IEICE TRANS. COMMUN., VOL.E91–B, NO.9 SEPTEMBER 2008

PAPER IEICE/IEEE Joint Special Section on Autonomous Decentralized Systems Theories and Application Deployments

A Hybrid Greedy Routing with Location Information for Mobile Ad Hoc Networks*

Hiroshi NAKAGAWA[†a)], **Kazuyuki NAKAMARU**[†], ***Student Members***, **Tomoyuki OHTA**[†], ***and*** **Yoshiaki KAKUDA**[†], ***Members***

SUMMARY Recently, in mobile ad hoc networks, routing schemes using location information have been proposed. Most of these schemes assume that the source node already knows the location information of the destination node. However, since all nodes are always moving, it is difficult to apply this assumption to the real mobile ad hoc environment. In order to cope this difficulty, this paper presents a new routing scheme HGR (a Hybrid Greedy Routing with location and velocity information), which considers the location and velocity information of the destination node and the neighboring nodes. In HGR, when a source node creates a route to a destination node, the future location of neighboring nodes and the destination node predicted by the source node is calculated using these location and velocity information. And the source node sends data packets to the neighboring node that is the closest to the destination node based on these predicted location and velocity information. This paper shows that HGR achieves high data delivery ratio and fewer overheads for the route creation and maintenance through simulation experiments.

key words: *ad hoc networks, geographical routing*

1. Introduction

Recently, along with the development in wireless device techniques, Mobile Ad-hoc NETworks (MANETs) [1], [2] which consist of mobile nodes and wireless links without base stations have become popular. Since MANET can form a temporary network autonomously, it can be applied for various purposes such as civilian use, disaster recovery and military use. Many routing schemes [3], [4] have been proposed to send data packets from a source node to a destination node for MANETs.

Generally, as the number of nodes with GPS function become popular, it is expected that routing schemes using the location information [5], [6] become more available. However, many routing schemes using the location information assume that a source node has already known the location information of the destination when it sends data packets. Under this assumption, it is difficult to control networks and experiment more realistically because a destination node moves. Therefore, routing schemes, GLS [7] and Octopus [8] which get the location information of the destination node and send data packets have been proposed. Both GLS and Octopus divide the space into horizontal and vertical. Especially, in Octopus, each node shares the location information of all nodes that are in the same horizontal and vertical strips. However, Octopus floods a great amount of control packets to get the location information of the destination node. Additionally, when a source node sends data packets, a fixed route from the source node to the destination node is not made and the data delivery cannot quite give assurance.

Therefore, in this paper, we propose HGR (Hybrid Greedy Routing with location and velocity information), which aims to improve the data delivery assurance. HGR consists of three mechanisms, location update, getting the location information of the destination node and making and maintaining a route. And HGR uses not only the location information but also velocity information of nodes. In order to show the effectiveness of HGR, we compare HGR with Octopus using network simulator QualNet ver.3.9 [9].

The rest of the paper is organized as follows. In Sect. 2, we describe GPSR and LAR. In Sect. 3, we describe Octopus. In Sect. 4, we propose a novel routing scheme HGR. In Sect. 5, we analyze the routing performance of HGR through simulation experiments. In Sect. 6, we conclude this paper with future works.

2. Related Works

Routing schemes for MANET are classified into proactive schemes and on-demand schemes. In Sect. 2.1, we describe GPSR which is one of the representative proactive schemes using the location information. In Sect. 2.2, we describe LAR which is one of the representative on-demand schemes using the location information.

2.1 GPSR: Greedy Perimeter Stateless Routing

GPSR assumes that a source node knows the location information of the destination node in advance. Each node

Manuscript received December 26, 2007.
Manuscript revised April 18, 2008.

[†]The authors are with the Faculty of Information Sciences, Hiroshima City University, Hiroshima-shi, 731-3194 Japan.

[*]This work was supported in part by the Ministry of Internal Affairs and Communications of Japan under Grant-in-Aid for Strategic Information and Communication R&D Promotion Programme (SCOPE), the Ministry of Education, Science, Sports and Culture of Japan under Grant-in-Aid for Scientific Research (C) (No.18500062) and Young Scientists (B) (No.18700070), and Hiroshima City University under their research grants. This work is based on "(A Hybrid Routing with Location Information for Mobile Ad Hoc Networks)," by (Hiroshi NAKAGAWA, Tomoyuki OHTA, Kenji ISHIDA, and Yoshiaki KAKUDA) which appeared in Proc. IEEE International Symposium on Autonomous Decentralized Systems (ISADS 2007), Sedona, USA, March 2007, ©2007 IEEE.

a) E-mail: hiroshi@pe.ce.hiroshima-cu.ac.jp

DOI: 10.1093/ietcom/e91–b.9.2806

periodically broadcasts a beacon packet with its own identifier (ID) and the location information to the neighboring node and keeps neighbor nodes' ID and the location information. When a source node wants to send data packets, at first the source node calculates distances between the destination node and neighboring nodes, including source node itself. And the source node compares these distances, it selects the closest node to the destination node in its own communication range and forwards the data packet to this node. The node which receives the data packet also repeats these procedures until the destination node receives the data packets. However, if the source node or a node that forwards a packet becomes the closest node to the destination node, the node cannot forward data packets. In this case, at first this node calculates the straight line between it and the destination node. Then the node searches a node in the counterclockwise direction from the straight line and the node which is found out first becomes the next hop node.

2.2 Problems of Previous Routing with Location Information

Many routing schemes with location information, including GPSR and LAR, assume that a source node has already known the location information of the destination. This assumption is effective if the destination node is fixed the location information like a server. However, when a destination node moves, it is difficult to resolve this assumption. If the practical environment is considered, it is necessary to experiment with the method including getting location information like GLS [7].

3. Octopus: A Routing Scheme Gets the Location Information of the Destination Node

Octopus is a typical routing scheme which gets the location information of the destination node and forwards data packets. Octopus assumes that each node gets its own location information using a positioning device such as a GPS. Under this assumption, Octopus consists of three functions, (1) Location update, (2) Getting the location information of the destination node and (3) Forwarding data packets. In this section, we describe these three functions of Octopus.

