Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

Adaptive swarm-based routing in communication networks

LÜ Yong (吕 勇)†, ZHAO Guang-zhou (赵光宙), SU Fan-jun (苏凡军), LI Xiao-run (历小润)

(*College of Electrical Engineering, Zhejiang University, Hongzhou 310027, China*)

†E-mail: lvyongs@sohu.com

Received July 30, 2003; revision accepted Sept. 11, 2003

Abstract: Swarm intelligence inspired by the social behavior of ants boasts a number of attractive features, including adaptation, robustness and distributed, decentralized nature, which are well suited for routing in modern communication networks. This paper describes an adaptive swarm-based routing algorithm that increases convergence speed, reduces routing instabilities and oscillations by using a novel variation of reinforcement learning and a technique called momentum. Experiment on the dynamic network showed that adaptive swarm-based routing learns the optimum routing in terms of convergence speed and average packet latency.

Key words: Communication networks, Ant based, Adaptive routing
Document code: A **CLC number:** TN915.11

INTRODUCTION

Communication network management is becoming increasingly difficult due to the increasing size, rapidly changing topology, and complexity of communication networks. Current routing algorithms cannot adequately tackle such networks. Centralized algorithms have scalability problems; static algorithms have trouble keeping up-to-date with network changes; and other distributed and dynamic algorithms have oscillation and stability problems. This is why new routing systems should be defined to efficiently deal with the changing traffic loads and topologies.

Swarm intelligence is an evolving field yielding methods for distributed systems optimization. For the purpose of network routing, the paradigm that has already been proposed as most promising is that of ants (Heusse *et al*., 1998). These remarkable insects have the ability to discover routes to their food sources through simple primitive interactions (Goss *et al*., 1989), via a chemical substance called a pheromone. Pheromones are used for indirect communication between ants, a phenomenon known as stigmergy. There already exist many successful adaptations of ant behavior to network routing. The work by Schoonderwoerd *et al.*(1996) was the first attempt to apply it to a routing problem. Their algorithm, called ant-based control (ABC), was applied to the case of virtual circuit based on symmetric networks (e.g. identical costs associated with both link directions). For asymmetric networks (e.g. packet-switching networks in which asymmetric delay occur due to different queue lengths), this cannot be done anymore. AntNet introduced by Di Caro and Dorigo (1997; 1998) for routing in packet-switching networks outperformed all conventional algorithms on several packet-switched communications networks in their simulations. However, in the AntNet algorithm, routing is determined by means of very complex procedure and its convergence speed and routing result are still unsatisfied.

This paper describes an algorithm called Adaptive Swarm-based Routing (ASR) that in-

creases the speed of convergence and has quite good stability by using a novel variation of reinforcement learning and a technique called momentum. We implemented a small network simulator to evaluate our approach. Experimental results showed that the adaptive swarm-based routing algorithm has good adaptability to changing traffic conditions in the network.

COMMUNICATION NETWORK MODEL

Recent research in ethology suggested that ants can discover routes to their food sources through simple primitive interactions (Beckers and Deneubourg, 1992). In nature, ants lay down a thin layer of signaling chemicals called pheromones, wherever they travel to find food. When other ants detect these pheromones, they instinctively follow the path the chemicals mark. The thicker the pheromone trail, the more likely other ants will follow the path. Moreover, the ant itself is not a complex insect. For any ant considered individually, it has very simple and limited behaviors. In fact, all of its movements are based on immediate reactions to its surroundings or to its fellow ants. However, ants are social insects. By acting as a group, they represent a highly structured and complex social organization. Such general properties have resulted in interest in applying the methodology to problems in networks routing.

In order to apply this trail laying and following behavior to networks routing, we replaced the routing tables in the network nodes by tables of probabilities, which we call pheromone tables, see Table 1, as the pheromone strengths are represented by these probabilities.

Each row in this table corresponds to a neighbor and each column to a destination. The entries in the table are the probabilities that the next-hop is a specific neighbor. These probabilities are used by ants to allow them to randomly explore the network and possibly find new and better routes. Then once the routes are discovered, the next-hop probabilities are updated to reflect the new discoveries. All routing table entries conform to the constraint:

$$\sum_{j\in L_n} P_{jd} = 1, \quad d \in [1, N], \quad L_n = \{neighbor(n)\} \qquad (1)$$

where P_{jd} expressing the goodness (desirability), under the current network-wide routing policy, of choosing j as next node when the destination node is d.

Throughout this paper we consider that the topology of the network is modeled by a directed weighted graph $G=\{N, A\}$ with a node set N and an arc set A. At each node n of N, an array D^n stores the last two estimated delays, $D^n_{j,d}(t)$ and $D^n_{j,d}(t-1)$, experienced by the ants from node n to each destination node d via neighboring node j.

