PAPER *IEICE/IEEE Joint Special Issue on Assurance Systems and Networks*

A Hierarchical Routing Protocol Based on Autonomous Clustering in Ad Hoc Networks*

Tomoyuki OHTA†a), ***Regular Member***, **Munehiko FUJIMOTO**†, ***Student Member***, **Shinji INOUE**†, ***and*** **Yoshiaki KAKUDA**†, ***Regular Members***

SUMMARY Recently, in wired networks, a hierarchical structure has been introduced to improve management and routing. In ad hoc networks, we introduce a hierarchical structure to achieve the same goal. However, it is difficult to introduce the hierarchical structure because all mobile hosts are always moving around the network. So, we proposed the clustering scheme to construct the hierarchical structure before. In this paper, we propose a new hierarchical routing protocol called Hi-TORA based on the clustering scheme. And we show the experimental evaluation of Hi-TORA with respect to the number of control packets, accuracy of packet delivery and hop counts in comparison with TORA.

key words: *ad hoc networks, hierarchical routing, clustering, autonomous decentralized systems, mobile computing*

1. Introduction

An ad hoc network is a wireless mobile network in which a collection of mobile hosts and wireless links connecting them forms a temporary network without the aid of any base stations and mobile switching centers. The ad hoc network can be applied for various purposes such as civilian use, disaster recovery and military use. Originally, it was used for military use such that, in a battlefield, each soldier communicates to remote soldiers via data forwarding from one mobile device installed in mobile vehicles, tanks and trucks to another. It has been developed to communicate at the place where base stations can not be installed. Recently, it can be applied for car-to-car mobile communication in a desert and boat-to-boat mobile communication on a sea. Along with recent development of the ad hoc networks, we expect that their size will become large. It is thus required to assure the management of the large ad hoc networks efficiently.

Many routing protocols have been proposed for the ad hoc networks. They are classified into plane routing protocols and hierarchical routing protocols.

The plane routing protocols are composed of table driven (proactive) and on-demand (reactive) protocols. In the table driven protocols [14], each mobile host maintains a routing table where the next node into which packets are to be delivered is stored and updates it periodically. However, since mobile hosts move around in the network, it is difficult to adapt the routing table to frequent change of the network topology. As the size of the networks becomes larger, the protocols have higher overhead to maintain the routing table.

In the on-demand protocols [7], [12], [13], a route to a destination is required only when there is a data packet to deliver to the destination. In principle, each source broadcasts route request packets in the network to find the route. When the destination receives the request packet, it sends back the route reply packet containing the complete route from the source to the destination along the reverse route to the source. When the route is disconnected due to movement of mobile hosts, the route discovery may be locally performed.

On the other hand, in the hierarchical routing protocols [6], [8], [10], [11], the network is divided into multiple clusters and a route from a source to a destination is constituted with subroutes within clusters and subroutes among clusters. The hierarchical routing protocols are designed for the purpose of efficiency and robustness of routing. For example, the routing protocol proposed in [6] combines the table driven and on-demand routing protocols in such a way that they are applied to routing within clusters and among clusters, respectively. Since each mobile host within the cluster maintains a routing table by which the route to any destination in the cluster can be found, data packets generated in a mobile host are reachable to any mobile host within another cluster if routes among clusters are discovered on demand.

In this paper, we propose a new hierarchical routing protocol called Hi-TORA. The routing within clusters and between clusters apply link state protocol and on-demand protocol called Temporally-Ordered Routing Algorithm (TORA) [12], respectively. Since TORA forgoes propagation of link-state or distance information, TORA is able to localize its reaction to topological changes. As the effect of this localization, scalability of TORA is greatly increased. And Hi-TORA

Manuscript received March 20, 2003.
Manuscript revised May 19, 2003.
†The authors are with the Faculty of Information Sciences, Hiroshima City University, Hiroshima-shi, 731-3194 Japan.
a) E-mail: ohta@ce.hiroshima-cu.ac.jp
*This work was supported in part by the Ministry of Education, Science, Sports and Culture of Japan under Grant-in-Aid for Scientific Research (C) (No.15500049), Telecommunications Advanced Foundation and Hiroshima City University under their research grants.

utilizes the clustering scheme proposed in [5]. In [1], [2], we showed that the clustering scheme has high adaptability to node movement. The size of all clusters in the network is almost the same by using the clustering scheme. Therefore, we expect that the routing protocol is more effective and robust. In this paper, we call a mobile host a node from now on.

