International Journal of Computer Theory and Engineering, Vol. 1, No. 4, October2009
1793-8201

# Link State Generation Using Antnet Algorithm

D.N. Mallikarjuna Rao and Dr. V. Kamakshi Prasad

***Abstract*—Foraging behaviour of the ants is used for solving various problems in Computer Science. Foraging behaviour is one in which food is searched for by insects by exploring the environment in parallel. We take the example of ants' foraging behaour in our paper. The ants coordinate in indirect form by depositing a chemical substance called pheromone. The other ants follow the path with the help of the pheromone trails. We use [2]ACO (Ant Colony Optimization) for discovering the routes in a communication network. AntNet[1] is one of the approaches to adaptive learning of routing tables in wide area best-effort datagram networks. In this algorithm two types of ants are generated viz. Forward ant and Backward ants. While traveling from a source to a destination the forward ants store, in their memory, the paths and of the traffic conditions they encounter. After reaching the destination the forward ant transfers its memory to the backward ant and dies. The backward ant retraces the path traversed by the forward ant and updates the routing tables in the path. In this paper we combine the concepts of Antnet algorithm and Linkstate algorithm. In the [3]Link State algorithm, periodically each router discovers its neighbours and finds their network addresses, measures the cost to approach each neighbour, constructs a packet containing the information it has found from the neighbours and sends this packet to all other routers in the network. Each router which gets the information from every node about its neighbours computes the shortest path. AntNet is designed in such way that the forward ants carry the information about the status of the links it traverses. This status information can be captured and can be used to find the best path. We use the dijsktra algorithm to find the best path after capturing the information about all the links.**

***Index Terms*— forward ants, backward ants, Djkstra, Linkstate.**

## I. INTRODUCTION

A Computer network is a collection of autonomous computers and devices connected to each other through some medium. The medium can be copper wire, fibre optic cable or satellite link or radio frequency wave. With the help of the connectivity the computers can communicate with each other and share resources and information.

Routing[4] is the process of selecting paths in a network along which network traffic can be transmitted. Primarily routing protocols are classified into two types: dynamic and static. In static algorithms the routes are predetermined. The routing tables are loaded offline into the routers before making the router active. Dynamic routing algorithms are one in which the routers are adaptive to the changes in the network topology. Distance Vector Routing and Link State algorithm are some of the examples of the dynamic routing protocols.

AntNet is one of the dynamic routing algorithms for learning new routes. We describe the AntNet algorithm in the second section, the Link State algorithm in the third section, experimental setup in fourth section, results and comparison in 5th section.

## II. ANTNET ALGORITHM

Ant Colony Optimization[5] is a model in which multiple agents(ants) parallelly try to achieve a specific task by cooperating among themselves. A natural ant generates chemical substance and deposits on its way to the food source. Since an artificial ant(packet) cannot generate chemical substance, it is modeled to carry memory with it. While moving, the ant can change the pheromone trails (probabilistic value) associated with the problem.

AntNet is an ACO algorithm proposed by Gianni Di Caro and Marco Dorigo for data communication networks.

Brief description of the AntNet is given below: A forward ant is launched, at regular intervals, towards a destination 'd' from source 's' to find a feasible path from 's' to 'd'.

1. At each node, each forward ant which is going towards destination d, selects the next node to move on.
2. A cycle is detected when the forward reaches a node which is already present in its memory. If a cycle is detected, the nodes which are forming a cycle are removed from the memory.
3. When the forward ant reaches the destination node, the Ant agent generates a backward ant, transfers all of its memory to it and the forward ant is killed.
4. The backward ant retraces the same path the forward ant traversed but in the reverse direction.
5. The backward ant updates the routing table at every node in the path.

## III. GENERATION OF LINK STATES

We describe the Link State algorithm and then the changes we made in Link State algorithm and the Antnet algorithm.

Link State[3] Routing is being used from 1979. Now variants of link state routing are used. The idea behind the link state can be described like this, every node (we use the words router and node interchangeably) must do the following:

When a router comes up, it has to discover the neighboring nodes and learn their network addresses. This can be achieved by sending HELLO packets. All the neighbours are expected to send back a reply. Once it finds the neighbours, each router has to measure the cost (we used delay) to each of its neighbours. This can be accomplished by sending ECHO packets to each neighbour to which the neighbours have to reply immediately. Next step is to construct a packet containing the information it has just learnt. Then the packet

Final paper submitted on 26.06.2009 by D.N. Mallikarjuna Rao and Dr. V. Kamakshi Prasad

has to be broadcasted to the all the remaining nodes. Since every node gets these packets from every other node, each node can use these packets to find the shortest path.