3.1 Location Update

Octopus divides the ad hoc network into horizontal and vertical strips using latitude and longitude. The strip width is constant and known to all nodes. Each strip has a unique ID. And the area which overlaps horizontal and vertical strips is called the grid. Under this network, each node maintains two tables, the neighboring table and the strip table, and these tables have node ID, timestamp, location information of the node, belonging to strips ID. Each node periodically updates the neighboring table to trade HELLO packets with each other. The strip table is updated by the following procedures:

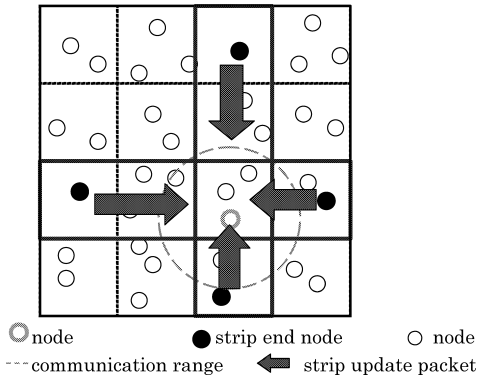

Fig. 1 Strip Update.

1. The nodes that are at the end of each strip, called the strip end node, make Strip Update packets.
2. This packet includes ID of strip updated, the direction of this packet, ID of the target node which is at the opposite end of strip and this packet sender and timestamp.
3. The strip end node additionally screens nodes belonging to the same strip from the neighboring table and records these nodes' ID, the latest location information, the belonging strip ID and timestamp in Strip Update packet.
4. The strip end node floods this packet within the strip for updating information of the strip.
5. If the node that is out of the updated strip receives this packet, it drops it.
6. If the node which is in the updated strip receives this packet, it performs the procedure 3 and 4.
7. If the target node receives this packet, it stops forwarding.

Figure 1 shows dividing network into strip and updating Strip Update packets. For these procedures, each node knows ID and location information of node which is in its own communication range and the same strips.

3.2 Getting the Location Information of the Destination Node

If a source node wants to send data packets, it tries to get the location information of the destination node at first. Figure 2 shows getting the location and information of the destination node. In Octopus, Getting the location information of the destination node is the following procedures:

1. The source node makes a QUERY packet, recording the source node ID, the destination node ID, the location information of the source node and the strip ID to which the source node belongs.
2. The source node sends two QUERY packets to its north-most and south-most neighboring nodes in its square or in adjacent squares in its vertical strip.
3. The node which receives the QUERY packet searches the destination node ID in its own neighboring table

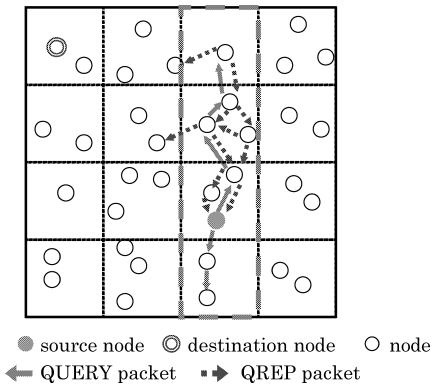

Fig. 2 Getting the location information.

and strip table.

4. If the destination node ID is not in two tables, the node forwards the QUERY packet according to the procedure 2.
5. If the destination node ID is in either one of the two tables, the node records the location information of the destination node, strip ID to which the destination node belongs and timestamp in the received QUERY packet and it change the Query packet to the Reply packet.
6. The node forwards the REPLY packet by flooding in its vertical strip.
7. If the searching in the vertical strip fails, the horizontal strip is searched by the same procedure after a certain period of time.

3.3 Forwarding Data Packets

Octopus uses Greedy Forwarding to forward data packets. Greedy Forwarding is the method that selects a next hop node that is the closest to the destination node in its own communication range and forwards data packets to that node. The node that receives data packets also repeats Greedy Forwarding and data packets are forwarded to the destination node. However, if the packet forwarding node is the closest to the destination node, it dose not forward data packets. In this case, this node decides the virtual destination node that is the closest node to the destination node in its own strip table and selects the next hop node that is the closest to the virtual destination node by Greedy Forwarding. When the next hop node receives data packets, it forward to the destination node by Greedy Forwarding.

4. HGR: Hybrid Greedy Routing with Location and Velocity Information

4.1 Problems of Octopus

Octopus has the following problems:

- When a source node gets the location information of the destination node, searched area is limited by a strip.
- A Reply packet is flooded.

Table 1 Difference between Octopus and HGR.

	Octopus	HGR
Network management	strip	strip
Using of information	location	location and velocity
How to get the destination's location	limited flooding	unicast
Creating a route	No	Yes

- When data packets are forwarded, a fixed route is not created.

At first, if no nodes exist in the searched strip, there is a possibility that to get the location information of the node fails. And if the source node moves to outside of the belonging strip, it might fail to get a Reply packet. Next, although to limit the area where Reply packets are flooded such flooding rains the number of control packets. These are dependent on the density of nodes or the width of strip. Thirdly, there is no assurance that data packets arrive in the destination node since a fixed route is not created from the source node to the destination node. In addition, when the node moving speed is higher, problems 1 and 3 are more significant.

4.2 The Outline of HGR

We propose a novel routing scheme HGR overcoming the above problems of Octopus. The outline of HGR is as follows.

- Location and velocity update; updating and maintaining velocity information as well as location information.
- The method of creating a route; when a route is created, Greedy Forwarding that regards velocity information is used.
- Getting location and velocity information of the destination node; searched area is not limited and forwarding a Query/Reply packet is unicast.
- Data transfer; creating a route by Greedy Forwarding with Velocity Information.
- Maintaining a route; periodically sending the location and velocity information of the destination node and temporarily repairing the broken route.

Table 1 shows the difference between Octopus and HGR.

4.3 Location and Velocity Update

HGR also divides ad hoc network into horizontal and vertical strips using latitude and longitude as the same as Octopus. In addition to the neighboring table and the strip table, HELLO packet and Strip Update packet have the velocity information as well as the location information of each node. The velocity information is calculated by the current location, the past location and the difference time when getting this location information. Therefore, compared to Oc-

topus, additional devices are not needed. However, HGR updates horizontal and vertical strips separately. That is, HGR updates these strips half as many as that of Octopus. Therefore, as compared to Octopus, HGR reduces the control overhead but has difficultly up-to-date information of destination nodes.

4.4 Greedy Forwarding with Velocity Information

Since problems of Octopus occur in the movement of nodes, the movement of nodes should be regarded to resolve this problem. Therefore, HGR uses Greedy Forwarding with velocity information (V-Greedy Forwarding) to create a route. V-Greedy Forwarding based on the following formula:

$$[((x_d - x_n) + (V_{xd} - V_{xn})(t - t_n))^2 + ((y_d - y_n) + (V_{yd} - V_{yn})(t - t_n))^2]^{\frac{1}{2}} \quad (1)$$

x_d and y_d are x-coordinate and y-coordinate of the destination node, V_{xd} and V_{yd} are x and y velocity vector components of the destination node. Also, x_n, y_n, V_{xn} and V_{yn} are x and y-coordinate and velocity vector components of a neighboring node of the destination node. And t is the current time and t_n is the time when the location and velocity information of the neighboring node are obtained.