Table of probabilities and the array D^n can be seen as memories local to nodes capturing different aspects of the network dynamics. The latter maintains absolute time estimates to all the nodes, while the former gives relative probabilistic goodness measures for each link-destination pair under the current routing policy implemented over all the network.

Table 1 Pheromone table for node *n*

Next node	Destination node			
	1	2 …		N
1	P_{11}	P_{12}	...	P_{1N}
2	P_{21}	P_{22}	...	P_{2N}
…	...	...	...	...
L	P_{L1}	P_{L2}	...	P_{LN}

ADAPTIVE SWARM-BASED ROUTING

In the ASR algorithm, two types of ants are defined: (a) Forward Ant, denoted F_{ant}, which will travel from the source node s to a destination d. (b) Backward Ant, denoted B_{ant}, will be generated by a forward ant F_{ant} in the destination d. It will come back to s following the same path traversed by F_{ant}, with the purpose of using the information already picked up by F_{ant} in order to update routing tables of the visited nodes. For greater intelligibility, we only describe the steps of the algorithm for a single forward ant F_{ant} originating from any node s of N

and going toward any node d of N. We first detail the updates of the array D^n and then, we look at the probabilities of the next node selection in the routing tables.

At regular intervals, from every network node s, a forward ant F_{ant} is launched, with a randomly selected destination node d. Each F_{ant} selects the next hop node using the information stored in the routing table. The route is selected, following a random scheme, proportionally to the goodness (probability) of each neighbor node. When a traveling F_{ant} headed towards its destination, it keep track of its journey from s to d and of the time elapsed since its launching time to arrive at the current node n. The cost $d^n_{j,d}$ is defined as the sum of all the costs from node n to destination d via neighboring node j.

If a cycle is detected, that is, if an ant is forced to return to an already visited node, all the memory about the cycle's node is destroyed. When the destination node d is reached, the forward ant F_{ant} generates B_{ant}, transferring to it all its memory. The backward ant takes the same path as that of its corresponding forward ant, but in the opposite direction. They do not 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.

In order to increase the speed of convergence and avoid high oscillation, a technique called momentum is used. When the arrival of B_{ant} on the current node n from a node d via neighboring node j updates the estimated delay of $D^n_{j,d}$, a fraction of the previous change of estimated delay is added. This additional term tends to keep the weight changes going in the same direction. This update rule is also called the general delta rule and can be expressed as follows:

$$D^n_{j,d}(t) = D^n_{j,d}(t) + \eta((1-\alpha)\Delta D^n_{j,d}(t) + \alpha \Delta D^n_{j,d}(t-1)) \quad (2)$$

$$\Delta D^n_{j,d}(t) = d^n_{j,d} - D^n_{j,d}(t) \quad (3)$$

$$\Delta D^n_{j,d}(t-1) = D^n_{j,d}(t) - D^n_{j,d}(t-1) \quad (4)$$

where η is a learning rate parameter (experimentally $\eta = 0.2$). α is the momentum parameter. In all the experiments we ran, we observed that the introduced momentum is a very effective mechanism that seeks to incorporate a memory in the learning process, increases the stability of the scheme, and helps to increase the learning efficiency of the network. Depending on the characteristics of the problem, the best value to assign to the parameter α can vary, but if α ranges from 0.2 to 0.45, performance does not change appreciably.

We introduce a very efficient improvement that updates at once all estimated delays corresponding to intermediate nodes visited by the agent. We apply the well-known Bellman's principle of decomposing an optimal path into optimal sub-paths. Every intermediate node on the path $n \to d$ is treated like a destination and its associated estimated delay is updated according to the relative cost between that visited node and the current updating ones. This can efficaciously utilize the information gathered by forward ant.

According to the estimated delay $D^n_{j,d}(t)$, the routing table on n, which gives the probability $p_{j,d}$ of selecting each arc (n, j) linked to n for a packet with destination d, is then periodically recalculated as follow:

$$p_{j,d} = \frac{\left(\frac{1}{D^n_{j,d}(t)}\right)^{\beta}}{\sum_{l \in L_n}\left(\frac{1}{D^n_{l,d}(t)}\right)^{\beta}} \quad (5)$$

where β is a non-linearity parameter taken superior to 1 to favor short paths (we found that setting β to 4 was a reasonable choice).

Traditional routing methods generally choose the path with minimum cost between a pair of nodes. This can lead to "bottlenecks", because all traffic congests the best path, while other paths are left without use. In this algorithm, routing tables are

used in a probabilistic way not only by the ants but also by the data packets, which means that the way the routing tables are built in ASR is well matched with a probabilistic distribution of the data packets over all the good paths. This diminishes the congestion probability in more inexpensive links, and improves the overall network performance.

