AN ADAPTIVE ANT-BASED ROUTING ALGORITHM USED ROUTING HISTORY IN DYNAMIC NETWORKS

Kazuyuki Fujita, Akira Saito, Toshihiro Matsui, Hiroshi Matsuo

Deparment of Electrical and Computer Engineering,
Nagoya Institute of Technology,Gokiso-cho,Showa-ku,Nagoya,466-8555,JAPAN
fujita@mars.elcom.nitech.ac.jp,matsuo@elcom.nitech.ac.jp

ABSTRACT

Recently, small and highly efficient personal digital assistants such as cellular phone are developed, and mobile communications are being indispensable in life. The ad hoc network is observed in such circumstances. However, conventional routing algorithms that keep the optimum routing is difficult to perform flexible routing in dynamic topology network, such as the ad hoc network. This paper proposes *Ants-Routing with routing history*(ARH) and *Ants-Routing with routing history and no return rule*(ARHnr), that can perform a robust routing by selecting stochastically the good route, and learn quickly the route by using routing history. ARH and ARHne adapt reinforcement learning to the routing algorithm.

1. INTRODUCTION

In recent years, services provided on mobile communications go on diversifying with according to an explosive popularization of a cellular phone. In such circumstances, the ad hoc network is observed as a new medium of communication environment. The ad hoc network is the network that is autonomously constructed by mobile terminal(in the following, called "node"), and it enables to communicate and share the information between the nodes at any time, anywhere, freely. The characteristics of the ad hoc network are stated as follow.

1. The network topology varies dynamically according to movement of a node.
2. Each node communicates with wireless multi-hop communications.
3. No centralized control node is existed.
4. Each node is equivalent, and they construct the network in which bidirectional communication is possible.

This paper proposes ARH and ARHnr as stochastic routing algorithm that choose a suitable route and efficiently acquire the information of a route, when the nodes communicate man-to-man in such above environments. ARH and ARHnr are the routing algorithm adapting reinforcement learning. They regard each node as agent, and cope with the dynamic environments by learning the route. Additionally, they can perform a robust routing by selecting stochastically the good route, and efficiently acquire the informations of a route by using routing history.

This paper is organized as follows. In section 2, we describe the problem of conventional routing algorithm. In section 3, ARH and ARHnr are proposed. In section 4, ARH and ARHnr are estimated using computer simulation. The rest of the paper provides the conclusion of this study.

2. THE PROBLEM OF CONVENTIONAL ROUTING ALGORITHM

Major problems of conventional routing algorithm[1][2] with reinforcement learning are necessity of a lot of learning step and packets for learning. In particular, when there are many packets, or when movement of the node is intense, the route may not converge on a good route because learning speed can not follow environmental change.

Moreover, the problem peculiar to stochastically routing algorithm such as Ants-Routing and AntNet is *routing-lock*[3]. Routing-lock is a phenomenon in which a route is fixed because a routing probability converge in the vicinity of one during the progress of learning. When change of topology takes place and it becomes impossible to use the routes according to this phenomenon, a lot of time is required to discover the new route. In addition, although a new efficient route is discovered, selecting the conventional route will be continued without the ability of discovering a new one.

To overdue these problems, in this study, we pro-

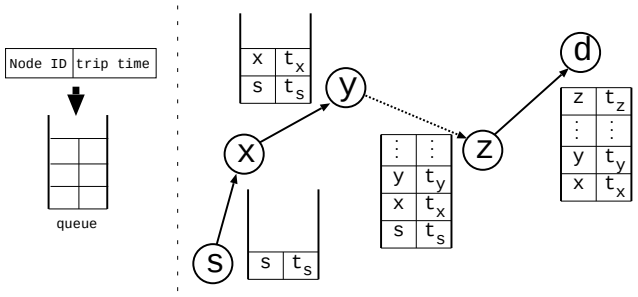

Figure 1: Transmit of routing history

pose *Ants-Routing with routinghistory*(ARH) based on Ants-Routing[4] and AntNet[5] that improves learning efficiency by recording routing history in a message packet. We also propose *Ants-Routing with routing history and no return rule*(ARHnr) that improves tolerance over routing-lock by enhancement of *No return rule* included in Accelerated Ants Routing[6].