In the formula 1, the first and third entries show the distance between the destination node and its neighboring node. Additionally, the second and forth entries show that the relative velocity of x and y and the difference time of t_d and t_n are multiplied. Therefore, the formula 1 shows predictive relative location between the destination node and its neighboring node.

However, this formula might select a node out of its own communication range. Alternatively, a node where is close to the boundary of the communication range might be selected. If the node is selected, node movement or the fluctuation of radio wave easily breaks down the route. To deal with this problem, HGR defines a certain threshold within the communication range and selects the next hop node based on this threshold. This threshold is smaller than the communication range. When the next hop node is selected, the following formula is used:

$$[((x_s - x_n) + (V_{xs} - V_{xn})(t - t_n))^2 + ((y_s - y_n) + (V_{ys} - V_{yn})(t - t_n))^2]^{\frac{1}{2}} < threshold \quad (2)$$

x_s, y_s, V_{xs} and V_{ys} are also x and y-coordinate and velocity vector components of the source node. This formula shows predictive relative location between the source node and its neighboring node. The source node selects the next hop node that satisfies the formula 2 and is the smallest value of the formula 1 as shown in Fig. 3.

Figure 3 shows how to decide the next hop by V-Greedy Forwarding. Circles denoted by the smallest solid lines, are locations which the source node got by receiving HELLO packets. And circles of the smallest dotted line are locations which the source node predicts using the velocity of neighboring nodes. If the predicted location is not used, the source node selects the node B as the next hop node to forward a packet but the node B is already outside of the communication range of the source node, and then the source node fails to forward a packet. On the other hand, If the predicted location is used, the source node selects the node A and succeeds to forward a packet.

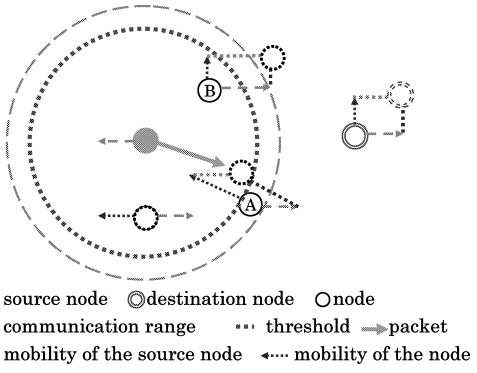

Fig. 3 V-Greedy Forwarding.

4.5 Getting Location and Velocity Information of the Destination Node

When HGR gets location and velocity information of the destination node, V-Greedy Forwarding is used to make a fixed route. The procedures to get this information are as follows.

1. A source node sets down target positions at the point of both edges belonging to vertical strip.
2. The source node sends QUERY packets to each target position by V-Greedy Forwarding.
3. When the node receives a QUERY packet, it makes the query routing table and records the previous hop ID.
4. If this node does not keep the location and velocity information of the destination node, it forwards the packet by V-Greedy Forwarding.
5. If it keeps this information, it sends a QREP packet to the source node along the reverse route to the source node.
6. If the searching in the vertical strip is failed, the horizontal strip is searched by the same procedure after a certain period of time.

Figure 4 shows how to get the location and velocity information of the destination node. Therefore, to get location and velocity information of the destination node can be flexible without relying on limiting strips. And the number of control packets is reduced in comparison with Octopus since control packets are not flooded.

4.6 Creating a Route

After the source node gets location and velocity information of the destination node, it sends a RREQ packet using V-Greedy Forwarding to make a fixed route from it to the destination node. The RREQ packet includes the source node

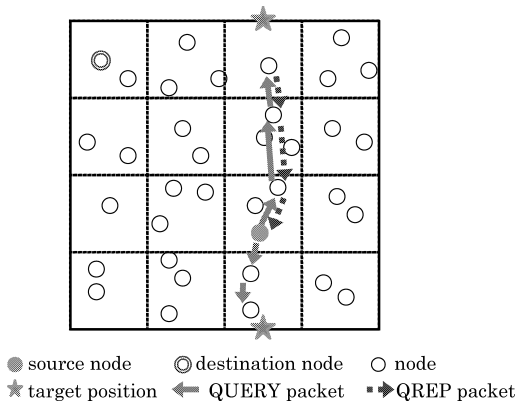

Fig. 4 Getting the location and velocity.

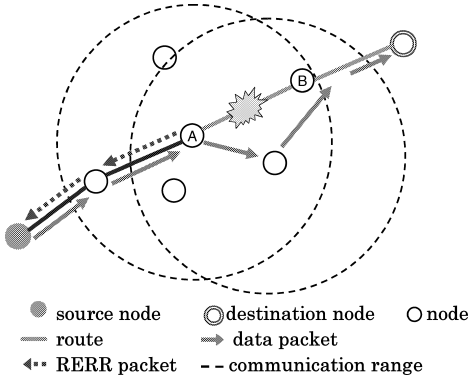

Fig. 5 Temporarily repairing the broken route.

ID, the destination node ID and location and velocity information, the next hop node ID, the packet forwarding node ID and the timestamp of making this packet. When a node receives this packet, it makes a routing table and records the packet forwarding node ID as the previous hop node ID and the timestamp of making this packet. Then it forwards this packet to the destination node by V-Greedy Forwarding. If the destination node receives this packet, it makes a RREP packet and forwards the source node with the reverse route to the source node. This packet includes the source node ID, the destination node ID, the packet forwarding node ID, the timestamp of making this packet. When the intermediate node receives this packet, it records the packet forwarding node ID as the next hop node in its own routing table. Then, the source node receives this packet, the route is created from the source node to the destination node, and data packets begin to be sent.

4.7 Maintaining a Route

4.7.1 Periodically Sending the Location and Velocity Information of Destination Node

In HGR, Using a route to transfer data packets, the destination node periodically sends a packet with its own location and velocity information to the source node. If the route is broken, the source node gets location and velocity information of the destination node before it creates a new route again. However, this method enables the source node to create a new route again immediately although the route is broken.

4.7.2 Temporarily Repairing the Broken Route

HGR temporarily repairs the broken route and creates a new route. This method is as follows.

1. The broken upstream node which is close to the source node on the broken route detects that the route is broken using Call-Back from MAC layer.
2. This node sends a RERR packet indicating that the route is broken to the source node through the created route.
3. When a node receives the RERR packet, it forwards this packet to the source node and does not erase the route information.
4. When the source node receives the RERR packet, it stops sending data packets and create a new route.
5. The broken upstream node forwards holding data packets to the neighboring node that is the closest to the broken downstream node using V-Greedy Forwarding.
6. When the neighboring node receives data packets, it searches the broken downstream node ID in its own neighboring table.
7. If the broken downstream node ID exists, data packets are forwarded.
8. If not, data packets are discarded.

