Antnet: Modified Routing Algorithm for Packet Switched Networks.

Shuchita Upadhyaya[1] and Richa Setiya[2]
[1]*Department of Computer Sc. & Applications, Kurushetra University Kurukshetra.*
[2]*Department of Computer Sc & Engg, Israna, Panipat.*

Abstract: **Antnet is an agent based routing algorithm based on real ants' behavior. Ants are able to find shortest path to food source .In real life, ants drop some kind of chemical substances to mark the path that they used. Then on their way back they choose the path with the highest pheromones which becomes the shortest path. But Antnet Algorithms may cause the network congestion and stagnation. Here, some more special type of ants is proposed with little overhead to optimize the antnet routing.**

Keywords: Routing, antnet agent, swarm intelligence, adaptive.

1. INTRODUCTION

Today with the fast growth of Internet everybody wants to get connected to the internet. Millions of people use the Internet for daily business all over the world. Today the Internet has become very large, complex and dynamic. Failures and challenges occur at every step because of traffic flow from one part of network to another part. Routing is the process of selecting paths in a network to send traffic. Routing is an important aspect of network communication which affects the performance of any network, since other characteristics of the network like throughput, reliability and congestion depend directly on it. In packet switching networks when a packet travels from a source to destination, it has to pass through a number of networks with varying characteristics. So an ideal routing algorithm is one which is able to deliver the packet to its destination with minimum amount of delay. It must be adaptive and intelligent enough to make the decisions. The growing size and increasing demands of Internet impelled the study of more powerful routing algorithms which can optimize the flow of traffic.

The routing algorithms currently in use (e.g. OSPF, RIP, and BGP) are not sufficient to tackle the increasing complexity of such networks. They are not adaptive, intelligent and fault intolerant. The routing tables in them are updated by exchanging routing information between the routers. Different routing protocols use different approaches to exchange the routing information. There are mainly two approaches for routing algorithms, distance-vector algorithms and link-state algorithms. Distance vector algorithms use the Bellman algorithm. This approach assigns a number, the cost, to each of the links between each node in the network. Nodes will send information from one point to another point via the path that result in the lowest total cost. In link-state algorithms for example, Open Shortest Path First (OSPF), the routers exchange link-state information by flooding the link state packets. The link state updates are generated only when the link status changes. Once a node has obtained topology information of the entire network, Dijkstra's algorithm is generally used to compute the shortest path.The two main performance metrics that are affected by the routing algorithm are throughput and average packet delay. Coordination is needed between nodes. Nodes and links can fail, and congestion can arise in some areas. Thus, the routing algorithm needs to modify its routes, redirecting traffic and updating databases very quickly and adaptively.

In recent years, a new kind of routing protocols influenced by software agents called Ant Based routing is developed. S. Appleby and S. Steward were the first ones to introduce the concept of software agents used for control in telecommunication networks [1].This approach of software agents was modified for routing problem by R. Schoonderwoerd[2].The research process continued and it was applied to connection oriented networks [3]. Ant based routing was then applied to packet based connection less systems [4].This agent based approach was further researched and was modified for adaptive routing [5].Swarm intelligence provides a promising alternative to traditional routing algorithms by utilizing mobile software agents for network management.

Although, an ant [1], [2] is a simple and unsophisticated creature, collectively a colony of ants can perform useful tasks such as building nests, and foraging (searching for

food)[1],[2],[3].What is interesting is that ants are able to discover the shortest path to a food source and to share that information with other ants through stigmergy [1]-[5]. Stigmergy is a form of indirect communication used by ants in nature to coordinate their problem-solving activities. Ants achieve stigmergic communication by laying a chemical substance called pheromone.

Ant Colony Optimization (ACO) [5]-[9] is a family of optimization algorithms based on real ants' behavior. ACO is inspired by the foraging behavior of ant colonies, where in they are able to find shortest path to food source. It has been observed that of available routes, ants find shortest route to food source. In real life, ants deposit some kind of chemical substances to mark the path that they used. Then on their way back they choose the path with the highest pheromones which becomes the shortest path. AntNet is an Ant Colony Optimization (ACO) meta heuristic for data network routing proposed by Gianni Di Caro and Marco Dorigo [6]-[9]. In this network routing algorithm, a group of mobile agents (or artificial ants) build paths between pair of nodes, exploring the network concurrently and exchanging obtained information to update the routing tables. This information is also used to direct the data packets towards their destination.