MODIFICATIONS PROPOSED TO ASR

One possible problem with this ASR algorithm is that the distribution of probabilities eventually "would freeze" with a probability value, near to one, with the rest of them remaining with insignificant values (Barán and Sosa, 2001). Thus, in any node, ants and data packets would mostly choose the output line with the highest probability. In order to prevent this, the simulation system is to define a noise factor of q_0, and $q_0 \in (0,1)$. Such that at every time step an ant has probability q_0 of choosing a purely random path, and probability $(1-q_0)$ of choosing its path according to the pheromone tables on the nodes. With this, the ants could discover new and better paths. Then, both the delay and throughput could potentially improve.

In order to accelerate convergence of the routing tables, an initialization of each node routing table that reflects previous knowledge about the network topology is proposed. Furthermore, an initial greater probability value is assigned to the neighboring nodes j that simultaneously could be destinations d. Then the initial probability in the routing table is given by:

$$p_{j,d} = \begin{cases} \dfrac{1}{L_n} + \dfrac{2(L_n - 1)}{3L_n^2} & \text{if } j = d \\ \dfrac{1}{L_n} - \dfrac{2}{3L_n^2} & \text{if } j \neq d \end{cases} \tag{6}$$

If the destination d is not a neighbor node, then a uniform distribution is initially assumed:

$$p_{j,d} = 1/L_n \tag{7}$$

Of course, Eqs.(6) and (7) satisfy constraint Eq.(1).

EXPERIMENTAL RESULTS

In order to validate our algorithm, we compared it with Di Caro and Dorigo's AntNet that use similar forward agents but with more complex back-propagating agents and a different routing table update procedure. Their AntNet system has been shown to outperform OSPF and Bellman-Ford algorithms in their simulations. In this paper we use a simulated network that is composed of 14 nodes and 21 bi-directional links. The topology and the propagation delays are shown in Fig.1. The bandwidth is 1.5 Mbit/s for every link. Every node is a router identical to the others. At each simulation time step, incoming packets are pushed to the buffer corresponding to the chosen outgoing link. These buffers are supposed to be infinite: packets are queued but never discarded and there is no congestion control other than the routing algorithm itself. Each node removes the packet in front of its queue, examines the destination of this packet and uses its routing table to send the packet to one of its neighboring nodes.

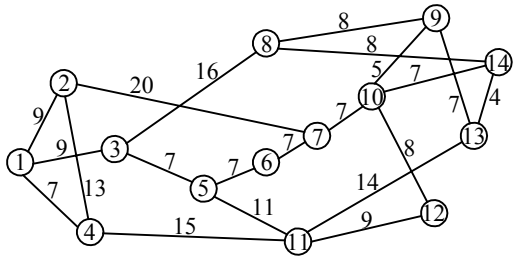

Fig.1 Network topology

Numbers within circles are node identifiers, while numbers on links are propagation delays in msec.

Both algorithms presented here use probabilistic routing, i.e. the routes are selected, following a random scheme, proportionally to the routing table values that give the probability of selecting a given destination node. Compared to deterministic routing, this method makes it possible to achieve load balancing, as flows in the network are naturally split among the different available routes.

In our experiments, the packets were emitted from each node to a random destination following Poissonian distribution. All the packets appeared similar to the link: this does not allow us to compare algorithms by considering the amount of data carried by each routing packet, only their number is critical. And, we studied the reaction of the algorithms to simulated events like bandwidth changes. For AntNet we set the (c, a, a', ε, h, t) parameters as defined in Di Caro and Dorigo (1997) to (2, 10, 9, 0.7, 0.04, 0.5). Concerning ASR, we observed that the algorithm was very robust to internal parameters tuning. We did not finely tune these parameters that were previously given in the text at the moment the parameters were discussed and we chose (0.2, 0.2, 4) for (η, α, β) parameters.

To evaluate the result of simulation we used two measurements: average delay on the network and throughput. The average delay experienced by all the packets that arrived during a given interval of time gave another view of the general status of the network. The throughput is the number of packets that went through the network during one unit of time.

Fig.2 compares the average packet delay and throughput for ASR and AntNet, and clearly shows ASR algorithm's faster convergence. During the first several time steps, the average packet delivery delay were small because the packets destined for distant nodes had not yet reached their destinations, and statistics are available only for packets destined for nearby nodes (with small delivery delays). As distant packets started arriving, the average packet delay increased. The next peak was due to the delay and local saturation resulting from the values as signed to the routing table at the beginning. While learning was still in progress, eventually the learning converged, and each of the curves settled down, indicating a stable routing.