Figure 5 shows how to repair the broken route temporarily. This method is grounded in the idea that the broken upstream node exists near the broken downstream node. Therefore, relying on the idea, it is possible to repair the broken route temporarily.

5. Simulation Experiments and Evaluation

GPSR [5] and LAR [6] and the method [10] on which HGR is based have been evaluated through simulation experiments in [10] and they do not have a function of getting location information.

5.1 Simulation Model and Parameters

We now evaluate HGR, Octopus and AODV, which is the most popular and high performance routing protocol, through simulation experiments. HGR, Octopus and AODV are implemented in QualNet network simulator. All nodes use the IEEE 802.11 radio and MAC model. In most of our simulations, there are 100 and 500 nodes in 1000×1000 square meters (this environment is used by experiments of the paper of Octopus) and 2500×2500 square meters. All nodes are deployed at random. Each node moves according to the random waypoint model [11]: it chooses a random destination and moves toward it with a definite speed chosen uniformly. When a node reaches the destination, it stays

during a specified period of time, chooses a new destination and begins moving toward the new destination immediately in the same speed. In our simulation experiments, the specified period of time is set to zero. Each node moves at the speed of 1, 5, 10, 15 and 20 meter per second. The number of pairs of a source node and a destination node (we call SD pairs) is 10 on 1000×1000 environment, and 30 on 2500×2500 environment. The simulation time is 300 seconds. The node's radio range is 250 meters. The interval of sending a HELLO packet is 2 seconds and the interval of sending a Strip Update packet is 10 seconds. For each set of parameters, we performed ten simulation runs with combinations of the number of nodes and the node moving speed.

On these environments, we compare HGR with Octopus and AODV in terms of the data delivery ratio, the hop count, data delivery delay and the sum of control overhead. Moreover, we evaluate these schemes in terms of the throughput on various load environments, 2500×2500 square meters when the node speed 20, SD pairs 30 and the number of node 500. Packets used in HGR and Octopus are MAC packet (that is RTS/CTS and ACK), HELLO packet, STRIP UPDATE packet (see Fig. 1), QUERY packet (that is used to get information of location and velocity of the destination node), ROUTE packet (that is used to create and maintain a route), and DATA packet.

5.2 Effect of the Threshold and Expected Location of Nodes

5.2.1 Purpose of Experiments

HGR uses the threshold of communication range and the expected location information to create a route to adapt the movement of nodes. However it is influenced by some environments of the network such as the speed of nodes and so on. We now evaluate the data delivery ratio, the hop counts, the data delivery delay and the sum of control overhead. In evaluation of the threshold, we use the several thresholds, 150, 200 and NO threshold, of communication range. And, in evaluation of the expected location information, we experiment two cases, using the expected location information or not.

5.2.2 Evaluation of the Threshold

Figures 6, 7, 8, 9 and 10 show the data delivery ratio, the hop count, data delivery delay, the sum of control overhead and the data transmission efficiency.

As shown in Fig. 6, the data delivery ratio of all of the threshold cases is the lowest at the node's speed 1. HGR uses the strip in order to maintain location and velocity information of nodes, and this method depends on the initial placement of nodes. Therefore, if there are no or few nodes in an area, it is difficult for a source node to get location and velocity information of a destination node. Additionally, if the node's speed is 1, this condition extends over a long period of time. As a result, the data delivery ratio is the lowest

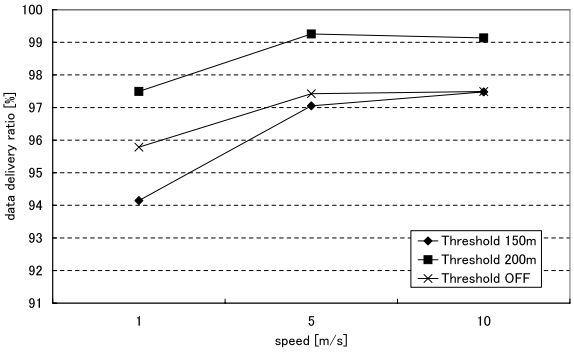

Fig. 6 The data delivery ratio.

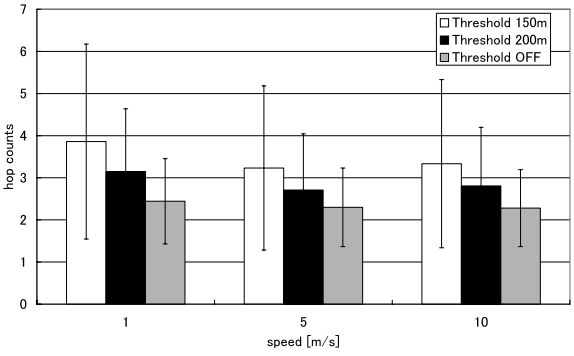

Fig. 7 The number of hop counts.

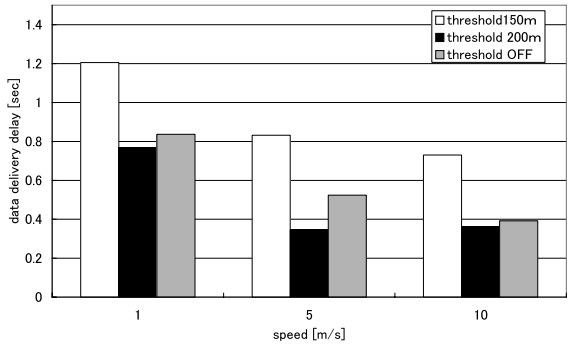

Fig. 8 The data delivery delay.

at the node's speed 1.

Considering this graph, when node's speed is 1, the threshold OFF is the best of all. This is because a stable route which is from a source node to a destination node is created and data packets are delivered absolutely without the threshold in the environment of low speed of nodes. On the other hand, when node's speed is 5 or 10, the threshold 200 is the best of all, and the threshold 150 becomes better. In the case of the threshold OFF, when node's speed becomes higher, a created route is easy to be broken and it is difficult to deliver data packets. And threshold 150 is superior to the threshold OFF on this point as the range of the threshold is enough. However, as this range is too enough, it is difficult to get location information of a destination node and create a route.

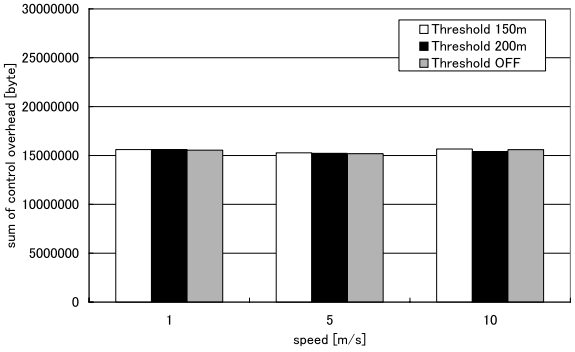

Fig. 9 The sum of control overhead.