3. ANT ROUTING WITH ROUTING HISTORY(ARH)

3.1. Selecting next route

In ARH, each node refers the routing table and determine next hop according to the stochastic values on the table, when they send to the message packets. Unlike the algorithm that select next hop deterministically, it is not a fatal problem for ARH that a loop arises for a route. Hence, ARH performs a robust routing in dynamic environments, as Ants-Routing. The stochastic values satisfied the following normalizing conditions for each destination node d.

$$\sum_{z \in \text{neighbor of } k} P_k(d, z) = 1 \qquad (1)$$

$P_k(d, z)$ denotes the sending probability from k to z as next hop, and z is the neighbor node of k.

3.2. Updating process

Each node stores its own ID and sum of transmission time, message processing time, and other information as routing history in a message packet, and send to next hop(Fig. 1). The size of queue to store routing history is fixed. If the size of stored routing history is larger than the size of queue, the oldest history is removed, and new history is stored. It is because reliability of the oldest history is low. A node that accepts a message packet is trained by *backward exploration*. The training phase is performed to the routing table of which its destinations correspond to the relay nodes that have been recorded in queue.

The probabilistic table is updated using the following equations,

$$\Delta p = \gamma^h \cdot \frac{k}{f(t_{d'})} \quad \gamma \in (0,1),\ k > 0 \qquad (2)$$

$$P_y(d', x) = \frac{P_y(d', x) + \Delta p}{1 + \Delta p} \qquad (3)$$

$$P_y(d', z) = \frac{P_y(d', z)}{1 + \Delta p} \quad z \in \text{neighbor of } y, z \neq x \qquad (4)$$

$P_y(d', x)$ denotes the sending probability from y to x as next hop, d' is the destination node, z is neighbor node of y, k is learning rate, $f(t_{d'})$ is a non-decreasing function of $t_{d'}$, $t_{d'}$ is elapse time from the message is generated, γ is a discount rate and h is the number of hop from y to d'.

Δp denotes the amount of change of the stochastic value. In equation (3), sending probability from y to x as next hop is updated. In equation(4), the stochastic values of neighbor node of y without x are normalized. According to Δp, the great value of Δp is assigned to the nearer and higher reliability route and the small value is assigned to the farther and lower reliability route. This updating process makes the possibility to learn efficiently by providing the appropriate Δp.

3.3. Ant routing with routing history and no return rule (ARHnr)

Routing-lock is also a problem in ARH, similar to those of Ants-Routing or AntNet as described in Section 2. To solve the problem, we propose ARHnr that adds enhanced no return rule to ARH. No return rule is introduced into Accelerated Ants Routing[6].

No return rule functions to eliminate *return rule* while selecting next node. In generally, eliminating *back track path* significantly degrade searching performance in a trial-and-error algorithm. However, probability of discovering another detour is high in the ad hoc network in which the node may have many links to others. Thus, Accelerated Ants Routing can efficiently search the route at the early phase of training.

Moreover, in this study, no return rule is enhanced as following.

- If selecting probability without just before node is smaller than infimum, selection is random.

In the case that back track path is always eliminated, if once wrong route is selected, the probability that a wrong one will continue being selected henceforth is high. It has a significantly bad influence on routing. Moreover, the situation that probability of selection

node except just before node is smaller than infimum tends to happen, when routing-lock occurred. In this situation, to select randomly next hop will discover a new route and increases a possibility of cancelling routing-lock.

As a solution to routing-lock, although there is the method to add noise[5] or periodically averaging a probability of selection [3], even when not related to routing-lock, they make route selection produce futility because of randomly selecting or periodically canceling learning data. Moreover, the method that solves routing-lock by assigning minus Δp[3] needs to sense routing-lock, thus it is difficult to determine Δp. Avoidance routing-lock by no return rule does not need to sense routing-lock. It simply and efficiently discovers a new route and it makes possible to solve fixing route.

