomly selected source to a randomly selected destination.

b) This ant will read the routing table of the source node and find out the pheromone values of the path from this source node to its neighbors corresponding to the destination.

c) If it is the first ant, all the pheromone values will be equal. Therefore, it will randomly select the next node.

d) If it is not the first ant, it will select the next node based on the pheromone values. It will choose the node with the maximum pheromone value along the path to it.

e) On reaching the next node, it pushes the node and the time elapsed since the ant left the source onto its stack.

f) It then checks the routing table of the current node to make its next routing decision.

g) Carrying on in this way, the ant reaches its destination.

h) Now, a backward ant is generated. This backward ant follows the same path that the forward ant followed.

i) The information regarding the path of the forward ant in its stack is then transferred to the backward ant.

j) If there are loops in the forward ant's path, all the loops are popped from the ant's stack so that the path in the backward ant's stack does not contain any loops.

k) At every intermediate node, the backward ant updates the pheromone value in the routing table according the trip time of the forward ant.

l) The shorter the trip time, the better the path. Hence the magnitude by which the pheromone value is increased is higher for shorter paths and lower for longer paths.

m) Since the backward ant updates routing information, it should not be delayed. If it is delayed, the routing information that it carries will become outdated and the performance of the algorithm will deteriorate. Thus, backward ants have the highest priority. They should not be made to wait at the intermediate nodes.

n) When the backward ant reaches the source, its journey ends and it is destroyed.

B. *Implementation of Priority Mechanism*

a) When actual data packets are sent from source to destination, the priority mechanism will come into play.

b) Packets that have time-sensitive services will have a high priority. Other packets will have low priority. Forward ant packets will be given medium priority whereas backward ants are given highest priority.

c) While making routing decisions if a condition occurs such that the packet needs to form a loop, the priority of the packet will determine whether or not the packet should be allowed to loop.

d) High priority packets will be allowed to form loops in the hope that they will eventually find another path to the destination and break out of the loop.

e) Low priority packets will not be allowed to form loops. They will be simply discarded and their journey will end.

f) This priority mechanism will ensure that the success rate for high priority packets will be high, without hampering the throughput of the network.

Once tested under various conditions, it will be possible for us to reach to a conclusion regarding the performance of a network using the modified ACO Routing Algorithm proposed by us.

V. Experimental Results

We first simulated a few links going down to test how quickly the ACO algorithm adapts to such events. In the diagrams below, the red packets are data packets travelling from node 0 to node 10.

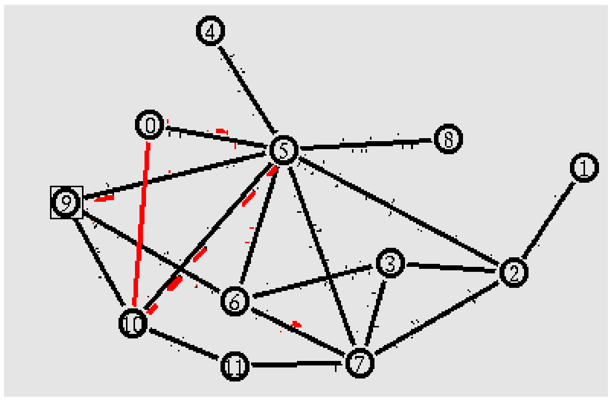

Figure 1 The network topology on NS2

The direct link connecting node 0 and node 10 is down. Therefore, the packets are following path 0-5-10 instead.

Now a few seconds later, link 5-10 goes done as well, thus cutting of the second shortest path from node 0 to node 10. After a couple of seconds the algorithm dynamically adapts to this change and routes the packets via node 9 or node 6.

We then bring link 0-10 up again. Therefore, once the ants have sufficiently updated the pheromone values, the data packets follow directly from node 5 to node 10.

After this all links are brought up. The algorithm adapts to this and routes most of the packets from node 0 directly to node 10. As time goes by, the pheromone values are reinforced and more and more packets are sent along the shortest path 0-10.