The rest of the paper is organized as follows. Section 2 addresses the clustering scheme we proposed in [5]. Section 3 describes TORA briefly. Section 4 describes the proposed hierarchical routing protocol Hi-TORA in detail. The evaluation results of Hi-TORA are shown by simulation experiments in Sect. 5. We discuss some related work in Sect. 6. Finally, Sect. 7 concludes this paper with future research.

2. Clustering Scheme

2.1 Preliminary

Let the ad hoc network be denoted by an undirected graph $G = (V, E)$. A node $v_i \in V$ represents a node with node ID i and it is simply called node v_i afterwards. An edge $(v_i, v_j) \in E$ represents a wireless link between nodes i, j and then node v_j is called the neighboring node of node v_i.

A cluster C_i is a subset of V. C_i consists of clusterhead v_i and some clustermembers. We define a node which manages and belongs to a cluster as a *clusterhead* and a node which has a cluster ID as a clustermember. A clusterhead manages the number of clustermembers in the cluster. A cluster satisfies two following conditions. One is that all clustermembers must be connected with the clustermember through other clustermembers which have identical cluster ID. The other is that the number of clustermembers ($|C_i|$) which the clusterhead has is bounded by the two constants, the upper bound (U) and lower bound (L). If the number of clustermembers which a clusterhead has is less than L, the clusterhead tries to merge with one of neighboring clusters. On the contrary, if the number of clustermembers which a clusterhead has is greater than U, the clusterhead tries to divide the cluster which the clusterhead manages into two clusters.

The cluster division and merger of the clustering scheme were described in [5] in detail and have been evaluated by theoretical considerations in [4].

Finally, we define a node which has a neighboring node with different cluster ID's as a *gateway*. A clusterhead can communicate with the neighboring cluster through the gateways.

In addition to cluster ID, a node holds a state representing the role of the node, that is, clusterhead, gateway and clustermember. The cluster ID and the state which each node holds are changed according to node movement in the ad hoc network.

2.2 Node Actions and Control Packets for Clustering

In order to maintain the hierarchical structure, each node (v_i) performs the following two actions.

Action 1: A node exchanges the information with the neighboring nodes periodically by using *hello packet*.

Action 2: If a node is a clustermeber, the node broadcasts it's information to all clustermembers in the some cluster periodically by using control packets called *periodic notification packet*.

By exchanging hello packets between node v_i and its neighboring nodes, node v_i checks the cluster ID's and states the neighboring nodes hold. And by sending and receiving periodic notification packets, node v_i can know information on all nodes in the cluster to which node v_i belongs. After that, the cluster ID and state node v_i holds are changed if the conditions described in Sect. 2.5 are satisfied.

We will define states representing the role of several types of nodes in the following subsection.

2.3 States and Actions

First, we provide the definitions of states of each node and explain the actions which each node performs in a state.

Five types of states are defined as follows. Node v_i holds one of five states and performs the following actions in relation to the state.

NSN(Normal State Node) Node v_i with state NSN is a clustermember which is neither any clusterhead nor any gateway. Node v_i periodically broadcasts control packets including node ID i and cluster ID j ($j \neq i$). v_i holds all nodes' IDs in the cluster which node i currently recognizes. When nodes with cluster ID j receive such control packets, they recognize that node v_i belongs to the cluster with cluster ID j and update information the nodes have.

CN(Control Node) Node v_i with state CN plays the role of a clusterhead. Node v_i periodically broadcasts control packets including node ID i and cluster ID j ($j = i$). v_i holds all nodes' IDs in the cluster which node v_i currently recognizes. When nodes with cluster ID i receive such control packets, they recognize that node v_i is currently a clusterhead in the cluster with cluster ID i and update information they have.

BN(Border Node) Node v_i with state BN plays the role of a gateway. Node v_i periodically broadcasts control packets in the same way as shown in NSN. Suppose that node v_i belongs to an cluster with cluster ID j ($j \neq i$). For a broadcast, information on a neighboring node of node v_i in the cluster with cluster ID k ($k \neq j$) is additionally included in control packets. This information is obtained by receiving a hello packet from the neighboring node of node v_i.