2. ANTNET ROUTING ALGORITHM

In antnet [6]-[9] software agents explore the network to find the optimal paths from the randomly selected source destination pairs. Moreover while exploring the network ants update the probabilistic routing tables and build statistical models of the nodes local traffic. Ants use these tables to communicate with each other.

2.1 Data Structure maintained at each node

Routing table T_k is a local data-base that helps router to decide where to forward data packets. It contains the information which specifies the next (neighbor) node that should be taken by a data packet to get to any possible destination in the network. Each routing table is organized as a set of

- All the possible destinations (all the nodes in the network).
- The probabilities to reach these destinations through each of the neighbors of the node.

T_k stores a probability value P_{nd} which express the goodness of choosing n as next node when the destination node is d

$$\Sigma P_{nd} = 1,\ d \in [1, N], N_k = neighbors(k)$$
$$n \in N_k$$

Probability value P_{nd} represents pheromone concentration along the link from node k to neighbor node n for destination node d.

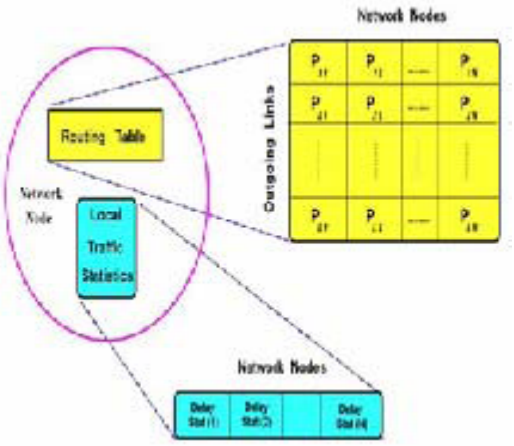

Figure 1

Local traffic statistics is a second data-structure that each node has. The main task of this structure is to follow the traffic fluctuations as seen by local node k. For each destination d in the network, an array $M_k(\mu_d, \sigma_d^2, W_d)$ contains an estimated mean $\mu_{d,}$ an estimated variance σ_d^2 computed over the times experienced by the artificial ants, and a moving observation window $W_{d..}$ The moving observation window W_d , of size W_{max}, represents an array containing the trip times of last W_{max} forward ants that travel from the node k to the destination d.

2.2 The AntNet algorithm

The operation of AntNet as proposed by Di Caro and Dorigo is based on two types of
agents:

• Forward Ants who gather information about the state of the network, and

• Backward Ants who use the collected information to update the routing tables of routers on their path

The AntNet algorithm is described as follows

1. At regular time intervals Δt from every node s, forward ant $F_{s\to d}$ is launched toward a destination node d to discover a possible, low-cost path to that node and to explore the load status of the network. The forward ants make use of the same priority queues as used by data packets.

Destinations are locally selected according to the data traffic patterns generated by the local workload: if f_{sd} is a measure (in bits or in number of packets) of the data flow $s \rightarrow d$, then the probability of creating at node s a forward ant with node d as destination is

$$p_{sd} = \frac{f_{sd}}{\sum_{i=1}^{N} f_{si}}$$

2.The forward agents create a stack where they store memory of their paths and of the traffic conditions found. The identifier of every visited node i and the time elapsed since the launching time to arrive at this i-th node is pushed onto the memory stack.
3. At each node i, each forward ant moving towards its destination d chooses the next node j to move to as follows:

- If all the neighboring nodes have not been visited, then the next neighbor is chosen among the nodes that have not been visited as:

$$P_{ijd} = \frac{\tau_{ijd} + \alpha\eta_{ij}}{1+\alpha(|N_i| - 1)}$$

Here, N_i represents the set of neighbors of the current node i and $|N_i|$ the cardinality of that set, i.e., the number of neighbors, while the heuristic correction η_{ij} is a normalized value [0,1] such that $1 - \eta_{ij}$ is proportional to the length q_{ij} (in bits waiting to be sent) of the queue of the link connecting the node i with its neighbor j:

$$\eta_{ij} = 1 - \frac{q_{ij}}{\sum_{i=1}^{Ni} qi1}$$

The value of α weighs the importance of the instantaneous state of the node's queue with respect to the probability values stored in the routing table. η_{ij} reflects the instantaneous state of the node's queues, and assuming that the queue's consuming process is almost stationary or slowly varying, η_{ij} gives a quantitative measure associated with the queue waiting time.

- If all the neighboring nodes have been visited previously, then the next node is chosen uniformly among all the neighbors. In this case, since all the neighbors have been visited previously the forward ant is forced to return to a previously visited node. Thus, irrespective of which neighbor is chosen as the next node, the forward ant is in a loop (cycle).

4. If a cycle is detected, that is, if an ant is forced to return to an already visited node, the cycle's nodes are popped from the ant's stack and all the memory about them is destroyed. If the cycle lasted longer than the lifetime of the ant before entering the cycle, (that is, if the cycle is greater than half the ant's age) the ant is destroyed.

5. When the destination node d is reached, the agent $F_{s \rightarrow d}$ generates backward agent $B_{d \rightarrow s}$, transfers to it all of its memory, and dies.

6. The backward ant takes the same path as that of its corresponding forward ant, but in the opposite direction. At each node i along the path it pops its stack to know the next hop node. Backward ants don't share the same link queues as data packets; they use higher priority queues, because their task is to quickly propagate to the routing tables the information accumulated by the forward ants.

7. Arriving at a node i coming from a neighbor node f, the backward ant updates the two main data structures of the node, the local model of the traffic M_i and the routing table T_i, for all the entries corresponding to the (forward ant) destination node d.

2.3 Update traffic model M_i

M_i is updated with the values stored in the stack memory. The time elapsed to arrive (for the forward ant) to the destination node d starting from the current node i is used to update the mean μ_d, variance estimates σ^2_d and the best value over the observation window W_d. The moving observation window W_d, of size W_{max} represents an array containing the trip times of last W_{max} forward ants that travel from the node i to the destination d. The moving observation window W_d is used to compute the best trip time t_{bestd} i.e., the best trip time experienced by a forward ant travelling from the node i to the destination d among the last W_{max} forward ants that travel from the node i to the destination d.

The mean value of this time and its dispersion can vary strongly, depending on the traffic conditions: a poor time (path) under low traffic load can be a very good one under heavy traffic load. The statistical models had to be able to capture this variability and to follow in a robust way the fluctuations of the traffic.

The estimated mean and variance are updated as follows:

$$\mu_{id} \leftarrow \mu_{id} + \square\ (o_{i\to d} - \mu_{id})$$
$$\sigma^2_{id} \leftarrow \sigma^2_{id} + \square\ ((o_{i\to d} - \mu_{id})^2 - \sigma^2_{id})$$

Where $o_{i\to d}$ represents newly observed ant's trip time from node i to destination d. The factor $\square$ weighs the number of most recent samples that will really affect the mean μ_d and the variance σ_d^2 .

$$W_{max} = \frac{5c}{\square} \qquad \text{where } c < 1$$

The best value t_{bestd} of the forward ants trip time from node i to the destination d stored in the moving observation window W_d is also updated by the backward ant. If the newly observed forward ant's trip time $o_{i\to d}$ from the node i to the destination d is less then t_{bestd} then t_{bestd} is replaced by $o_{i\to d}$.

2.4 Update pheromone matrix T_i

The reinforcement $r = r(T,M_i)$ is defined to be a function of the goodness of the observed trip time as estimated on the basis of the local traffic model. r is a dimensionless value, $r \in (0,1]$, used by the current node i as a positive reinforcement for the node f the backward ant $B_{d\to s}$ comes from. Reinforcement value r takes into account some average of the so far observed values and of their dispersion to score the goodness of the trip time T, such that the smaller T is, the higher r is.