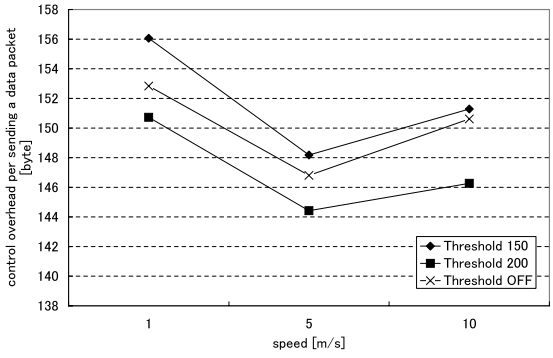

Fig. 10 The data transmission efficiency.

Next, as shown in Fig. 7, the larger the range of the threshold is, the more the hop count is. In general, data packets are delivered effectively as the hop count is low. However, to take into consideration the result of the data delivery ratio, the lowest hop count does not always make the best data delivery ratio.

Thirdly, as shown in Fig. 8, the threshold 200 is the best performance but the threshold 150 is the worst performance. This is because the data delivery delay depends on the hop count. However, the data delivery delay of the threshold OFF is larger than that of the threshold 200 although the hop count of the threshold OFF is smaller than that of the threshold 200. This is because a created route is not stable and a new route is re-created more than once with the threshold OFF.

Fourthly, as shown in Fig. 9, each value of threshold is not different in this option. This is because most of the control packets are Hello packet and Strip Update packet which nodes periodically exchange with each other without delivering data packets.

Finally, as shown in Fig. 10, the threshold 200 is the best performance. This is because that the data delivery ratio of the threshold 200 is the best and the control overhead of all thresholds is not different.

We can say that these results show that the threshold of communication range 200 is the best of three.

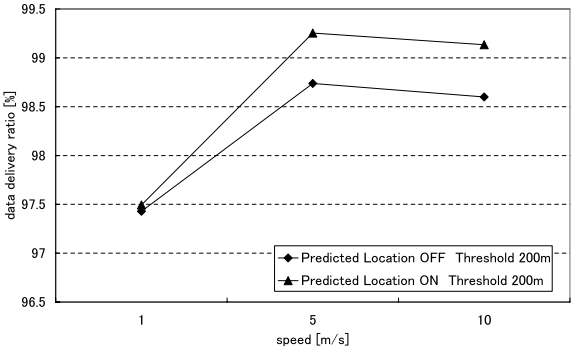

Fig. 11 The data delivery ratio.

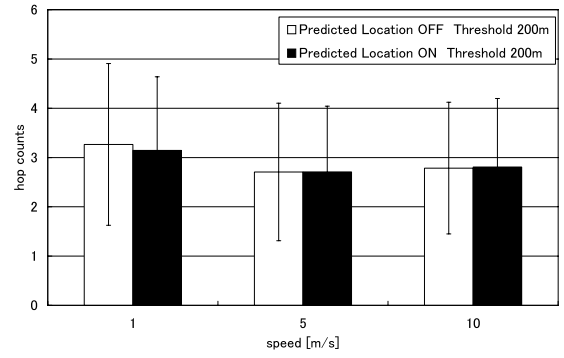

Fig. 12 The number of hop counts.

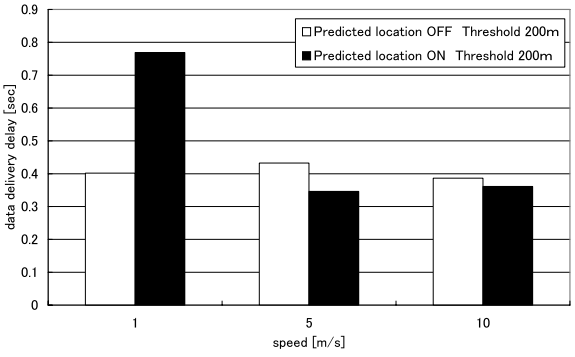

Fig. 13 The data delivery delay.

5.2.3 The Expected Location of Nodes

Figures 11, 12, 13, 14 and 15 show the data delivery ratio, the hop count, data delivery delay, the sum of control overhead and the data transmission efficiency. As shown in Fig. 11, both data delivery ratios are almost the same when the node's speed is 1. This is because the predicted location information is mostly nonfunctional when node's speed is low, and the threshold is only effective. When the node's speed becomes high, this ratio of the predicted location ON is better than that of the predicted location OFF. In this environment, the function of the predicted location information is effective and a high-precision and high-stable route is created. However, when the node's speed becomes high, it is

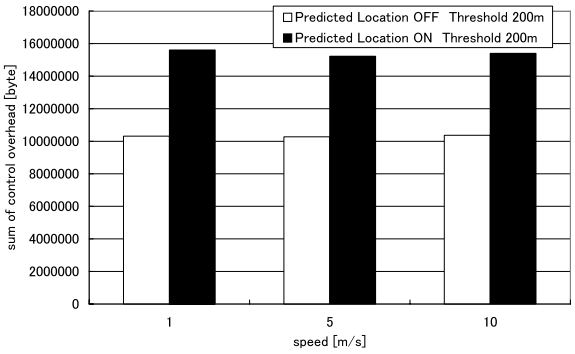

Fig. 14 The sum of control overhead.

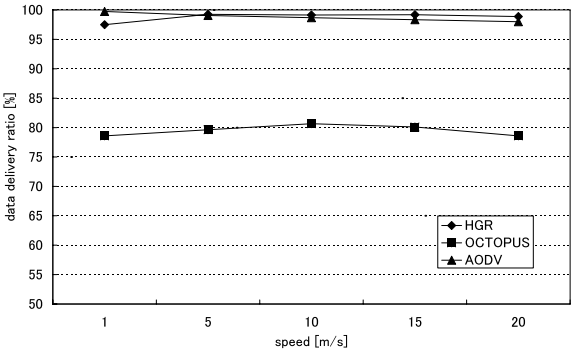

Fig. 16 The data delivery ratio.

