Ant Colony Optimization Technique Applied in Network Routing Problem

Debasmita Mukherjee
West Bengal University of Technology
BF-142, Sector-1, Salt lake, Kolkata- 700 064

Sriyankar Acharyya
West Bengal University of Technology
BF-142, Sector-1, Salt lake, Kolkata- 700 064

ABSTRACT:
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. Three variations of the Ant Colony Optimization Technique, ACO1, ACO2 and ACO3 has been proposed and applied on different standard network models and the results has been analyzed and concluded. A Tabu list is also maintained for a network with large number of nodes and results were collected to find the optimum size of the Tabu list in one of the algorithms proposed here. Experiments have also been performed by varying the load of the network. Here the throughput and the reliability of the network has been specially taken as the performance factor of the network.

Categories and Subject Descriptors
Network Protocols and Management, Optimization Techniques.

General Terms
Algorithms, Management, Performance.

Keywords
Ant Colony, Optimization, Routing, Throughput, Tabu list, Delay factor, Pheromone.

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[12] to find the shortest path and also to increase the throughput of the network. In this paper, a routing algorithm has been proposed which may be applied in large network structure with heavy load. The algorithm proposed here is selecting the path of the data packets guided by the shortest path but with a probabilistic nature. The goal of every network routing algorithm is to direct the traffic from source to the destination maximizing the network performance. The performance measure that is usually taken into account is the throughput (bits delivered per unit time) and number of packets successfully reaching the destination. Here the performance measure is taken as the throughput and also the number of packets successfully reaching the destination, i.e., the reliability of the network. This paper is divided into seven Sections. Section 1 gives the introduction of the paper, in Section 2 there is a brief explanation of the ACO, Section 3 explains how ACO can be applied in the routing problems of the network. It also explains the working methodologies of the three modifications of the ACO that is ACO1, ACO2 and ACO3 that are proposed in this paper, Section 4 contains the algorithms of the three modifications of the ACO, Section 5 deals with the results of the experiments that are being performed based on some of the standard network topologies. Section 6 contains the discussion on the results and conclusion and also some future scope of work. Section 7 contains the references used in this paper.

2. ACO
ACO[1],[5],[6] is an algorithm based on the behaviour of the real ants in finding a shortest path from a source to the food. This algorithm utilizes the behaviour of the real ants while searching for the food. It has been observed that the ants deposit a certain amount of pheromone in its path while traveling from its 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 ants following a longer path. ACO takes the inspiration from the foraging behaviour of the ants. These ants deposit pheromone on the ground in order to mark some favourable path that should be followed by other members of the colony. However, the pheromone is subjected to evaporation by a certain amount at a constant rate after a certain interval and therefore the paths visited by the ants frequently, are only kept as marked by the pheromone deposit, whereas the paths rarely visited by the ants are lost because of the lack of phreromone deposit on that path and as a result the new ants are intended to follow the frequently used paths only. Thus, all the ants starting their journey can learn from the information left by the previously visitor ants and are guided to follow the shorter path directed by the pheromone deposit. In ACO, a number of artificial ants(here data packets) build solutions to the considered optimization problem at hand and exchange information on the quality of these solutions via a communication scheme that is pheromone deposit on the path of the journey performed by it.

3. ACO IN NETWORK ROUTING PROBLEMS:
ACO algorithms can be applied in the network routing problems to find the shortest path[2]. In a network routing problem, a set of artificial ants(packets) are simulated from a source to the

destination. The forward ants[4] are selecting 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 are updating the pheromone deposit at the edges visited by them by an amount ($^{C}/_{L}$), where 'L' is the total pathlength 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[9] 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

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

if a link exists between nodes i and j or

$$p_{ij} = 0,$$

if there is no link between nodes i and j or

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}.

The main characteristics of the ACO is that, 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:

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

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. Here the roulette wheel selection[13] method has been used to select the next node in the path.

One more advantage of the application of ACO in network routing problem is that when the number of packets increases, this algorithm can be applied for controlling the congestion also. In static routing algorithm all the packets from a given source to destination are bound to follow a constant path as calculated by the algorithm and therefore may give arise to the problem of congestion when the number of packets increases. As a result some of the packets may have to wait to be transferred. But in case of ACO, as the next node is selected dynamically and randomly, with the probability to choose the shortest path more, so some of the packets may follow some other path which increases the net throughput of the network when the number of packets in the network increases.