BCN(Border and Control Node) Node v_i with state BCN plays the roles of both a clusterhead and a gateway. A broadcast is performed in the same way as shown in CN (Control Node) and BN (Border Node).

ON(Orphan Node) Node v_i with state ON does not have any cluster ID. Note that, when a node is newly added, a state of the node becomes ON.

2.4 Assignment and Update of Cluster ID

Let NN_i denote a set of neighboring nodes of node v_i. Node v_i recognizes this set by receiving hello packets from them. Cluster IDs of the nodes in NN_i are obtained by receiving hello packets from them. Based on such cluster IDs, the cluster ID to be held by node v_i is assigned or updated in the following. Let v_i's cluster ID and v_i's state be denoted by $Cluster(v_i)$ and $State(v_i)$, respectively.

[Definition:] Let NN_i^k denote a set of nodes such that they belong to NN_i and the cluster ID held by them is k, that is, let $NN_i^k = \{v_j | v_j \in NN_i \wedge k = Cluster(v_j)\}$. Let $NN_i^{\bar{k}}$ denote a set of nodes such that they belong to NN_i and the cluster ID held by them is not k, that is, let $NN_i^{\bar{k}} = \{v_j | v_j \in NN_i \wedge k \neq Cluster(v_j)\}$.

The cluster ID to be held by node v_i is assigned or updated as follows.

Case 1: $State(v_i) = ON$ and when v_i received a hello packet from an ON node, v_i recognizes that all neighboring nodes have been also orphan nodes (that is, $NN_i = \{v_j | v_j \in NN_i \wedge State(v_j) = ON\}$), v_i assigns the cluster ID to i (that is, lets $Cluster(v_i) = i$) with some probability.

Case 2: $State(v_i) = ON$ and when v_i received a hello packet from a neighboring node with its cluster ID $= k$, v_i recognizes a neighboring node holding cluster ID k, (that is, there is a node in NN_i^k), v_i assigns the cluster ID to k, (that is, lets $Cluster(v_i) = k$).

Case 3: There is a node v_i such that the cluster ID held by v_i is k and v_i has only a neighboring node holding cluster ID k, (that is, there was a node in NN_i^k). When v_i received a hello packet from the neighboring node v_t and the packet indicates $Cluster(v_t) = m$, v_i recognizes that all the neighboring nodes holding cluster ID k have disappeared, (that is, $NN_i = NN_i^{\bar{k}}$), v_i updates the cluster ID from k to m such that $m \neq k$ (that is, lets $Cluster(v_i) = m$).

Note that when nodes are newly added into the network, they become orphan nodes which do not have any cluster ID. In Case 1 or Case 2, the orphan node can have some cluster ID. In order to avoid that all orphan nodes have different cluster IDs and form a set of clusters in which there is only one node, in Case 2 some probability that cluster ID is assigned to orphan nodes is given.

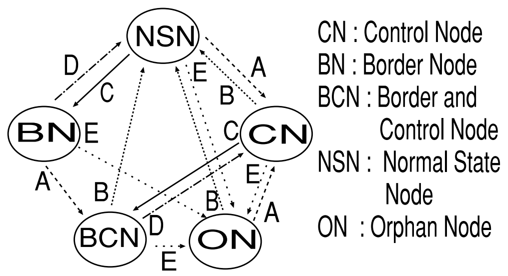

Fig. 1 State transition diagram in node v_i.

2.5 State Transitions in Nodes

Let NN_i denote a set of neighboring nodes of node v_i. Node v_i recognizes this set by receiving hello packets from them. Cluster IDs held by the nodes in NN_i are obtained by receiving hello packets from them. Based on such cluster IDs held by the nodes in NN_i, the states to be held by node v_i is changed in the following. This state change is called state transition.

Node v_i executes state transitions A–E as shown in Fig. 1 if the conditions for them described below are satisfied. Let $Cluster_{-T}(v_i)$ denote a cluster ID held by node v_i just before receiving hello packets from the neighboring nodes and updating the cluster ID. Let $Cluster(v_i)$ denote a current cluster ID held by node v_i just after receiving hello packets from the neighboring nodes and updating the cluster ID.