Distributing[3] the Link State Packets is the most difficult aspect of the algorithm. Different types of distribution algorithms are available. Easiest of all the methods is the flooding algorithm, but this causes too many duplicate packets. Broadcasting another method which can be used to distribute the link state packets.

Once a full set of packets have been received by a router, it can construct the graph representing the entire network. Dijkstra's algorithm can be run locally at every router(node) to construct the shortest path to all possible destinations.

How the Antnet algorithm is used for Link State generation:

In the Antnet forward ants are generated periodically at every node and, as described in the Antnet algorithm above, they reach the destination by traversing various nodes in the network. The forward ant carries the path (sequence of nodes and the corresponding time delay from the source) it has traversed. In the basic algorithm when the forward ant reaches the destination a backward ant is created. Then the entire memory carried by the forward is transferred onto the backward ant and then the forward ant is killed. In this algorithm we do not create any backward ant, instead we use the information carried by the forward ant to construct a graph representing the entire network. We also use, like in Link State routing, the Dijkstra algorithm to compute the best path after which we update the routing table using the result of the Dijkstra algorithm.

The proposed changes have been implemented in ns-2.33[6] network simulator based on the Antnet implementation by V. Laxmi, Lavina Jain[5]. We have used arbitrary topology as provided in the documentation of [3] for better comparison. We have used the same link capacities.

## IV. Experimental Setup and Results

The proposed changes have been implemented in ns-2.33[6] network simulator based on the Antnet implementation by V. Laxmi, Lavina Jain[5]. We have used arbitrary topology as provided in the documentation of [3] for better comparison. We have used the same link capacities.

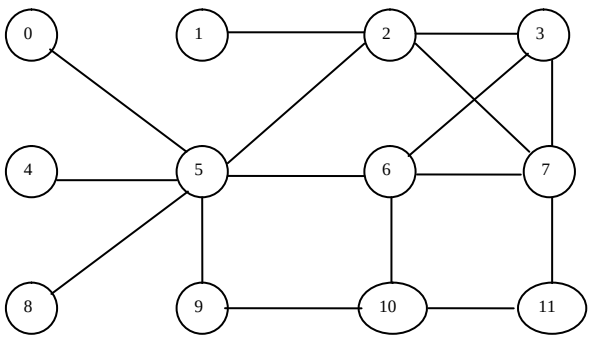


Fig .1 Network Topology used for the experiment

In this network, the propagation delay in each line is 155 ms. We have experimented by generating forward ants periodically every 0.03 time units. These ants travel as per the algorithm described above. When the ant reaches the destination, its memory contains all the nodes (routers) visited and the time taken to reach every node. That means every forward ant contains the link state between every two nodes, it has visited. We use these link states to construct a graph of the links of the network.

At the first instance, we used only the ants which reached destination to construct the graph. Then we used all the ants which are passing by this node (the node where we are trying to construct a graph) to construct the graph. The experiments have shown excellent results. Less than half the number of ants(when compared to the basic AntNet) were sufficient to construct the graph using which the routing table is constructed.

The table shown is the routing table at node 3 after applying the basic Antnet algorithm. This is given for comparison with our algorithm.

Table 1

***Routing Table at Node 3***

| Dest node | Next node | Ph Value | Dest node | Next node | Pheromone value |
|---|---|---|---|---|---|
| 0 | 7 | 0.273717 | 6 | 2 | 0.047743 |
| 0 | 6 | 0.422875 | 7 | 7 | 0.722928 |
| 0 | 2 | 0.303408 | 7 | 6 | 0.102545 |
| 1 | 7 | 0.281825 | 7 | 2 | 0.174526 |
| 1 | 6 | 0.172239 | 8 | 7 | 0.255596 |
| 1 | 2 | 0.545936 | 8 | 6 | 0.401245 |
| 2 | 7 | 0.165806 | 8 | 2 | 0.343159 |
| 2 | 6 | 0.046219 | 9 | 7 | 0.301843 |
| 2 | 2 | 0.787975 | 9 | 6 | 0.529727 |
| 4 | 7 | 0.277198 | 9 | 2 | 0.168430 |
| 4 | 6 | 0.372067 | 10 | 7 | 0.397410 |
| 4 | 2 | 0.350735 | 10 | 6 | 0.439464 |
| 5 | 7 | 0.197638 | 10 | 2 | 0.163126 |
| 5 | 6 | 0.482314 | 11 | 7 | 0.582402 |
| 5 | 2 | 0.320048 | 11 | 6 | 0.193428 |
| 6 | 7 | 0.158613 | 11 | 2 | 0.224170 |
| 6 | 6 | 0.793644 | | | |

The table given above (Table 1) is the routing table generated by AntNet algorithm at node 3. It contains three columns namely Destination Node, Next Node and Pheromone value.