In this paper three variations of the ACO algorithms that is ACO1, ACO2 and ACO3 has been suggested and applied over different network models as discussed below:

a) ACO1: In the first method, no ants(packets)[4] are allowed to visit a node that is already visited by it in its journey that is no ants(packets)[are allowed to make a loop in its path. While selecting the next node, it is checking that the node has already been visited by it or not. If the node is an already visited one, then just discard the node and check for the other available nodes. If a packet reaches in a state such that it has no other way except to select an already visited node, then stop the packet and mark the packet as unsuccessful.

b) ACO2: In the second method, the packets are allowed to make loops in their paths and visit an already visited node provided that it will not visit only the last node visited by it. However, this method may give rise to a situation where an ant(packet)may fall in an infinite loop. To avoid this situation after a certain interval of time if an ant(packet)has not reached the destination, the packet is marked as unsuccessful and its journey is forcefully stopped. In this method, after the completion of the journey of the successful packets, the paths of the packets are checked and made free of any of the loops that may have formed during the course of its journey, and then the pheromone deposit of the nodes of the loop free paths are updated.

c) **c) ACO3:** When the network model is large enough, the restriction of not visiting the last visited node only can be little modified with the restriction that the ants(packets)will not visit the last n number of nodes already visited by it. This modification is done to reduce the chances of forming loops in the path of the journey and hence reducing the overhead of removing the cycles from the path of the successful packets after the completion of its journey. Here also the journey of the ants(packets)are suspended after a certain interval of time and the packets failed to reach the destination are marked as unsuccessful. In this mtehod, a Tabu list is maintained to keep the list of the last n number of nodes visited by a packet so that the last n number of nodes are not selected while selecting the next node for going to the destination. Here experiments are carried out for large networks by varying the size of this Tabu list to find the optimum size of the Tabu list with the given number of nodes in the network model.

4. ALGORITHMS:

Method1:

```
Procedure ACO1(source, destination) {
    assign initial positions to a set of ants(packets) at the
    given source;
    while(source != destination) {
        call findnextnode(source);
        set newsource = selected node;
        source = newsource;
    }
    update the pheromone table using the paths selected by the
    successful packets;
    send a next set of packets(ants) guided by the information
    left by all the previous visitor ants(packets);
}
```

```
Procedure findnextnode(source) {
   check the routing table entries corresponding to the source
   if a link exists with the source {
      if (the node is already visited) {
         cancel the node;
      else if (the node not already visited) {
         mark the node as eligible of being selection
      }
   }
   using the pheromone information, select a node
   from the list of eligible nodes;
   return the selected node;
}
```

Method 2:

```
Procedure ACO2(source, destination) {
      assign initial positions to a set of ants(packets) at the
      given source;
      while(source != destination && no_of_hops <=maxhops) {
            call findnextnode(source);
            set newsource = selected node;
            source = newsource;
      }
      Remove all the cycles that may have formed in the path of
      the packets(ants);
      Update the pheromone table with the cycle free paths
      selected by the successful packets;
      Send a next set of ants(packets) followed by the information
      left by all the previous visited ants(packets);
}
```

```
Procedure findnextnode(source) {
   check the routing table entries corresponding to the source
   if (a link exists with the source){
      if (the node is the last visited node) {
         cancel the node;
      else if ( not the last visited node),) {
         mark the node as eligible of being selection
      }
   }
   using the pheromone information, select a node
   from the list of eligible nodes;
   return the selected node;
}
```

Method 3:

```
Procedure ACO3(source, destination) {
      assign initial positions to a set of ants(packets) at the
      given source;
      while(source != destination && no_of_hops <=maxhops) {
            call findnextnode(source);
            set newsource = selected node;
            source = newsource;
      }
      Remove all the cycles that may have formed in the path of
      the packets(ants);
      Update the pheromone table with the cycle free paths
      selected by the successful packets;
      Send a next set of ants(packets) followed by the information
      left by all the previous visited ants(packets);
}
```

```
Procedure findnextnode(source) {
   check the routing table entries corresponding to the source
   if (a link exists with the source){
      if (the node is the last visited node) {
         cancel the node;
      else if (the node is among the last n visited nodes) {
         mark the node as eligible of being selection
      }
   }
   using the pheromone information, select a node
   from the list of eligible nodes;
   return the selected node;
```