The backward ant $B_{d\to s}$ moving from node f to node i increases the pheromone values $\tau_{ifd'}$

$$\tau_{ifd'} \leftarrow \tau_{ifd'} + r.(1 - \tau_{ifd'})$$

Pheromones $\tau_{ifd'}$ for destination d' of the other neighboring nodes $j, j \in Ni, j \neq f,$ evaporate implicitly by normalization. That is, their values are reduced so that the sum of probabilities will still be 1.

$$\tau_{ijd'} \leftarrow \tau_{ijd'} - r.(\tau_{ijd'}), j \in N_i, j \neq f$$

The factor of reinforcement r is calculated considering three fundamental aspects: (i) the paths should receive an increment in their probability of selection, proportional to their goodness, (ii) the goodness is a traffic condition dependent measure that can be estimated by Mi and (iii) they should not continue all the traffic fluctuations in order to avoid uncontrolled oscillations. It is very important to establish a commitment between stability and adaptability. Between several tested alternatives was chosen to calculate r:

$$r = c_1\left(\frac{W_{best}}{T}\right) + c_2\left(\frac{I_{sup} - I_{inf}}{(I_{sup} - I_{inf}) + T(I_{sup} - I_{inf})}\right)$$

where: W_{best}. best trip of an ant to node d', in the last observation window $W_{d'}$,
$I_{inf} = W_{best}$ lower limit of the confidence interval for μ,
$I_{sup} = \mu + z * \sigma(/\sqrt{|W|})$ upper limit of the confidence interval for μ, with:
$z = 1/(\sqrt{1-\square})$, $\square$ = confidence level, $\square \in [0.75, 0.8]$.
c_1 and c_2 are weight constants, chosen experimentally as $c_1 = 0.7$ y $c_2 = 0.3$.

2.5 Limitations

Although Experiments of AntNet have shown very promising results, antnet has outperformed under different experimental conditions with respect to other dynamic routing algorithms e.g. RIP, OSPF. Still there are some problems with this adaptive algorithm. One of the major problems is that the network gets trapped because a node prefers a link with higher probability to a destination when choosing an outgoing link say n_o to send a packet. If link n_o keeps good condition for a long time, its probability to that destination will be very high. Such a condition may cause congestion of n_o; it also reduces probability of selecting other paths. Hence the node will stuck to this outgoing link and loose its adaptive ability. This is called the problem of "stagnation". Stagnation is reached when a node reaches its convergence. Stagnation is a very critical problem for any network because

1) n_o may lose its optimality if it gets congested;
2) If the network gets fails at any time then the most preferred path n_o may become unavailable
3) Other non-optimal paths may become optimal due to changes in network topology

Many researchers [11]-[13] have tried to provide the solution for the same. These methods are

- Evaporation
- Aging
- Limiting and Smoothing Pheromone
- Privileged Pheromone Laying

- Pheromone-Heuristic Control

But all the methods are very complex and needed extra overhead.

3. ANTNET MODIFICATIONS

***Proposed Scheme**:* Here, other than forward and backward ants clone ants are introduced. The clone ants are generated to avoid the frequent generation of forward ants from source node to explore the path. This will be able to reduce the total number of generation of forward ants by introducing a little overhead. The number of clones generated depends upon number of multiple paths identified. Multiple paths are identified according to the probabilities in the pheromone table. Clone ants will explore these paths and will provide one optimal path amongst all these paths.

3.1 Proposed Antnet Scheme

1. At regular time intervals Δt from every node s, forward ant $F_{s\to d}$ is launched toward a destination node d to discover a possible, low-cost path to that node and to explore the load status of the network. The forward ants make use of the same priority queues as used by data packets.

Destinations are locally selected according to the data traffic patterns generated by the local workload: if f_{sd} is a measure (in bits or in number of packets) of the data flow $s\to d$, then the probability of creating at node s a forward ant with node d as destination is

$$p_{sd} = \frac{f_{sd}}{\sum_{i=1}^{N} f_{si}}$$

2. The forward agents create a stack where they store memory of their paths and of the traffic conditions found. The identifier of every visited node i and the time elapsed since the launching time to arrive at this i-th node is pushed onto the memory stack.

