computing@tanet.edu.te.ua
www.tanet.edu.te.ua/computing

ISSN 1727-6209
International Scientific Journal of Computing

A PROPOSAL AND EVALUATION OF THE FAST RECONNECT AD-HOC NETWORK ROUTING PROTOCOL

Kazuhiro MIZOGUCHI[1)], Shinichi FURUSHO[1)], Teruaki KITASUKA[1)], Tsuneo NAKANISHI[1)2)], Akira FUKUDA[1)]

[1)] Graduate School of Information Science and Electrical Engineering, Kyushu University,
Kasuga-koen 6-1, Kasuga-shi, Fukuoka, 816-8580, Japan,
{kazuhiro, furusho, kitasuka, tun, fukuda}@f.csce.kyushu-u.ac.jp
http://f.csce.kyushu-u.ac.jp
[2)] System LSI Research Center, Kyushu University,
Kasuga-koen 6-1, Kasuga-shi, Fukuoka, 816-8580, Japan,

Abstract: *An ad-hoc network works without any infrastructures. It consists of wireless mobile nodes .In this paper, we propose an ad-hoc network routing protocol, called FR-DSR(Fast Reconnect Dynamic Source Routing), which is an improved DSR.When a route is disconnected, FR-DSR can reconnect fast by using prepared spare routes. During communication, spare routes are prepared by sending route check packets through routes in a cache, and an additional route request packet is sent if a spare route is broken. We show that FR-DSR gives better performance than DSR through simulation experiments.*

Keywords: – *Ad-Hoc network, Wireless network, Routing, DSR, QoS*

1. INTRODUCTION

Today, communication service form has been changing with rapid progress of information and communication technology. These services must have infrastructure such as base stations and a backborn network. Therefore the technology called ad-hoc network has been researched. The ad-hoc network is a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure. In the ad-hoc network, if there is a long distance between a source node and a destination node, nodes between them must relay packets for them to communicate. Each mobile node in the ad-hoc network operates not only as a host but also as a router.

Since the ad-hoc network consists of mobile nodes, the communication route is frequently changed. Since the route change takes time, it causes deterioration of quality in real-time communication such as voice or video communication. In this paper, we propose FR-DSR (Fast Reconnect Dynamic Source Routing) which is an improved DSR so that reconnection time at the route change is short.

2. DSR AND ITS ISSUES

2.1 DSR

DSR [1, 2, 3] is one of the typical ad-hoc network routing protocols and has the features of both on-demand routing protocol and source routing protocol. In on-demand routing, looking for a route starts when a source node wants to send packets. In source routing, only a source node controls a route for relay nodes to send packets according to this route information. DSR consists of two mechanisms; route discovery and route maintenance.

2.2 Route Discovery

At first, a node looks for a route when it wants to send packets. Each node has a route cache. The route cache contains routes to the node which has communicated before. If the source node has routes to the destination node, these routes are contained in data packets to send them. If there is no route in the cache, the source node broadcasts a RREQ (Route Request) packet. A RREQ packet includes the source node ID, the destination node ID, the request ID, the

list of passed nodes, and so on. Receiving a RREQ packet, each relay node adds the own ID to the list in the RREQ packet and re-broadcasts it. In this way, a RREQ packet is spread over network and reaches the destination node. The destination node returns a RREP (Route Reply) packet to the source node when it received a RREQ packet. In this way, a route is established and communication between the source node and the destination one can start.

2.3 Route Maintenance

Each node does not have to periodically broadcast update packets because DSR is an on-demand routing protocol. Only source node maintains the route. Consider that the source node, S, and the destination node, D, are communicating through the route S-A-B-D. When the link A-B is down, the following action is done. After the node A transfers a packet to the node B, it tries to confirm that the node B receives the packet. This confirmation is used by a standard part of the MAC protocol. The node A decides that the link A-B is down if the node A can not confirm it. The node A deletes this route from the route cache. Then the node A sends a RERR (Route Error) packet to the source node S. The RERR packet includes the information that the link A-B is down. After receiving the RERR packet, the node S deletes the route including the link A-B from the route cache. The node S sends packets to the node D through other routes in the route cache, if there are other routes. If there is no route to the node D in the route cache, the node S looks for new route.