5. RESULTS

a) With Constant load:

Experiments has been done with four different network models and the results are summarized below: It can be seen that the throughput achieved is maximum in ACO1 for all of the experimental set up. In ACO1 and ACO2, the possibility of making the cycles increases and hence the total time taken to reach the destination also increases. Also, as we have considered that there is a certain delay factor for transmitting the packets from one node to another node, the time consumed in reaching the destination increases as there is the probability of making the cycles and hence the throughput falls for ACO2 and ACO3.

Table 1: Variation of throughput with constant load

No. of nodes	No. of packets	Method	Success (%)	Throughput (Mbps)
8	28000	ACO1	97.04	164.837

		ACO2	99.96	137.359
		ACO3	99.99	127.363
11	55000	ACO1	95.93	106.698
		ACO2	99.99	90.994
		ACO3	99.99	83.409
14	91000	ACO1	96.61	76.194
		ACO2	99.88	69.826
		ACO3	99.89	66.175
24	110400	ACO1	89.65	22.733
		ACO2	98.82	21.821
		ACO3	99.04	19.861

It can be seen that the throughput achieved is maximum in ACO1 for all of the experimental set up. In ACO2 and ACO3, the possibility of making the cycles increases and hence the total time taken to reach the destination also increases. Also, as we have considered that there is a certain delay factor for transmitting the packets from one node to another node, the time consumed in reaching the destination increases as there is the probability of making the cycles and hence the throughput falls for ACO2 and ACO3. In ACO,1 we are checking the ants are not making any cycles in its path whenever it is going to select a new node in its journey, thereby eliminating the extra time consumed in removing the cycles formed in the path as in the case of ACO2 and ACO3 and also the extra time consumed due to the delay factor of transmitting the packets from one node to another(which is considered to have a constant value in all the experimental set up). So, for any number of nodes present in the network, it has been found that ACO1 is best from the point of view of the throughput

However, if we consider from the point of view of the number of packets that has been reached successfully in its destination, ACO2 and ACO3 are better because in ACO1 whenever an ant is following a path, that has no other solution except to form a cycle, we are canceling the journey of the ant and hence reducing the number of successful packets which is a serious drawback of ACO1 when all the packets in the network holds equal importance. (This problem can be minimized if we simulate more number of ants for the journey because the more number of ants(packets) will deposit more amount of pheromone in the correct path and thereby guiding the rest of the ants followed by them to follow the paths which are free of cycles. So, the possibility of canceling the packets for its journey is also minimized). In ACO2 and ACO3, the packets are given more freedom to select any of the nodes randomly in its path and therefore the possibility of the cancellation of the journey reduces here in comparison to ACO1.

b) Results by varying the loads:

Experiments are also carried out by varying the load of the network with the above three methods of ACO with different network models and their results are summarized below(here the possibilities of the transfer of the packets from any source to destination, except with the same source and destination has been taken into account):

Table 2: Variation of throughput by varying the load(with 8 nodes)

No. of Nodes	No. of Packets	Method	Success(%)	Throughput (Mbps)
8	22400	ACO1	97.12	136.965
		ACO2	99.98	135.869
		ACO3	99.94	101.865
	25200	ACO1	97.01	148.305
		ACO2	99.98	136.855
		ACO3	99.98	114.641
	28000	ACO1	97.04	164.837
		ACO2	99.96	137.359
		ACO3	99.99	127.363
	30800	ACO1	97.13	181.490
		ACO2	99.97	140.108
		ACO3	99.97	132.099
	33600	ACO1	97.22	198.173
		ACO2	99.98	152.843
		ACO3	99.98	141.282