3. At each node i, the forward ant moving towards its destination d calculates the probability of neighbor node j as follows:

- If all the neighboring nodes have not been visited, then the probability of neighbor node is calculated among the nodes that have not been visited as:

$$P_{ijd} = \frac{\tau_{ijd} + \alpha\eta_{ij}}{1+\alpha(|N_i| - 1)}$$

Here, N_i represents the set of neighbors of the current node i and $|N_i|$ the cardinality of that set, i.e., the number of neighbors, while the heuristic correction η_{ij} is a normalized value [0,1] such that $1-\eta_{ij}$ is proportional to the length q_{ij} (in bits waiting to be sent) of the queue of the link connecting the node i with its neighbor j:

$$\eta_{ij} = 1 - \frac{q_{ij}}{\sum_{i=1}^{Ni} qi1}$$

The value of α weighs the importance of the instantaneous state of the node's queue with respect to the probability values stored in the routing table. η_{ij} reflects the instantaneous state of the node's queues, and assuming that the queue's consuming process is almost stationary or slowly varying, η_{ij} gives a quantitative measure associated with the queue waiting time.

A table consisting of every unvisited neighbor node $j \in N_k$ and the associated probabilities is maintained.

- If all the neighboring nodes have been visited previously, then the next node is chosen uniformly among all the neighbors. In this case, since all the neighbors have been visited previously the forward ant is forced to return to a previously visited node. Thus, irrespective of which neighbor is chosen as the next node, the forward ant is in a loop (cycle).

4. If a cycle is detected, that is, if an ant is forced to return to an already visited node, the cycle's nodes are popped from the ant's stack and all the memory about them is destroyed. If the cycle lasted longer than the lifetime of the ant before entering the cycle, (that is, if the cycle is greater than half the ant's age) the ant is destroyed.

Introducing Cloning Ants

5. The forward agent produces cloning ants. The number of clones generated depends upon number of multiple paths identified. Multiple paths are identified as below according to the table maintained in step 3 consisting of probabilities. So instead of one neighbor node more than one neighbor nodes are selected.

Identifying multiple optimal Paths

Multiple paths are identified by using the probabilities which are stored in a table calculated in step 3.

- Probability values in the table are picked up and a sorting algorithm is executed on these values.
- The table is reshuffled according to the sorting algorithm executed and sorted values ranging from higher to lower are stored in table.

- The difference p, amongst the adjacent values is calculated and is compared to some threshold value say p_m.
- If the difference p is less than p_m then those values are selected and comparison amongst the adjacent values is continued until difference is greater than p_m.
- Otherwise at the very first occurrence of difference greater than p_m, the comparison is stopped and the corresponding value(s) in the table is (are) selected.

6. Every clone ant will parallel travel through the above selected links.
7. At every node it will be checked to see whether it has been visited previously by the clone. This can be checked by fixing a flag in every node. A flag bit is set accordingly. The status of flag is checked, if it is "set" then that means the node has been visited previously and the clone is destroyed itself. If flag bit is not "set" then the node is added to the stack of the visiting clone ant. After this, node id is compared with destination id, if this is the destination node then the flag bit is converted as "set" , clone ant generates backward ant $B_{d\rightarrow s}$ transfers to it all of its memory, and dies. If this is not the destination node then again go to step5, i.e. again identify the multiple paths and accordingly generate clones.
8. The backward ant takes the same path as that of its corresponding forward ant, but in the opposite direction. At each node i along the path it pops its stack $S_{s\rightarrow d}$ to move to the next node. Backward ants do not share the same queues as data packets and forward ants; they use high priority queues to quickly propagate to the routing tables the information collected by the forward ants.
9. Arriving at a node i coming from a neighbor node f, the backward ant updates the two main data structures of the node, the local model of the traffic M_i and the routing table T_i, for all the entries corresponding to the (forward ant) destination node d.