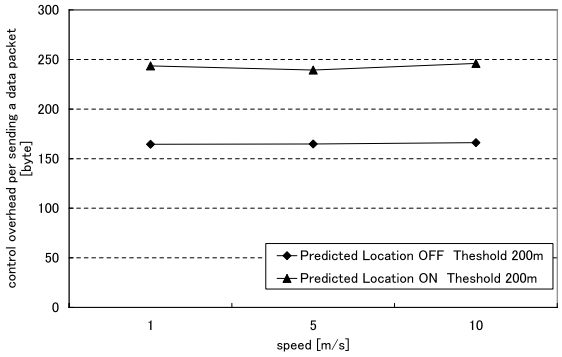

Fig. 15 The data transmission efficiency.

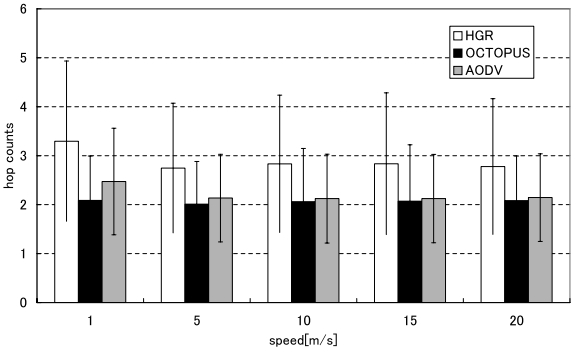

Fig. 17 The number of hop counts.

going to be more likely that the predicted location information does not work. If this function is not effective, a created route is easy to break. However, the slope of these graphs is almost the same. This is because the threshold compensates the mistake of the predicting the location information of nodes. Therefore, from the viewpoint of the data delivery ratio, both functions of the threshold and the predicted location information have a synergy effect.

Next, as shown in Fig. 12, the average and variance of the hop count of the predicted location information ON is less than that of the predicted location information OFF when the node's speed is 1. That is, the predicted location information makes a route stabilized. On the other hand, the hop count of both the predicted location ON and OFF is almost the same when the node's speed is 5 and 10. This is because the number of nodes which are able to be selected by next hop node decreases and a created route is slightly redundant. However, a created route is more stabilized because the data delivery ratio of the predicted location information ON is better than that of the predicted location information OFF. Therefore, as shown in Fig. 13, the data delivery delay decrease as the hop count decreases. Additionally, this is because the stabilized route decreases frequency of route repairing.

Thirdly, as shown in Fig. 14, the control overhead of the predicted location information ON is more than that of the predicted location information OFF. This is because velocity information (128 byte) is deleted in all packets when the predicted location information is not used.

Finally, as shown in Fig. 15, the data transmission efficiency of the predicted location information OFF is better than that of the predicted location information ON. This is because that the control overhead of the predicted location information ON is half as large again as that of the predicted location information OFF although the data delivery ratio of the predicted location information ON is better than that of the predicted location information OFF.

Therefore, when the predicted location information is used, there is the trade-off between the data delivery ratio and the control overhead.

5.3 Comparison with Octopus and AODV

5.3.1 Results for Small Network

Figures 16, 17, 18, 19, and 20 show the data delivery ratio, the number of hop counts, the data delivery delay, the control overhead and the data transmission efficiency in case of the small network.

First, as shown in Fig. 16, HGR is superior to Octopus. On the surface, Octopus certainly gets the location information of the destination node and sends data packets because HGR updates strips half as many as Octopus. However, HGR does not have to send out location information of the destination node many times once it creates the route. And HGR uses functions of the threshold and the predicted location information in order to create a sta-

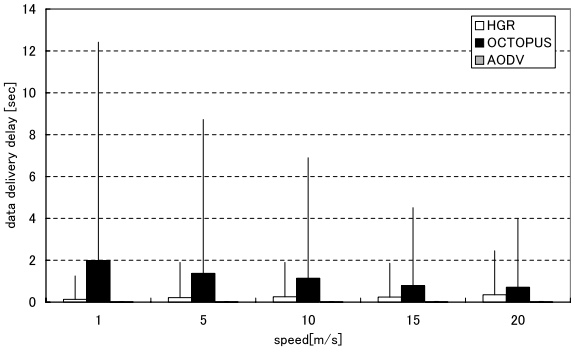

Fig. 18 The data delivery delay.

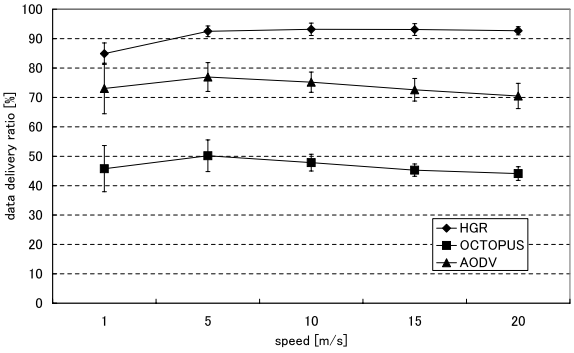

Fig. 21 The data delivery ratio.

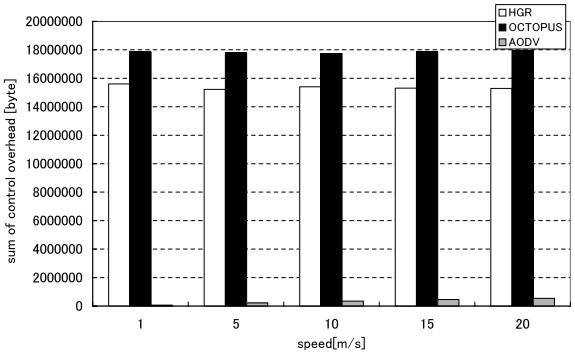

Fig. 19 The sum of control overhead.

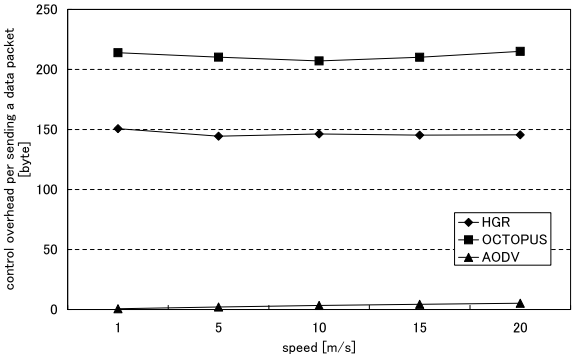

Fig. 20 The data transmission efficiency.

bilized route. Therefore, HGR is superior to Octopus from the viewpoint of the data delivery ratio. On the other hand, comparing HGR to AODV, this ratio of both two methods is almost the same. In general, AODV is familiar to be very effective in the small network. Therefore, in terms of data delivery ratio, HGR is very effective, too. However, as shown in Fig. 17 and Fig. 19, the hop count of AODV is about 2.5 but HGR is over 3, and the sum of control overhead of HGR is very larger than that of AODV. This is because HGR uses the function of threshold which tends to increase the hop count. And, in HGR, control packets which are HELLO packet and Strip Update packet are exchanged periodically and all packets including these packets are very larger than the control packets of AODV because there are location and velocity information in these packets. However, the data delivery ratio of two methods is the same, which means that HGR can create a stabilized route even when there are a lot of control packets. And as shown in Fig. 20, the data transmission efficiency of HGR is superior to that of Octopus but inferior to that of AODV. Therefore, in the small networks, to flood control packets in order to create a fixed route is more effective than maintaining a network periodically.