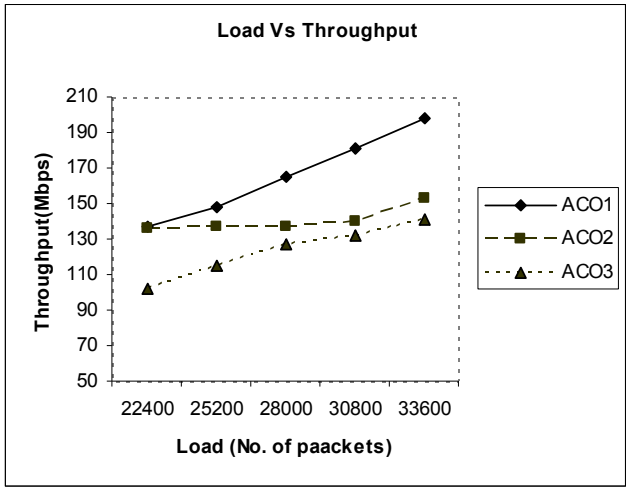

Figure 2.1: Variation of throughput with load corresponding to table 2

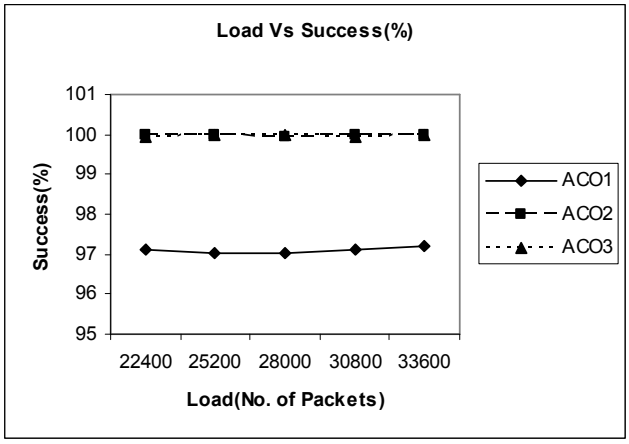

Figure 2.2: Variation of Percentage of Success Packets with load corresponding to table 2.

From Table 2, where the size of the network is smallest, we can see that the percent of successful packets for ACO2 and ACO3 are almost same and remains constant on increasing the load of the network also. So, we can conclude that on increasing the load of the network, the performance of the network will remain unchanged in both ACO2 and ACO3. In ACO1, the percent of successful packets is less than in ACO2 and ACO3, but in this case also the success percentage of the packets remains independent on the size of the network. There is variation however in the throughput(Figure 2.2). The throughput for ACO1 is gradually increasing with increasing the load of the network. In ACO2 also the throughput is increasing gradually with the load.

Table 3: Variation of throughput by varying the load(with 11 nodes)

No. of Nodes	No. of Packets		Success(%)	Throughput (Mbps)
11	44000	ACO1	95.96	96.055
		ACO2	99.99	80.076
		ACO3	99.99	73.796
	49500	ACO1	95.98	108.080
		ACO2	99.99	81.896
		ACO3	99.99	80.086
	55000	ACO1	95.93	106.698
		ACO2	99.99	90.994
		ACO3	99.99	83.409
	60500	ACO1	95.90	109.594
		ACO2	99.99	91.752
		ACO3	99.99	84.694
	66000	ACO1	95.91	115.213
		ACO2	99.99	92.399
		ACO3	99.99	100.09