3.2 Update traffic model M_i

The estimated mean and variance are updated as follows:

$\mu_{id} \leftarrow \mu_{id} + \square\ (o_{i\rightarrow d} - \mu_{id})$

$\sigma^2_{id} \leftarrow \sigma^2_{id} + \square\ ((o_{i\rightarrow d} - \mu_{id})^2 - \sigma^2_{id})$

Where $o_{i\rightarrow d}$ is the observed ant's trip time from node i to destination d. The factor $\square$ weighs the number of most recent samples that will really affect the average.

3.3 Update Pheromone matrix T_i

$\tau_{ifd'} \leftarrow \tau_{ifd'} + r.(1 - \tau_{ifd'})$

Pheromones $\tau_{ijd'}$ for destination d' of the other neighboring nodes j, $j \in Ni$, $j \neq f$, evaporate implicitly by normalization. That is, their values are reduced so that the sum of probabilities will still be 1.

$\tau_{ijd'} \leftarrow \tau_{ijd'} - r.(\tau_{ijd'}),\ j \in N_i,\ j \neq f$

Where r is the reinforcement factor as in original antnet algorithm.

4. CONCLUSION AND FUTURE WORK

In this paper an improved version of the AntNet algorithm is proposed. In the improved version, more than one optimal outgoing interfaces are identified and clone ants are introduced which will travel through these paths. This algorithm is supposed to provide higher throughput and will be able to explore new and better paths even if the network topologies gets changed very frequently. In the future work, we intend to simulate the same using simulator so that exact results can be found.

REFERENCES

[1] S. Appleby and S. Steward, "Mobile software agents for control in telecommunication networks," *BT echnol.* vol. 12, no. 2, 1994.
[2] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz,"Ants for Load Balancing in Telecommunication Networks," Hewlett Packard Lab., Bristol, U.K., Tech. Rep. HPL-96-35, 1996.
[3] E. Bonabeau ,D.Suyers,"Routing in telecommunication networks with smart like agents ",Proceedings of LATA,1998.
[4] D.Subbramanian,P.Druschel,and J.Chen,'Ants and reinforcement learning: A case study in routing in dynamic networks",Proces of IJCAI-97,Palo Alto,Morgan Kauffman,1997,pp 832-838
[5] M. Dorigo and G.D.Caro,"Antnet : A mobile agents approach to adaptive routing ",Tech Report, University Libre de Bruxelles,IRIDIA,1997.
[6] G. D. Caro and M. Dorigo, "AntNet: Distributed stigmergetic control for communications networks," Journal of Artificial Intelligence Research, 9, 1998 vol. 9, pp. 317–365,1998.
[7] G. D. Caro and M. Dorigo, "Ant colonies for adaptive routing in packet-switched communications networks," in Proc. 5th Int. Conf. Parallel Problem Solving from Nature, Amsterdam, The Netherlands, Sept. 27–30, 1998.
[8] G. D. Caro and M. Dorigo, "Two ant colony algorithms for best-effort routing in datagram networks," in Proc. 10th IASTED Int. Conf. Parallel Distributed Computing Systems, 1998, pp. 541–546.
[9] M. Dorigo, G. D. Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," Artif. Life, vol. 5, no. 2, pp. 137–172, 1999.
[10] B. Baran and R. Sosa, "A new approach for AntNet routing," presented at the Proc. 9th Int. Conf. Computer Communications Networks, Las Vegas, NV, 2000.
[11] T. Stuzle and H. H. Hoos, "MAX-MIN ant system," Future Gener. Comput. Syst. J., vol. 16, no. 8, pp. 889–914, 2000.
[12] Kwang Mong Sim and Weng Hong Sun, "Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions ",IEEE transactions on systems and humans,Vol.33,No.5 Sept 2003.
[13]Marco Dorigo and Krzysztof Socha," An Introduction to Ant Colony Optimization"IRIDIA- Technical Report Series Technical Report No. TR/IRIDIA/2006-010 April 2006, Last revision: April 2007.
[14]Marco Dorigo, Mauro Birattari, and Thomas Stutzle,"Ant Colony Optimization" IRIDIA – Technical Report Series, Technical Report No. TR/IRIDIA/2006-023 , September 2006.