2.4 Issues of DSR

In the ad-hoc network, a route is cut in the case of node movement, *etc* and communication becomes impossible in many cases. When a route is cut, a source node tries to reconnect by using route maintenance and route discovery mechanism described in Sec. 2.2 and 2.3. Then the source node looks up a route to the destination node from the route cache. However the reliability of cached routes is low since the routes in the route cache may be old. When the current route is broken, a possibility that the cached routes are valid may be low.

We consider the case where the routes in the route cache entirely becomes invalid. The source node tries to send packets through a route in the route cache when the current route is broken. When the source node decides the link-down by using route maintenance, it looks for a route from the route cache again. If there are no routes to the destination node, the source node broadcasts the RREQ packets. This process wastes the time. In other words, it takes long time to reconnect a route in this case.

3. FR-DSR PROTOCOL

3.1 FR-DSR

The cause of the problem described in Sec. 2.4 is low reliable routes in a route cache. If a reliability of routes in a route cache is high, communication will be

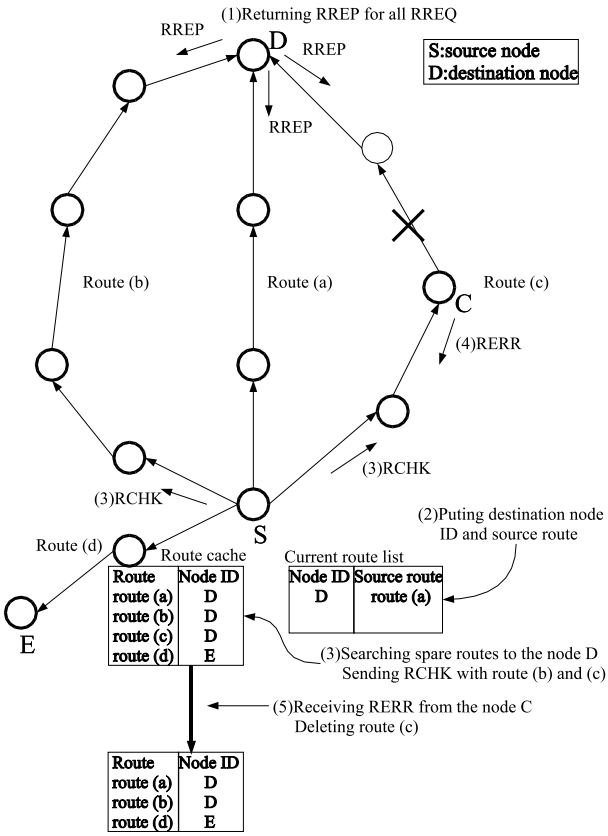

Fig.1 – FR-DSR

able to immediately restart even when a current route is broken. Of course, the source node does not have to look for a route.

From the discussions above, we propose an improved DSR, called FR-DSR, so that a source node prepares highly reliable spare routes in the route cache by deleting invalid routes from the route cache and getting new routes into the cache. Therefore communication can immediately restart by using these spare routes.

An outline of FR-DSR is shown in Fig.1. We consider the case where the source node S sends packets to the destination node D. It is supposed that the node S does not know a route to the node D at first and the route (c) becomes impossible to use during communication.

Each node has a current route list to put the destination node and the route into it. FR-DSR works as follows.

1. The source node S broadcasts a RREQ packet to find a route to the destination node D. The destination node D receives RREQ packets and

returns RREP packets for all RREQ packets.