4. EXPERIMENTAL RESULTS

4.1. Conditions of simulation

In this section, we present the result of extensive simulations that were performed to evaluate the proposed algorithms, and to compare their behavior with OSPF, AntNet, Ants-Routing and Accelerated Ants Routing (Acc.Ant). To compare ARHnr with Acc.Ant, no return rule that has been enhanced as explained in subsection3.3 is added to Acc.Ant.

On single simulation cycle, a node can be executed by one of following steps,

- Receive packet(if a packet to my node is in a queue)
- Relay packet(if a packet to other node is in a queue)
- Generate packet(generation rate is 5% regardless of existence of message in queue)
- No operation(others)

In this experiment, packet generation rate is 5% regardless the existence of message in queue, and source and destination nodes are selected randomly.

4.2. Node model

All packets that are accepted from neighbor nodes or generated are stored in queue, and first packet is received or sent to next hop(Fig. 2). Source and destination nodes are selected randomly. Since AntNet and OSPF send updating packet preferentially, they have two queues for normal message and prior process. However, one packet is processed at once.

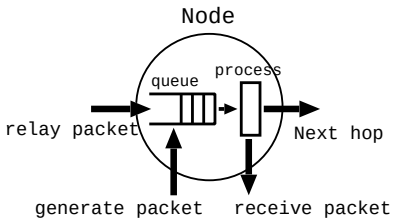

Figure 2: Node model

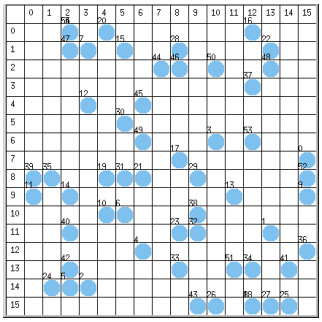

Figure 3: Grid field

4.3. Topology and dynamics of networks

Dynamic of the network uses in our simulations is shown in Fig.3. Fifty six mobile nodes are located randomly on 16 × 16 grid. Each node can directlly communicate within three blocks.

One mobile node which is selected randomly is moved one grid(north, east, west, south). To examine an ability of routing on dynamic topological environment, the parameter of mobilization occurrence rate MO is changed from 100% to 0%. MO =100% means mobilization is occurred every one simulation step, and 0% means no mobilization occurred. Furthermore, mobile node can not go on the outside of the grid.

The result is significantly influenced by the form of network in this dynamic environment. Thus, the following three items are the same in each algorithm.

- Initial location of node
- Movement of node
- Generation of the message packet

Moreover, the goal of this experiment is to estimate effectiveness of selecting route. Thus, nodes move randomly under the condition that all node can be certainly communicated with other all node.

The simulation that consists of 50000 steps was performed 100 times. Mobilization occurrence rate (MO) is changed 0%, 0.01%, and 0.1%. In OSPF, updating packet is generated when topology varies. In AntNet,

generation of a message stops during 20 steps after *forward ant* was generated. The *forward ant* is generated every 200 steps, and searching probabiliry is 0.2%. In Ants-Routing, learning rate is 64. In Acc.Ant, learning rate is 64, 2-step backward is performed, and the infimum in no return rule is 1.0×10^{-7}. In ARH and ARHnr, learning rate is 6, the number of routing histories is 64, discount rate is 0.6, and the infimum is 1.0×10^{-7}.

4.4. Simulation Result

4.4.1. $MO = 0\%$

$MO = 0\%$ means no mobilization occurred. First, we count the number of packets. In each algorithm, it is about 30 under the condition that stable routing is performed. When the situation that the number of packets is 300 or more is continued 3000 or more steps, routing should fail. Table 1 shows the number of convergence and the average value of the number of converged steps taken for packet to reach destination node in 45 trials which all algorithms succeeded.