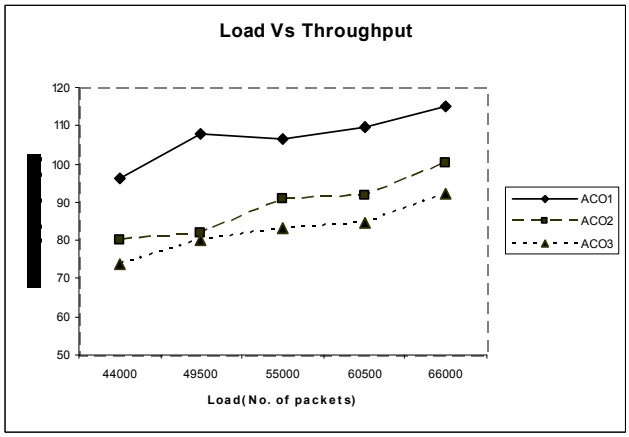

Figure 3.1: Variation of throughput with load corresponding to table 3

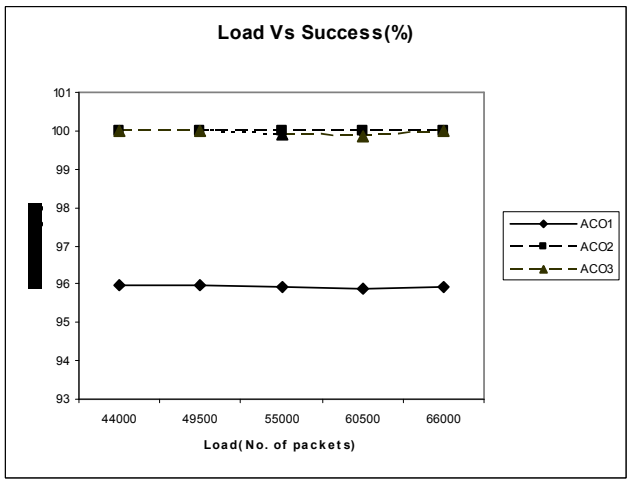

Figure 3.2: Variation of Percentage of Success Packets with load corresponding to table 3.

From Table3, we can see that in ACO2, the percentage of successful packets is same for the methods of ACO 2and ACO3 and remains almost constant with the variation of the load of the network. So, again we can conclude that the performance of the network remains unchanged with the increase of the load. However it can be seen that the percentage of successful packets is least in case of ACO1(Figure 3.2) but remains with the increase of load. It can also be concluded that the throughput is increasing with increasing the load in all the three methods. However, the throughput of ACO1 is always more than ACO2 and ACO3.and the throughput of ACO2 is always more than the throughput of ACO3 independent on the load of the network.

Table 4: Variation of throughput by varying the load(with 14 nodes)

No. of Nodes	No. of Packets		Success(%)	Throughput (Mbps)
14	72800	ACO1	96.53	71.058
		ACO2	99.88	60.155
		ACO3	99.87	63.012
	81900	ACO1	96.66	72.041
		ACO2	99.87	64.723
		ACO3	99.89	64.738
	91000	ACO1	96.61	76.194
		ACO2	99.88	69.826
		ACO3	99.89	66.175
	100100	ACO1	96.65	83.847
		ACO2	99.86	72.776
		ACO3	99.86	69.978
	109200	ACO1	96.99	87.621
		ACO2	99.88	76.351
		ACO3	99.89	70.906

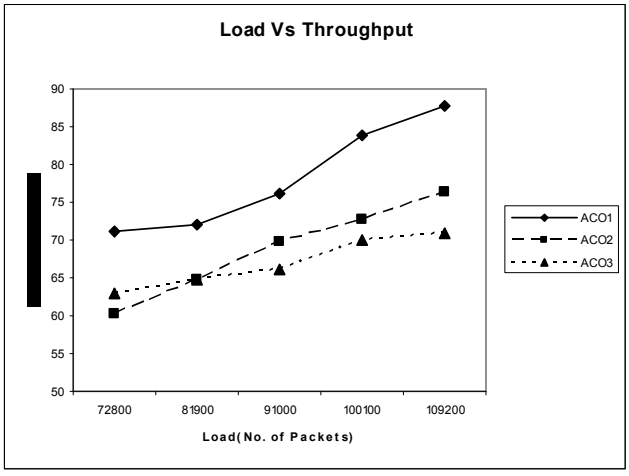

Figure 4.1: Variation of throughput with load corresponding to table 4

From Table 4, when the number of nodes increases, that is when the number of nodes are 14, it can be seen that the percentage of successful packets is almost same in ACO2 and ACO3 and least in the case of ACO1. Also the percentage of successful packets remains almost constant in each of the methods being independent on the load of the network as in the previous two cases. The throughput of the network increases in all the three methods. But however, the rate of increase of the throughput is less from the above two cases. From ACO2 and ACO3, it can be seen that initially, when the number of packets are less, the throughput of ACO2 is less than in ACO3. But with the increase of the load, the although the throughput of both of the methods increases, but the rate of increment of ACO2 is more than in ACO3 and gradually with the increase of load, the throughput of ACO3 becomes less than of ACO2.