Finally, as shown in Fig. 18, HGR is superior to Octopus in terms of the data delivery delay although Octopus is superior to HGR in terms of the hop count. Octopus gets location information of the destination node periodically. On the other hand, HGR gets location and velocity information of the destination node only once. And once the route is created from the source node to the destination node, the destination node sends its own location and velocity information to the source node periodically through the created route. Therefore, HGR decreases the data delivery delay compared with Octopus. On the other hand, comparing HGR with AODV, AODV is superior to HGR because it does not have to get the location information of the destination node. And the variance of HGR is larger than that of AODV because HGR sometimes fails to get location information of the destination node. However, the variance of HGR is large but the average of HGR is small. This is because that HGR stably delivers data packets, once it succeeds to get the location information of the destination node.

5.3.2 Results for Large Network

Figures 21, 22, 23, 24, and 25 show the data delivery ratio, the number of hop counts, the data delivery delay, the control overhead and the data transmission efficiency in case of the large network.

First we make a comparison HGR and Octopus. As shown in Fig. 21, HGR is superior to Octopus. This big difference between HGR and Octopus is the data delivery delay as shown in Fig. 23. The average and variance of the data delivery delay of Octopus are very large in comparison with the environment 1000 *times* 1000 field. On the other hand, those of HGR do not increase, compared with. This difference is caused by that Octopus takes out location information of the destination node repeatedly. However, there

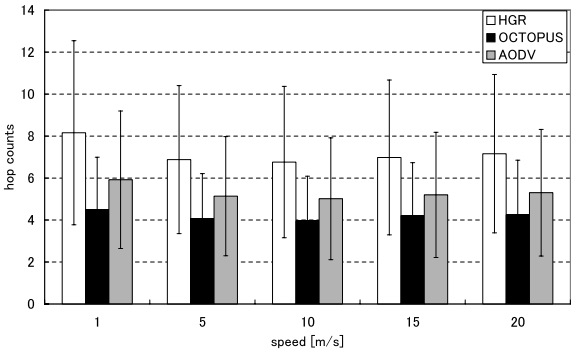

Fig. 22 The number of hop counts.

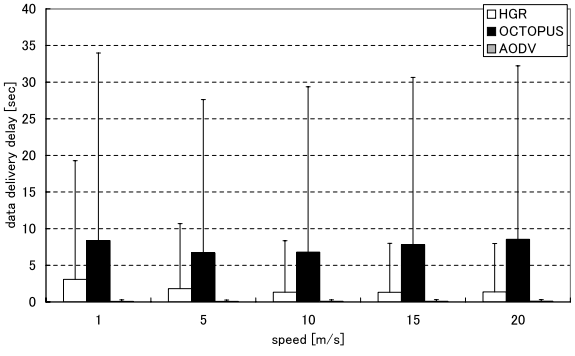

Fig. 23 The data delivery delay.

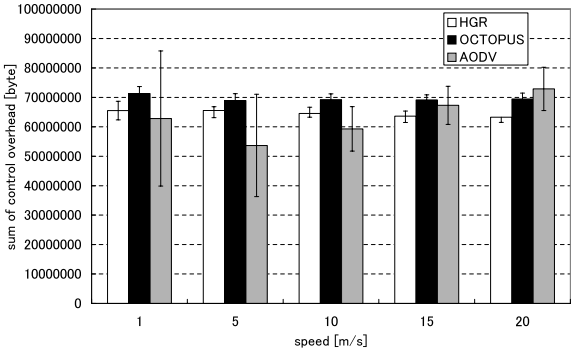

Fig. 24 The sum of control overhead.

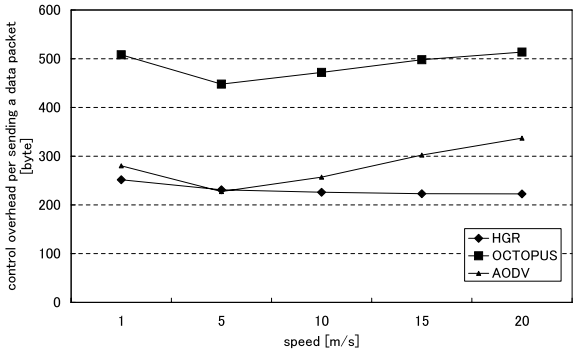

Fig. 25 The data transmission efficiency.

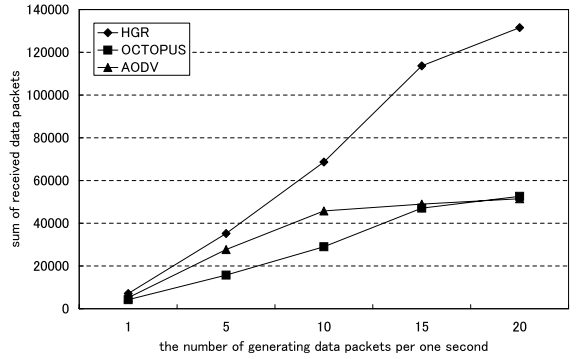

Fig. 26 The throughput.

are some areas where no nodes exist as the network area is large, and the number of taking out location information of the destination node increases. At the matters discussed in the Sect. 4.1, in this environment, it is difficult for the source node to get the location information of the destination node and it is hard to send data packets. On the other hand, in HGR, the method of getting location and velocity information of the destination node is improved and the source node stably gets this information. And then, as shown in Fig. 22, the hop count of Octopus is smaller than that of AODV. As AODV floods the control packets to create a route, this route is almost optimal. Therefore, Octopus almost never delivers data packets to destination nodes which are far from source nodes. Moreover, as shown in Fig. 24, HGR is superior to Octopus. This is because HGR updates these strips half as many as Octopus. And as shown in Fig. 25, the data transmission efficiency of HGR is very superior to that of Octopus. Especially, when the node's speed is higher, HGR uses the predicted location information to create a route and the data transmission efficiency almost remains unchanged. However, in Octopus, the sum of control overhead is almost remains unchanged but the data delivery ratio is decreasing at the node's speed high. Therefore, the data transmission efficiency of Octopus becomes exacerbated.