In spite of fixed network, OSPF has the worst convergence rate. OSPF does not consider processing latency at node, and congestion control is entrusted to implementation of router in general. Thus, in this model, if congestion occurred to node on shortest route, can not be avoided by OSPF. However, ARH, ARHnr, and Acc.Ant can acquire good effective route according to consider processing latency. Thus, these algorithms have high convergence. Although AntNet and Ants-Routing seem to approach optimal route after 50000 steps, it is clear that the training speed is slow.

If congestion occurred by increasing packet in network before training advances, the probability that routing fails is high. Compare with other algorithm, training speed of AntNet is slow, and its rate of convergence is low.

Figure 4 shows a relation between simulation steps (X axis) and average steps taken for packet to reach destination node (Y axis) at $MO = 0\%$ in 45 succeeded trial.

4.4.2. $MO = 0.01\%, 0.1\%$

Figure 5 and Tab.2 show the result at $MO = 0\%$. At $MO = 0.01\%$, 29 trial succeeded in all algorithms.

In AntNet and OSPF, it is clear that the number of convergence is smaller than the other routing algorithm. These are not a practical range. In ARH, the number of convergence decreases owing to routing-lock. However, ARHnr and Acc.Ant keep high convergence rate. These algorithms show high robustness. Moreover, in ARHnr, the average value of the number of converged steps taken for packet to reach destination node is the smallest. ARHnr can acquire good effective route.

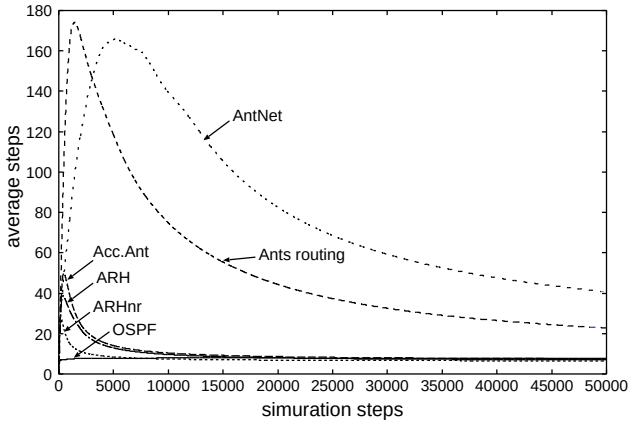

Figure 4: Average step (MO=0%)

Table 1: Number and average step of convergence (MO=0%

	Number of convergence	average steps
OSPF	59	7.8
AntNet	61	12.7
Ant routing	88	8.5
Acc.Ant	99	6.7
ARH	98	6.3
ARHnr	100	6.3

The tendency of convergence is the same as that of the case of $MO = 0\%$. The convergence of ARHnr is the fastest. Although Acc.Ant has equivalent robustness to ARHnr, ARHnr is superior with regard to learning speed and efficiency of selecting route.

Figure 6 and Tab.3 show the result at $MO = 0.1\%$. In this case, 45 trial succeded in three algorithm, Acc.Ant, ARH, and ARHnr.

The number of convergence decreased in each algorithm. In particular Ants-Routing and ARH, the decrease is remarkable. However, ARHnr and Acc.Ant keep high convergence rate.

In addition, in ARHnr, the average value of the number of converged steps is the smallest, and the convergence is the fastest. Thus, ARHnr can perform efficiently robust routing in environments that topology varies violently.