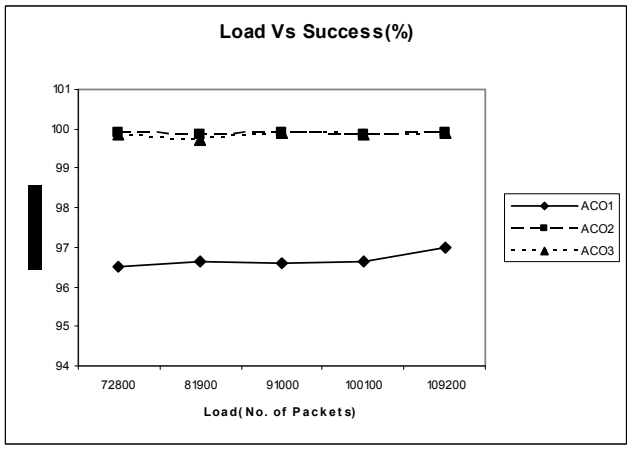

Figure 4.2: Variation of Percentage of Success Packets with load corresponding to table 4.

Table 5: Variation of throughput by varying the load

No. of Nodes	No. of Packets		Success(%)	Throughput (Mbps)
24	88320	ACO1	88.81	21.307
		ACO2	98.86	18.918
		ACO3	99.07	18.305
	99360	ACO1	89.28	23.400
		ACO2	98.54	20.084
		ACO3	99.06	19.685
	110400	ACO1	89.65	22.733
		ACO2	98.82	21.821
		ACO3	99.04	19.861
	115920	ACO1	89.83	26.641
		ACO2	98.86	22.67
		ACO3	99.05	20.86
	121440	ACO1	89.98	28.012
		ACO2	98.86	23.750
		ACO3	99.06	22.807

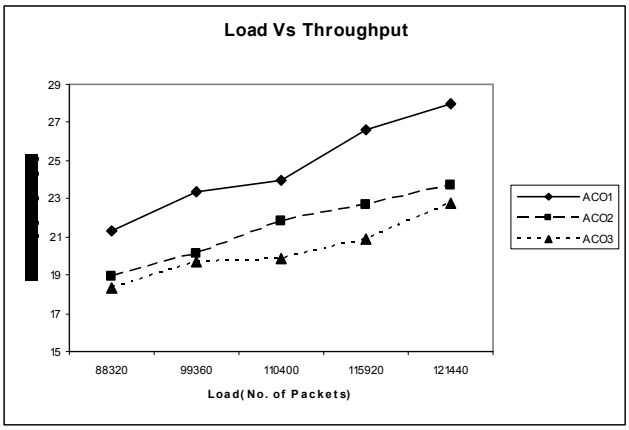

Figure 5.1: Variation of throughput with load corresponding to table 5

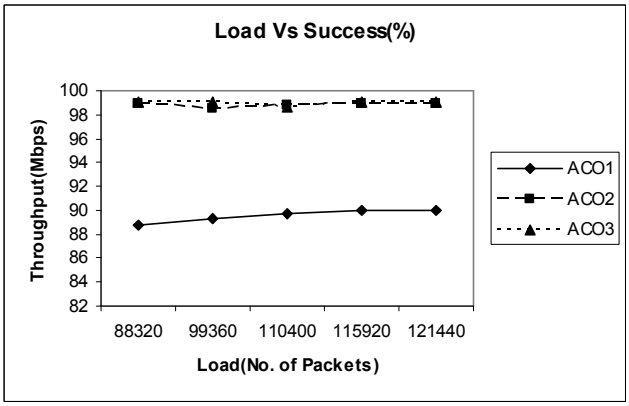

Figure 5.2: Variation of Percentage of Success Packets with load corresponding to table 5.

From Table 5, it can be seen that the percentage of successful packets decreases considerably in ACO1, but remains almost constant with increasing the load of the network. The throughput of ACO1 is however maximum for ACO1 in all the cases(Figure 5.1). For ACO2 and ACO3, the throughput is almost same and remains constant with load variations. But the throughput of ACO2 is always greater than the throughput of ACO3. However, the percentage of successful packets is slight more in ACO3 than in ACO2 when the load is low(Figure5.2).

c) Records by varying the size of the Tabu List in ACO3:

Tabu List: Tabu search[10] is a mathematical optimization method, belonging to the class of local search techniques. Tabu search enhances the performance of a local search method by using memory structures. The main objective of the Tabu list is to avoid cycles, thus making a global optimizer rather than a local optimizer. In ACO3, of the ACO[11] algorithms derived in this paper, a Tabu list has been maintained and the size of the Tabu list has been varied and the corresponding throughput and the percentage of successful packets are recorded for the network having large number of nodes.