We can observe that for each destination there are multiple next nodes with different pheromone (probability) values.

Table 2a

| ***node 0*** | | ***node 1*** | | ***node 2*** | | ***node 3*** | | ***node 4*** | | ***node 5*** | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 2 | 0 | 5 | 0 | 2 | 0 | 5 | 0 | 0 |
| 1 | 5 | 1 | 1 | 1 | 1 | 1 | 2 | 1 | 5 | 1 | 2 |
| 2 | 5 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 5 | 2 | 2 |
| 3 | 5 | 3 | 2 | 3 | 3 | 3 | 3 | 3 | 5 | 3 | 2 |
| 4 | 5 | 4 | 2 | 4 | 5 | 4 | 2 | 4 | 4 | 4 | 4 |
| 5 | 5 | 5 | 2 | 5 | 5 | 5 | 2 | 5 | 5 | 5 | 5 |
| 6 | 5 | 6 | 2 | 6 | 3 | 6 | 6 | 6 | 5 | 6 | 6 |
| 7 | 5 | 7 | 2 | 7 | 7 | 7 | 7 | 7 | 5 | 7 | 2 |
| 8 | 5 | 8 | 2 | 8 | 5 | 8 | 2 | 8 | 5 | 8 | 8 |
| 9 | 5 | 9 | 2 | 9 | 5 | 9 | 6 | 9 | 5 | 9 | 9 |
| 10 | 5 | 10 | 2 | 10 | 7 | 10 | 6 | 10 | 5 | 10 | 6 |
| 11 | 5 | 11 | 2 | 11 | 7 | 11 | 7 | 11 | 5 | 11 | 2 |

Table 2b.

| ***node 6*** | | ***node 7*** | | ***node 8*** | | ***node 9*** | | ***node 10*** | | ***node 11*** | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 2 | 0 | 5 | 0 | 5 | 0 | 9 | 0 | 7 |
| 1 | 7 | 1 | 2 | 1 | 5 | 1 | 5 | 1 | 6 | 1 | 7 |
| 2 | 7 | 2 | 2 | 2 | 5 | 2 | 5 | 2 | 6 | 2 | 7 |
| 3 | 3 | 3 | 3 | 3 | 5 | 3 | 6 | 3 | 6 | 3 | 7 |
| 4 | 5 | 4 | 2 | 4 | 5 | 4 | 5 | 4 | 9 | 4 | 7 |
| 5 | 5 | 5 | 2 | 5 | 5 | 5 | 5 | 5 | 9 | 5 | 7 |
| 6 | 6 | 6 | 6 | 6 | 5 | 6 | 6 | 6 | 6 | 6 | 10 |
| 7 | 7 | 7 | 7 | 7 | 5 | 7 | 6 | 7 | 11 | 7 | 7 |
| 8 | 5 | 8 | 2 | 8 | 8 | 8 | 5 | 8 | 9 | 8 | 7 |
| 9 | 9 | 9 | 6 | 9 | 5 | 9 | 9 | 9 | 9 | 9 | 10 |
| 10 | 10 | 10 | 11 | 10 | 5 | 10 | 10 | 10 | 10 | 10 | 10 |
| 11 | 10 | 11 | 11 | 11 | 5 | 11 | 10 | 11 | 11 | 11 | 10 |

When we generate link states using the forward ants we get one best route for each destination. The tables above (Table 2a and 2b) show one next node for each destination as generated by applying Dijkstra algorithm to the link states generated by capturing the forward ant packets at the destination and in the intermediate nodes. The results are conforming to the topology and the delays of the links Using the same figure (i.e. fig.1) we now change the link capacities to the following.

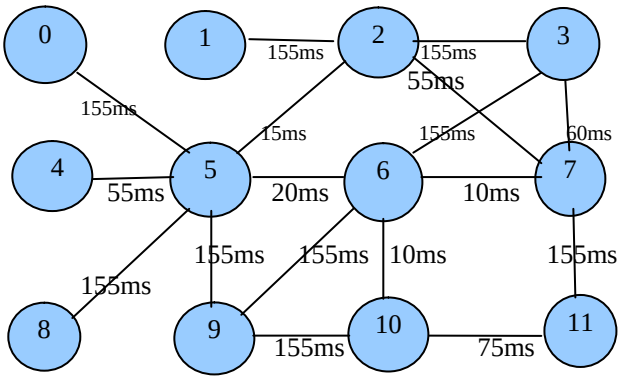


Fig.2 Same Network Topology but with different link capacities**.**

The following are the routing tables after applying the our proposed algorithm on the above network.