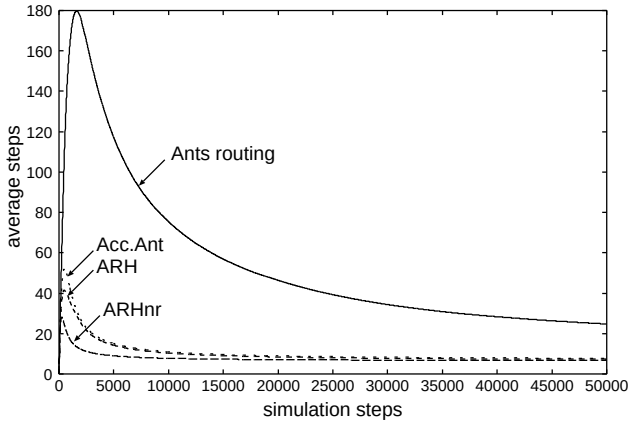

Figure 5: Average step (MO=0.01%)

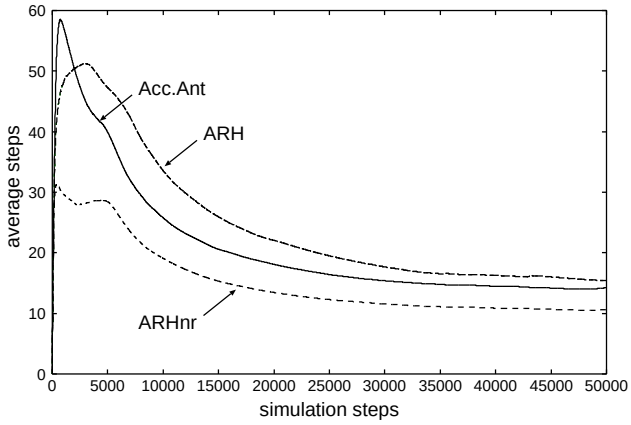

Figure 6: Average step (MO=0.1%)

Table 2: Number and average step of convergence (MO=0.01%)

	Number of convergence	average steps
OSPF	52	9.8
AntNet	38	16.1
Ants routing	73	10.4
Acc.Ant	95	7.0
ARH	87	6.4
ARHnr	96	6.4

Table 3: Number and average step of convergence (MO=0.1%)

	Number of convergence	average steps
OSPF	45	-
AntNet	11	-
Ants routing	30	-
Acc.Ant	87	11.7
ARH	51	11.1
ARHnr	87	8.8

5. CONCLUSION

In this study, new stochastic routing algorithms named ARH and ARHnr for dynamic topology network like the ad hoc network are proposed. In ARH, routing history achieves quickly learning and efficient route. Moreover, in ARHnr, no return rule achieves efficiently search at the early phase of training.

In particular, ARHnr can quickly solve routing-lock and it is excellent in discovering a new route in dynamic environment. Future reserch will be addressed to make a comparison with the other routing algorithm such as DSR and AODV proposed in IETF.

6. REFERENCES

[1] M.Littman and J.Boyan, A Distributed Reinforcement Learning Scheme for Network Routing, *Proc. the 1993 International Workshop on Applications of Neural Networks to Telecommunications*, pp.45–51, Hillsdale, NJ, 1993.

[2] S.Kumar and R.Miikkulainen, Dual Reinforcement Q-routing: An On-line Adaptive Routing Algorithm, *Volume 7 in Intelligent Engineering Systems Through Artificial Neural Networks (ANNIE-97)*, pp231–238, St. Louis, MO, 1997.

[3] K.Oida and A.Kataoka, Lock-Free AntNets and thier Adaptability Evaluations,*IEICE Transactions on Communications*, Vol. J82-B, No.7, pp.1309–1319, Jul. 1999.

[4] D.Subramanian, P.Druschel, and J.Chen, Ants and Reinforcement Learning: A Case Study in Routing in Dynamic Networks,*Proc. the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, Nagoya, Japan, 1997.

[5] Gianni Di Caro and Marco Dorigo ,AntNet: Distributed Stigmergetic Control for Communications Networks, *Journal of Artificial Intelligence Research*, vol.9, pp.317–165, 1998.

[6] H.Matsuo and K.Mori, Accelerated Ants Routing in Dynamic Networks, *2nd International Conf. on Software Engineering, Artificial Intelligence, Networking & Parallel/Distributed Computing*, pp.333–339, Nagoya, Japan, 2001.