2. When the source node S receives RREP packets, it starts communication to the node D with the route (a).At that time, the node S stores the route (a) into the route cache and sends the data packets through this route. When the node S sends the data packets, the node S puts ID of the destination node D and the source route (a) into the current route list.
3. The source node S periodically checks the current route list. Then the source node S takes a pair of the destination node ID and its route from the current route list. Then the node S searches spare routes to the destination node from the route cache. The node S finds the routes (b) and (c) to send RCHK (Route Check) packets to the destination node D through these routes.
4. The RCHK packet sent through the route (b) arrives at the destination node D. The other side, the RCHK packet sent through the route (c) can not arrive at the destination node D due to down of the route (c). After finding this, the node C returns a RERR packet to the source node S.
5. When the source node S receiving the RERR packet finds the route (c) is broken, it deletes the route (c) from the route cache. If the number of spare routes to the destination node becomes below a fixed number, the source node S tries to get spare routes by broadcasting a RREQ packet.

FR-DSR consists of three main mechanisms. They are explained below.

3.2 Getting Spare Routes

As described in Sec. 2.2, with DSR, each node broadcasts a RREQ packet when it looks for a route. With original DSR, a destination node returns only one RREP packet for a RREQ packet which is received first. That is, the destination node ignores existing other routes. With FR-DSR we propose in this paper, the destination node returns RREP packets for all received RREQ packets as in the same way of [5].

3.3 Current Route List

The goal of this paper is to have reconnecting time short by improving on-demand routing DSR. With on-demand routing, a source node does not have to send any periodic packets. Route check that will be described in Sec. 3.4 later is only done during communication. Therefore its advantage keeps.

In this paper, each node has a current route list to keep above advantage. When the source node sends data packets, it puts the destination node ID and the source route which are included in data packets into the current route list. Each node periodically checks the current route list. If there are the destination node ID and the source route in it, each node dose route check described in Sec. 3.4 for a pair of the node ID and its route. They clear the current route list after route check.

3.4 Route Check

Confirmation of spare routes in a route cache is done as follows.

Each node periodically checks the current route list. In the case where there are some routes in the current route list, the source node searches other routes to the destination node from the route cache. Then it sends RCHK (Route Check) packets through these routes. When the destination node receives a RCHK packet, this route can be used and the source node keeps it as a spare route. In the case where the route used by a RCHK packet is broken, a relay node finding link-down returns a RERR packet. When the source node receives this RERR packet, it deletes this route from the route cache. This action is the same as an original RERR packet of data packets. After that, the source node counts the number of spare routes to the destination node. It is the number of sending RCHK packets. If the number of spare routes is below the fixed number, the source node broadcasts a RREQ packet to discover spare routes.

4. EVALUATION THROUGH THE SIMULATION

We performed two simulation experiments using the ns-2 network simulator [4]. In these simulations, flow control which is an option of DSR is disabled.

4.1 Experiment-1

A situation that FR-DSR works most efficiently is as follows, where Experiment-1 is performed. We use Fig.1 to explain it. The source node S communicates with the destination node D. First routes discovered by route discovery are only the route (a) and (c). The source node S communicates through the route (a). After that, the route (b) can be used and the route (c) cannot be used due to node movement. The route (a) is broken at the time of the state that the route (b) is valid and the route (c) is broken. We measure packet delay and compare FR-DSR with DSR. The number of spare routes which the source node prepares for each destination node is set to be two.

Original DSR acts as follows. After the route (a) is broken, the source node S receives a RERR packet.

The source node S sends data packets through the route (c) in the route cache. Since the route (c) is already broken, a node on the route (c) returns a RERR packet to the source node S. The source node S starts the route discovery process because there is no

route to the node D in the route cache. It gets the route (b) by using route discovery, and sends data

packets through route (b).