Experiments has been carried out by varying the size of the Tabu list and the records are summarized below:

Table 6: Variation of throughput with the size of Tabu list

No. of nodes	Tabu list size	Success(%)	Throughput (Mbps)
14	3	99.89	70.906
	6	98.51	75.299
	9	97.22	84.016
	12	95.53	90.015
24	3	99.06	22.807
	6	94.89	24.116
	9	92.39	25.211
	12	91.24	26.189
	15	90.74	26.741
	18	90.32	26.979

In the table given above, all the records are taken by keeping the load of the network constant for a particular network structure. From the results of the experiments performed by varying the size of the Tabu List in ACO3, it can be seen that the percentage of successful packets and the throughput achieved in each of the cases are inversely related. As we are increasing the size of the Tabu list, the possibility of formation of the cycles during the path of the traversal is decreasing, and therefore the extra time consumed in removing the cycles from the path after the journey is complete decreases and so the throughput increases. But at the same time as more restrictions are imposed on the selection of the next node of the packets, the possibility of the cancellation of the journey of the packets also increases because when a packet is following such a path such that it has no other way to continue its journey except to repeat a node already visited by it which is present in the Tabu list maintained for that packet, then the only possible way to sort out the problem is to cancel the journey of the packet which increases the number of packets destroyed in the path of its journey. Comparing it with Table 1, it can be realized that as the size of the Tabu list is increased, the process of ACO3 tends to merge with the process of ACO1 and as the size of the Tabu list decreases, the process of the ACO3 tends to merge with the process of ACO2, where the packets are restricted not to visit the last visited node only.

CONCLUSION:

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. For example, from the reliability point of view, in all the cases, ACO2 and ACO3 are better than ACO1. This is because the number of packets

destroyed during the journey are less in these two cases. However, when the size of the network increases, the size of the Tabu list of ACO3 may be varied to obtain the optimum output.

Again, if we consider throughput as the only parameter, then ACO1 can be applied, as there is no possibility of making cycles in ACO1.This saves the wastage of time in removing the cycles from the path of the journey of the packets and hence increases the throughput of the system.

Future work on this algorithms can be carried out considering more factors that can be applied in a routing problems such a delay factor, congestion control etc.

REFERENCES:

[1] M. Dorigo, V. Maniezzo & A. Colorni, 1996. *"Ant System: Optimization by a Colony of Cooperating Agents"*, IEEE Transactions on Systems, Man, and Cybernetics–Part B, 26 (1): 29–41.

[2] Beckers R., Deneubourg J.L. and S. Goss (1992). Trails and U-turns in the selection of the shortest path by the ant Lasius niger. *Journal of theoretical biology*, 159, 397-415.

[3] *AntNet:ACO routing algorithm in practice*: Vincent Verstraete, Matthias Strobbe, Erik Van

[4] M.Dorigo & T.Stiitzle, Ant Colony Optimization, MIT Press, 2004.

[5]Mcgraw-Hill's Advanced Topics in Computer Science Series, New Ideas in Optimization.

[6] *Ant algorithms for discrete optimization.* Source, Artificial Life archive. Volume 5 , Issue 2 (April 1999).

[7] An adaptive multi-agent routing algorithm inspired by ants behavior. Gianni Di Caro and Marco Dorigo. IRIDIA – Université Libre de Bruxelles – Belgium.

[8] Ant Colonies for Adaptive Routing in Packet-Switched Communications Networks. Source, Lecture Notes In Computer Science

[9] The main characteristics of this model are positive feedback, ... 76**,** Positive feedback as a search strategy – Dorigo, Colorni – 1991

[10] Tabu Search**—**Part II. FRED GLOVER US WEST Chair in Systems Science, Center for Applied Artificial Intelligence.

[11] M. Dorigo**,** Optimization**,** Learning and Natural Algorithms**,** PhD thesis, Politecnico di Milano, Italie, 1992. S. Goss, S. Aron, J.-L. Deneubourg.

[12] Brian Hill, "The Complete Reference, CISCO", TATA MCGRAW-Hill publishing Ltd., N.Delhi.

[13] Evolutionary Algorithms 3 Selection-The simplest selection scheme-roulette-wheel selection.