Table 3a

| ***node 0*** | | ***node 1*** | | ***node 2*** | | ***node 3*** | | ***node 4*** | | ***node 5*** | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 2 | 0 | 5 | 0 | 7 | 0 | 5 | 0 | 0 |
| 1 | 5 | 1 | 1 | 1 | 1 | 1 | 7 | 1 | 5 | 1 | 2 |
| 2 | 5 | 2 | 2 | 2 | 2 | 2 | 7 | 2 | 5 | 2 | 2 |
| 3 | 5 | 3 | 2 | 3 | 5 | 3 | 3 | 3 | 5 | 3 | 6 |
| 4 | 5 | 4 | 2 | 4 | 5 | 4 | 7 | 4 | 4 | 4 | 4 |
| 5 | 5 | 5 | 2 | 5 | 5 | 5 | 7 | 5 | 5 | 5 | 5 |
| 6 | 5 | 6 | 2 | 6 | 5 | 6 | 7 | 6 | 5 | 6 | 6 |
| 7 | 5 | 7 | 2 | 7 | 5 | 7 | 7 | 7 | 5 | 7 | 6 |
| 8 | 5 | 8 | 2 | 8 | 5 | 8 | 7 | 8 | 5 | 8 | 8 |
| 9 | 5 | 9 | 2 | 9 | 5 | 9 | 7 | 9 | 5 | 9 | 9 |
| 10 | 5 | 10 | 2 | 10 | 5 | 10 | 7 | 10 | 5 | 10 | 6 |
| 11 | 5 | 11 | 2 | 11 | 5 | 11 | 7 | 11 | 5 | 11 | 6 |

Table 3b

| ***node 6*** | | ***node 7*** | | ***node 8*** | | ***node 9*** | | ***node 10*** | | ***node 11*** | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 5 | 0 | 6 | 0 | 5 | 0 | 5 | 0 | 6 | 0 | 10 |
| 1 | 5 | 1 | 6 | 1 | 5 | 1 | 5 | 1 | 6 | 1 | 10 |
| 2 | 5 | 2 | 6 | 2 | 5 | 2 | 5 | 2 | 6 | 2 | 10 |
| 3 | 7 | 3 | 3 | 3 | 5 | 3 | 6 | 3 | 6 | 3 | 10 |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 5 | | 4 | 6 | | 4 | 5 | | 4 | 5 | | 4 | 6 | | 4 | 10 |
| 5 | 5 | | 5 | 6 | | 5 | 5 | | 5 | 5 | | 5 | 6 | | 5 | 10 |
| 6 | 6 | | 6 | 6 | | 6 | 5 | | 6 | 6 | | 6 | 6 | | 6 | 10 |
| 7 | 7 | | 7 | 7 | | 7 | 5 | | 7 | 6 | | 7 | 6 | | 7 | 10 |
| 8 | 5 | | 8 | 6 | | 8 | 8 | | 8 | 5 | | 8 | 6 | | 8 | 10 |
| 9 | 9 | | 9 | 6 | | 9 | 5 | | 9 | 9 | | 9 | 6 | | 9 | 10 |
| 10 | 10 | | 10 | 6 | | 10 | 5 | | 10 | 10 | | 10 | 10 | | 10 | 10 |
| 11 | 10 | | 11 | 6 | | 11 | 5 | | 11 | 10 | | 11 | 11 | | 11 | 11 |

The results in the routing table are conforming to the changes in the line capacities.

## V. Conclusion and Future Work

We have established that we can use AntNet algorithm to build link states with the help of the forward ants. After generating the graph of the link states we can compute the best route with the help of the Dijkstra algorithm. In this an ant carries only delay. We can include Bandwidth also in the memory and compute best routes in terms of delay, bandwidth and number of hops.

## References

[1] Luca maria Gambardella, IDSIA, Lugano "Ant colony Optimization for ad-hoc networks", The first MICS Workshop on Routing for Mobile Ad-hoc Networks, Zurich, February 13, 2003.
[2] Gianni Di Caro and Marco Dorigo, Antnet: Distributed Stigmergic Control for Communication Networks. Journal of Artificial Intelligence Research, 9, 1998.
[3] Andrew S. Tannenbaum , Computer Networks , Fourth Edition, Pearson Education.
[4] http://en.wikipedia.org/wiki/PSTN
[5] V.Laxmi, Lavina Jain, M.S. Gaur, "International Conference on Wireless Communication and Sensor Networks, WSCN 2006".
[6] The VINT Project, The ns Manual, M. Young, *The Techincal Writers Handbook.* Mill Valley, CA: University Science, 1989.

**D.N. Mallikarjuna Rao is** working as Professor in Jyothishmathi Institute of Technology and Science , Karimnagar, Andhra Pradesh.
**Dr. V. Kamakshi Prasad** is a Professor with CSE department in Jawaharlal Nehru Technological Univeristy, Hyderabad, Andhra Pradesh, India.