On the other side, FR-DSR proposed in this paper acts as follows. The source node S sends a RCHK packet to the spare route (c). Then the source node S starts the route discovery process because the number

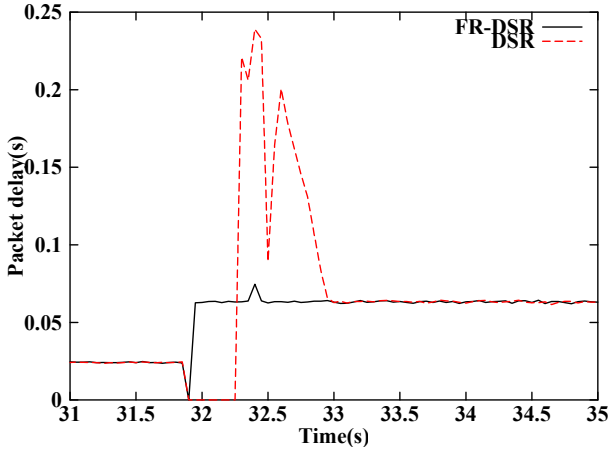

Fig.2 – Delay of each packet around the time when the route (a) is broken

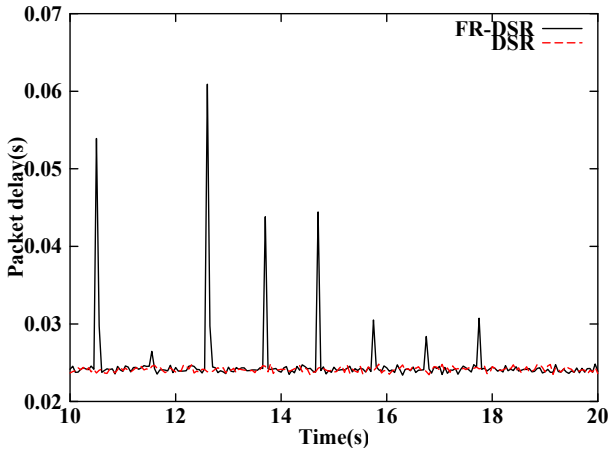

Fig.3 – Delay of each packet before the route (b) is broken

of spare routes is one below two. This action is repeated during communication. After that, the node S finds the route (b) by using route discovery. The node S stops route discovery because there are two spare routes, but the route check is continued. After that, the node S finds that the route (c) is broken by using route check. Then the node S restarts the route discovery process because it has only one spare route. The node S receives a RERR packet after the route (a) is broken. The source node S uses route (b) to send data packets.

An environment of Experiment-1 is described below.

- Transport layer protocol : UDP
- Interval of sending packets : 0.05s
- Interval of route check : 1~1.1s
- Simulation time : 60s

4.2 Result of Experiment-1

Fig.1 and 2 show delay of each packet around the time when the route (a) is broken and the time before the route (a) is broken, respectively. The horizontal axis is the time when packets were sent, and the vertical axis is the delay until a sent packet reaches a

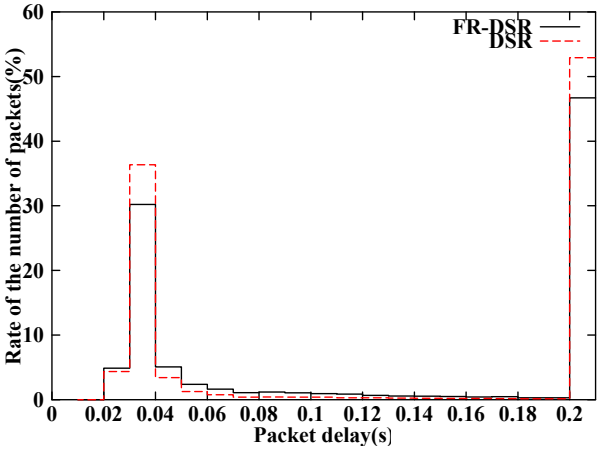

Fig.4 – Histogram of the packet delay

Table 1. Rate of invalid packets

QoS restriction[s]	FR-DSR[%]	DSR[%]
0.20	46.71	52.91
0.15	49.24	54.00
0.20	53.37	55.65
0.05	64.76	61.92

destination node. Dropped packets are counted as no delay packet in Fig.2.