Next, we make a comparison HGR and AODV. As shown in Fig. 21, 22, 23 and 24, HGR is superior to AODV in terms of the data delivery ratio but AODV is superior to HGR at others points. That is, in HGR, although a route is redundant and the number of control overhead and the data delivery delay are higher, data packets are stably delivered to their destinations. In comparison HGR with Octopus in terms of the data delivery delay, HGR makes this difference as we described in the foregoing section. And in terms of the data delivery ratio and the hop count, HGR makes a route stabilized and AODV makes it minimal-length. And, as shown in Fig. 24, the number of control overhead of HGR is proportional to the number of nodes in comparison with the small network in the foregoing section. This is because packets, HELLO and Strip Update, which nodes periodically exchange with each other, increase and other packets which is used by sending data packets do not rapidly increase. On the other hand, AODV uses only packets which

are used by creating a route and there are no packets which are exchanged periodically. However, compared with the small network in the foregoing section, the control overhead rapidly increases. This is because that created route is broken and new route is re-created again and again. Additionally, in the large network, broken routes frequently happen as the hop count is large. Therefore, the control overhead of AODV is larger than that of HGR when the node's speed is 15 and 20. Figure 25 shows this discussion prominently. The data delivery ratios of HGR and AODV do not decrease widely but the sum of control overhead of AODV increases when the node's speed is higher. Therefore, the data transmission efficiency of HGR almost remains unchanged but that of AODV becomes exacerbated.

These results show that HGR gives a great important to the data delivery ratio at the expense of the hop count using functions of the threshold and the predicted location information. Therefore, HGR is superior to AODV in terms of the data delivery ratio and the data transmission efficiency.

5.3.3 Throughput

Since throughput is defined as the sum of received data packets divided by simulation time 300. Figure 26 shows the sum of the number of received data packets of three schemes with respect to the number of generated data packets per second. In HGR, when the number of generated data packets increases, the sum of received data packets equally increases. On the other hand, in Octopus and AODV, these throughputs almost remain on the same level between 15 and 20. At first, we make a comparison between HGR and Octopus. The size of a control packet of Octopus is smaller than that of HGR because the control packet of Octopus does not include the velocity information. However, as shown in Fig. 24, the sum of control overhead of Octopus is larger than that of HGR. Thus, the number of control packet of Octopus is larger than that of HGR. Therefore, data packets tend to collide with control packets when the number of generated data packets increases.

Next, we make a comparison between HGR and AODV. In AODV, when the node speed is high, fixed routes from the source node to the destination node are often broken and the control overhead increases. Therefore, in AODV as well as Octopus, data packets tend to collide with control packets when the number of generated data packets increases. As a result, in HGR, although the control overhead temporarily increases to use strip update packet, the other kinds of packets are not heavy and the number of packets is not large. And HGR is able to make a stable route to deliver data packets. Therefore, data packets do not tend to collide with control packets when the number of generated data packets increases and the throughput remains at a high level.

6. Conclusions

We have presented HGR, a hybrid greedy routing scheme with location and velocity information for MANET under the assumption that destination nodes move, and evaluated HGR. The simulation results show that the threshold and the expected location work effectively and HGR is superior to Octopus in terms of the data delivery ratio, the control overhead and data delivery delay in small and large networks. Moreover HGR is superior to AODV in terms of data delivery ratio in large networks. In particular, HGR maintains high performance on the data delivery ratio in large networks. From the evaluation results, we can say that HGR is the best routing scheme among the routing schemes proposed so far to send data packets to the destinations with getting their location information.

References

[1] C.E. Perkins, Ad hoc networking, Addison Wesley, 2000.
[2] X. Hong, K. Xu, and M. Gerla, "Scalable routing protocols for mobile ad hoc networks," IEEE Netw., vol.16, no.4, pp.11–22, 2002.
[3] D. Johnson and D.A. Maltz, "Dynamic source routing in ad hoc wireless network," in Mobile Computing, pp.153–181, Kluwer Academic Publishers, 1996.
[4] C.E. Perkins, E.M. Belding-Royer, and S. Das, "Ad hoc on demand distance vector (AODV) routing," IETF RFC 3561, 2003.
[5] B. Karp and H.T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," Proc. MobiCom 2000, pp.243–254, 2000.
[6] Y.B. Ko and N.H. Vaidya, "Location-aided routing (lar) in mobile ad hoc networks," Proc. MobiCom 98, pp.66–75, 1998.
[7] J. Li, J. Jannotti, D. Couto, D.R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," Proc. MobiCom 2000, pp.120–130, 2000.
[8] R. Melamed, I. Keidar, and Y. Barel, "Octopus: A fault-tolerant and efficient ad-hoc routing protocol," Proc. SRDS 2005, pp.39–49, 2005.
[9] "Qualnet network simulator by scalable network technologies," http://www.scalable-networks.com/
[10] H. Nakagawa, K. Ishida, T. Ohta, and Y. Kakuda, "Goli: Greedy on-demand routing scheme using location information for mobile ad hoc networks," Proc. 26th IEEE International Conference on Distributed Computing Systems Workshops (ADSN2006), p.p.1 (6pages), 2006.
[11] J. Broch, D.A. Malts, D.B. Johnson, Y.C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," Proc. ACM/IEEE MOBICOM'98, pp.85–97, 1998.

Hiroshi Nakagawa received the B.E., and M.E. degrees from Hiroshima City University, Japan, in 2004 and 2006, respectively. He is currently a Ph.D. candidate at the Graduate School of Hiroshima City University. His current research interests include mobile communication systems.

Kazuyuki Nakamaru received the B.E., and M.E. degrees from Hiroshima City University, Japan, in 2006 and 2008, respectively. His current research interests include mobile communication systems.

Tomoyuki Ohta received the B.E., M.E. and Ph.D. degrees from Hiroshima City University, Japan, in 1998, 2000, and 2006, respectively. He is currently an Assistant Professor in the Graduate School of Information Sciences, Hiroshima City University. His current research interests include mobile communication systems. He is a member of IEEE (U.S.A) and ACM (U.S.A).

Yoshiaki Kakuda received the B.E., M.Sc., and Ph.D. degrees from Hiroshima University, Japan, in 1978, 1980 and 1983, respectively. From 1983 to 1991, he was with Research and Development Laboratories, Kokusai Denshin Denwa Co., Ltd. (KDD). He joined Osaka University from 1991 to 1998 as an Associate Professor. He is currently a Professor in the Graduate School of Information Sciences, Hiroshima City University, since 1998. His current research interests include network software engineering, assurance networks and mobile ad hoc networks. He is a member of IEEE (U.S.A) and IPSJ (Japan). He received the Telecom. System Technology Award from Telecommunications Advanced Foundation in 1992.