We will study the algorithm responses to sudden modifications in the network by changing the bandwidth on two links. At the beginning of the simulation, every link bandwidth was assigned to 1.5 Mbit/s. At time step 1000, a bandwidth of 0.5 Mbit/s and 2.5 Mbit/s had been respectively used for link (8,9) and link (8,14), and the traffic that originally went through link (8,9) immediately saturated. Delays increased and a new load balance had to be found. AntNet responded with a strong transient before returning to normal. This transient decreased immediately with ASR algorithm. The peak of waiting packets after the bandwidth changed around time step 1000 was followed by a burst in the mean delay. This was due to the presence of a large number of packets saturating the queue buffers after the modification. The presented algorithms reacted in a similar way to link failures. ASR algorithm dynamically adapts its response to the bandwidth changes with only a small transitory period. These properties were highly interesting and could contribute to Quality of Service for network communications.

Unbounded FIFO queues were used in the current simulations for simplicity. In the real world, the queue buffers of the network routers are finite, leading to possible congestion in heavily loaded pa-

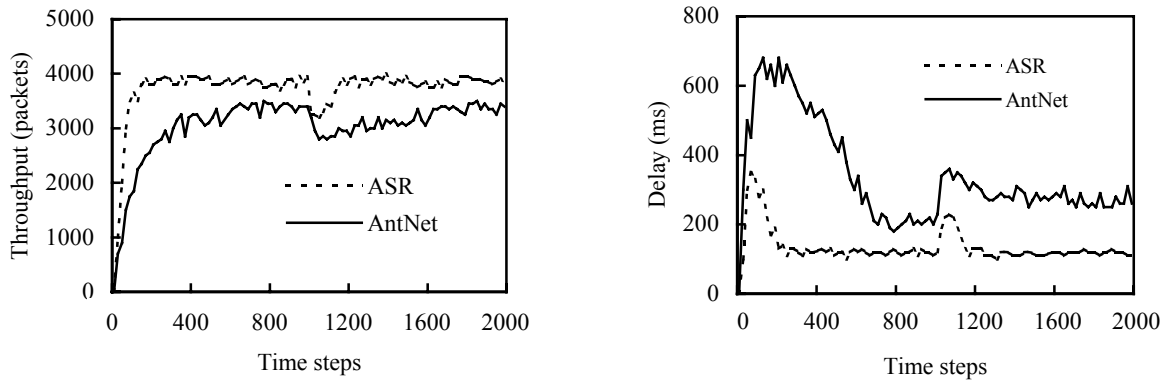

Fig.2 The average packet delay and throughput for ASR and AntNet

rts of the networks. Extension of the ASR to address the problem of finite buffer networks is another important future direction. This extension would make ASR a more realistic routing strategy that does not only route optimally, but can also sustain higher loads in finite buffer networks to avoid congestion and adapt quickly to changing network topology and traffic patterns.

CONCLUSION

In this paper, we have presented a new scheme based on swarm intelligence for routing in communication networks. By using a novel variation of reinforcement learning and a technique called momentum, our proposed algorithms can react and deal with changes of network. Reported results showed clearly that ASR is better than AntNet in performance and in behavior; that is, converging rapidly toward a good stable delay value after an initial transitory phase. We believe that ASR algorithm is of the highest for offering differentiated services, fault tolerant networks, and dealing with communication bursts in a timely manner.

References

Barán, B., Sosa, R., 2001. AntNet routing algorithm for data networks based on mobile agents. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, **12**:75-84.

Beckers, R., Deneubourg, J.L., 1992. Trails and U-turns in the selection of the shortest path by the ant Lasius Niger. *Journal of Theoretical Biology*, **159**:397-415.

Di Caro, G., Dorigo, M., 1997. AntNet: A Mobile Agents Approach to Adaptive Routing. Tech. Rep. IRIDIA/97-12, IRIDIA, Universite Libre de Bruxelles, Belgium.

Di Caro, G., Dorigo, M., 1998. AntNet: distributed stigmergetic control for communication networks. *Journal of Artificial Intelligence Research*, **9**:317-365.

Goss, S., Aron, S., Deneubourg, J.L., Pasteels, J.M., 1989. Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, **76**:579-581.

Heusse, M., Snyers, D., Guérin, S., Kuntz, P., 1998. Adaptive Agent-driven Routing and Load Balancing in Communication Network. Proc. ANTS'98, First International Workshop on Ant Colony Optimization, Brussels, Belgium, p.15-16.

Schoonderwoerd, R., Holland, O., Bruten, J., Rothkrantz, L., 1996. Ant-based load balancing in telecommunication networks. *Adaptive Behavior*, **5**(2):169-207.