The route (b) is broken at 31.9 seconds. We can see certainly dropped packets and the rapid rise of delay. With DSR, packets are dropped and the delay rises rapidly. It takes about 1.1 seconds to converge on the stable state. On the other side, with FR-DSR one packet drops but next packet can reach. It takes about 0.1 seconds to converge on the stable state. This reason is that with FR-DSR the communication restarts through route (b) immediately after the route (a) is broken. Converging on the stable state means a communicating route is not broken.

As shown in Fig.3, with DSR the packet delay is almost fixed in the stable state. With FR-DSR, the delay periodically increases. This cycle is the same as the cycle (1~1.1 seconds) of route check and route discovery. In the other word, RCHK and RREQ packets are periodically sent and they make delay of data packets a little long. Therefore the packet delay increases. The packet delay before the route (a) is broken is different from that after. This reason is the difference of routes. The route (a) before down is 4 hops and the route (b) after down is 10 hops.

This packet delay in the stable state is less than the reconnecting time by changing a route. Although the maximum delay with DSR is 0.239 seconds due to reconnection, with FR-DSR, it is 0.075 seconds.

From these results, FR-DSR is better than DSR since the packet delay is short when a route is broken and the communication can be immediately restarted.

4.3 Experiment-2

Experiment-2 is performed in the state where each node moves at random. A source node and a destination node are placed on the points (200, 200) and (800, 800) in the simulation field (1000m × 1000m), respectively. We suppose that these two nodes do not move. In addition, there are 100 mobile nodes in this field. The transmitting radius of each mobile node is 250m. The source node communicate with the destination node using these nodes. Mobile nodes move at random where pause time is 0 second and moving speed is 0~5m/s. The simulation is performed 10 times, and we take the average of them. Simulation time is 200 seconds. The other parameters are the same as those in Experiment-1.

4.4 Result of Experiment-2

Fig.4 shows the histogram of the packet delay. The horizontal axis is the packet delay, and the vertical axis is the rate of the number of packets at this packet delay. Packets of which delay is greater than 0.2 seconds are shown as one of 0.2 seconds in Fig.4. The QoS restriction time in this paper is decided to be 0.2 seconds. Packets of which delay are over 0.2 seconds are treated as invalid. This reason is that packets which reached over fixed time may cause deterioration of QoS of real-time restriction.

As shown in Fig.4, the rate of packets of which delay are over 0.2 seconds are 46.7% with FR-DSR, and 52.9% with DSR. These packets include packets of which delay increases due to reconnection or collisions, dropped packets, and packets left in the queues of relay nodes. Packets left in the queues of relay nodes are packets which are not sent by relay nodes despite that there are routes to the destination node. This is a fault of ns-2 network simulator used in this paper. For this fault, there are packets left in the queues until the end of simulation and of which delay is over 100 seconds. Therefore packets delayed over the QoS restriction 0.2 seconds increase. As shown in Fig.4, the rate of these bad packets with FR-DSR becomes less than one with DSR, from about 53% to about 47%.

Table 1 shows the rate of invalid packets when the QoS restriction time is changed between 0.05 and 0.20 seconds. The short is thought as more severe communication in real-time restriction. As shown in Table 1, DSR becomes better than FR-DSR when the QoS restriction time is 0.05 seconds. FR-DSR is better than DSR when the QoS restriction time is 0.10 seconds.

With DSR, when a route is changed, packets drop continuously and the packet delaying over the QoS restriction time increases. However with FR-DSR, this bad event does not arise. On the other side with FR-DSR, delay increases periodically due to route check and route discovery, but it is below the QoS restriction time. Delay at reconnecting is longer than delay at route check and route discovery. Therefore a performance of FR-DSR becomes lower than one of DSR if the real-time restriction becomes severe. The diverging time is 0.07 seconds in this simulation.