We then ran tests to evaluate the performance of the proposed algorithm in comparison with the existing algorithms. The tests were run on the same network topology under the exact same conditions. The success rate of packets and the overall throughput of the network were calculated. Multiple runs were taken to establish repeatability of results.

TABLE II
PERFORMANCE COMPARISON

	ACO1	ACO2	Our Algorithm
Success Rate	87-90%	93-95%	High Priority – 93-95% Low Priority – 87-90% Overall – 91-93
Throughput	60KBps	55KBps	58 KBps

The above table shows the performance of our algorithm in comparison to ACO1 and ACO2[1]. As can be seen, the algorithm works as we had envisioned when we proposed the algorithm.

The success rate of high priority packets is very high. It is same as the success rate provided by ACO2. The success rate of low priority packets is lower and is comparable to the success rate provided by ACO1.

When it comes to throughput, due to the combination of ACO1 and ACO2, the total throughput provided by our algorithm in our test network was higher than that provided by ACO2 and less than that provided by ACO1. Thus, in contrast to ACO1 and ACO2 our algorithm provides a balance between throughput and success rate.

VI. Conclusion

In this paper we first studied the variants of the ACO algorithm. We looked at the original ACO algorithm and studied the various modifications like non-linear ant launching, verification mechanism, priority scheduling, smart initialization and fixed sized ants. We have then proposed and implemented a priority scheme to route the packets to overcome the trade-off between success rate and throughput of the traditional ACO algorithm. The performance of this algorithm was analyzed on NS2 and it was found to improve the performance of ACO algorithm in network routing.

ACKNOWLEDGMENTS

We thank the management of D.J. Sanghvi College of Engineering and our Principal**,** for providing us with all the facilities needed to complete the project.

We would also like to express our gratitude for the teaching and non-teaching staff of Computer Department of D. J. Sanghvi College of Engineering, and thank them for their co-operation and assistance.

REFERENCES

[1] Debasmita Mukherjee and Sriyankar Acharyya, "Ant Colony Optimization Technique Applied in Network Routing Problem", International Journal of Computer Applications, Volume-**01**, Issue-**15**, Page no (**66-73**), May **2012**

[2] Chris Saliba and Reuben A. Farrugia, "Quality of Service Aware Ant Colony Optimization Routing Algorithm", 15th IEEE Mediterranean Electrotechnical Conference, ISBN: **978-1-4244-5793-9**, Page no (**343-347**) , April 26-28, **2010**.

[3] Masaya Yoshikawa and Kazuo Otani, "Ant Colony Optimization Routing Algorithm with Tabu Search", Proceedings of the International Mulitconference of Engineers and Computer Scientists 2010, Volume – **III**, ISBN: **978-988-18210-5-8**, Page no (**112-117**) March 17-19, **2010**.

[4] Vincent Verstraete, Matthias Strobbe, Erik Van Breusegem, Jan Coppens, Mario Pickavet and Piet Demeester, "AntNet: ACO routing algorithm in practice", Proceedings of the 8e INFORMS Telecommunications Conference, **2006**.

[5] Gianni Di Caro and Marco Dorigo, "Ant colonies for Adaptive Routing in Packet-Switched Communications Networks", Lecture Notes in Computer Science, Volume **1498**, Page no (**673-682**), June **2006**.

[6] The Network Simulator – NS2, http://www.isi.edu/nsnam/ns/, August **2014**.

[7] V. Laxmi, Lavina Jain and M.S. Gaur, "Ant Colony Optimization Based Routing on NS-2", International Conference on Wireless Communication and Sensor Networks (WCSN), India, December **2006**.

[8] Gianni Di Caro and Marco Dorigo, "AntNet : Distributed Stigmergetic Control For Communications Network", Journal of Artificial Intelligence Research, Volume-**09**, Issue-**01**, ISSN **1076–9757**, Page no (**317-365**), August **1998**.