5. CONCLUSION

We have proposed FR-DSR which improves the reconnecting delay. With FR-DSR, each node has the spare routes in advance. Through simulation experiment using the ns-2 network simulator, we have shown that with DSR the reconnecting time by route change becomes 1.1 seconds and one with FR-DSR becomes 0.1 seconds. Moreover we have shown that a performance of FR-DSR is better than one of DSR in the case the QoS restriction time is over 0.07 seconds. Future works include the following.

- Congestion of network by RCHK packets and RREQ packets
- Power consumption
- Selection of spare routes

6. ACKNOWLEDGMENT

This research has been partially supported by JPSP Grant-in Aid for Young Scientists (B) (KAKENHI 15700062) and NTT.

7. REFERENCES

[1] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hic Wireless Ad Hoc Network Routing Protocols," Mobicom'98, pp.85-97, 1998.

[2] X.-Y. Hong, K.-X. Xu, and M. Gerla, "Scalable Routing Protocols for Mobile Ad Hoc Networks," IEEE Network, Vol. 16, Issue 14, pp. 11-21, July/August 2002.

[3] C. E. Perkins et al., Ad Hoc Networking, Addison-Wesley, 2001.

[4] K. Fall and K. Varadhan (Eds.), ns notes and documentation, The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, April 2002.

[5] S. Furusho, K. Moriwaka, T. Kitasuka, T. Nakanishi, and A. Fukuda, "Load Balancing Routing for Wireless Ad-hoc Network," IPSJ DICOMO2002, pp.413-416, July 2002 (in Japanese).

Kazuhiro Mizoguchi *received the B.E. degree in computer science from Kyushu University, Japan, in 2003. Since 2003, he has been a master course student of Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan. His research interests include mobile computing and ad-hoc network.*

Shinichi Furusho *received the B.E. degree in computer science from Kyushu University, Japan, in 2002. Since 2002, he has been a master course student of Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan. His research interests include mobile computing and ad-hoc network.*

Teruaki Kitasuka *received the B.E. degree in information science from Kyoto University, Japan, in 1993 and M.E. degree in information science from Nara Institute of Science and Technology, Japan, in 1995. From 1995 to 2001, he worked for the Sharp Corporation, where he developed the personal computer. Since 2001, he has been a research associate of Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan. His research interests include mobile computing, embedded systems, parallel and distributed systems, compiler, and computer architecture.*

Tsuneo Nakanishi *received the B.E. degree in communication engineering from Osaka University, Japan, in 1993; and the M.E. and D.E. degrees in information science from Nara Institute of Science and Technology, Japan, in 1995 and 1998, respectively. From 1996 to 1998, he was a research fellow of the Japan Society for the Promotion of Science. From 1998 to 2002, he was an assistant professor of Graduate School of Information Science, Nara Institute of Science and Technology, Japan. Since 2002, he has been an associate professor of Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan. His research interests include compilers, embedded systems and parallel computing. He is a member of the ACM, the IEEE Computer Society and the IPSJ.*

Akira Fukuda *received the B.E., M.E., and PhD degrees in computer science and communication engineering from Kyushu University, Japan, in 1977, 1979, and 1985, respectively. From 1977 to 1981, he worked for the Nippon Telegraph and Telephone Corporation, where he engaged in research on performance evaluation of computer systems and the queuing theory. From 1981 to 1991 and from 1991 to 1993, he worked for the Department of Information Systems and the Department of Computer Science and Communication Engineering, Kyushu University, Japan, respectively. In 1994, he joined Nara Institute of Science and Technology, Japan, as a professor. Since 2001, he has been a professor of Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan. His research interests include embedded systems, system software(operating systems, compiler, and run-time systems), mobile computing, parallel and distributed systems, and performance evaluation. He received 1990 IPSJ(Information Society of Japan) Research Award and 1993 IPSJ Best Author Award. He is currently the chair of SIG System Evaluation in IPSJ. He is a member of the ACM, the IEEE Computer Society, the IEICE, the IPSJ, and the Operations Research Society of Japan.*