Condition for Transition A:

$Cluster_{-T}(v_i) \neq Cluster(v_i) \wedge i = Cluster(v_i)$: This condition represents that the cluster ID is updated and the updated cluster ID equals to the node ID held by node v_i.

Condition for Transition B:

$Cluster_{-T}(v_i) \neq Cluster(v_i) \wedge i \neq Cluster(v_i)$: This condition represents that the cluster ID is updated and the updated cluster ID does not equal to the node ID held by node v_i.

Condition for Transition C:

$NN_i \supset NN_i^{\bar{k}}$: This condition represents that a neighboring node appeared, which holds a cluster ID being different with cluster ID held by node v_i.

Condition for Transition D:

$NN_i = NN_i^k$: This condition represents that the cluster ID's held by all neighboring nodes became the same as the cluster ID held by node v_i.

Condition for Transition E:

$NN_i = \phi$: This condition represents that all neighboring nodes of node v_i disappeared.

3. TORA [12]

Since Hi-TORA is based on Temporally-Ordered Routing Algorithm (TORA), we now present the outline of TORA. However, due to lack of space, it is not possible to illustrate the details of TORA. The readers are referred to [12] for further explanations. TORA is one of a family of *link reversal* algorithm for routing in ad hoc networks. TORA maintains a destination-oriented directed acyclic graph (DAG) for each possible destination node. In this graph structure, any node leads to the destination node by following in logical direction which links have. TORA uses the notation of *height* to determine the direction of each link. When a node tries to communicate with another node, all nodes in the network make use of height. Height of the source node is the largest value and height of the destination node is the smallest value. The logical links are considered to be directed from nodes with higher height towards nodes with lower height. Despite dynamic link failures, TORA attempts to maintain the DAG such that each node can reach the destination node.

TORA performs three basic functions, that is, route creation, route maintenance, and route erasure and three control packets are used by each function, that is, query (QRY), update (UPD), and clear (CLR).

We describe the description of height. The height is defined as a five-tuple, $H_i = (\tau, oid, r, \delta, i)$. The height consists of two components:a *reference level* represented by the first three elements of the five-tuple, and a *delta* with respect to the reference level, represented by the last two elements of the five tuple. Each element of the five-tuple is explained below.

τ**:** A new reference level is defined each time a node loses its last outgoing link. τ represents the time of the link failure.
oid**:** Unique identifier of the node that defined the new reference level.
r**:** Reflection indicator bit.
δ**:** Propagation ordering parameter.
i**:** Unique node identifier(ID).

We present route creation phase in TORA. QRY and UPD packets are used for the route creation. We explain using the network as shown in Fig. 2. In Fig. 2, the double circle denotes a node such that *route-required* flag is set, and the arrow denotes a downstream logical link. Here, the route-requried flag is used for managing whether the node had already received QRY packet from any neighbors or not.

Initially, as shown in Fig. 2(a), height H_i of each node i except for the destination node F is set to $NULL$. $NULL$ is defined as $H_i = (-,-,-,-,i)$. The destination node F sets its height to be $ZERO = (0,0,0,0,F)$. Now, when node A requires a route to the destination node F because it has no outgoing links,

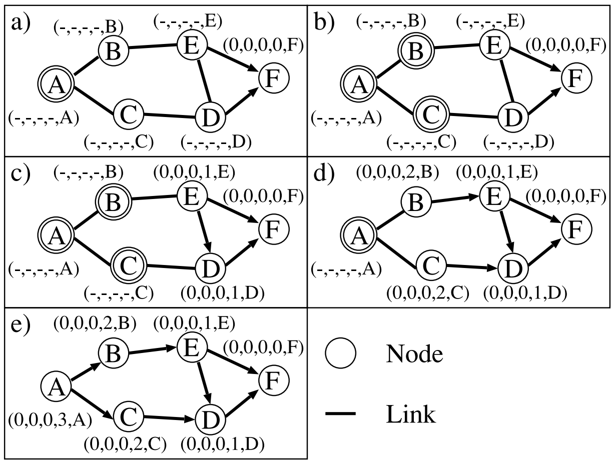

Fig. 2 Examples of route creation phase in TORA.

it broadcasts QRY packet to all of its neighbors and sets a route-required flag. As shown in Fig. 2(b), if the node received QRY packet has no downstream links and its route-required flag is un-set, then it just forwards the QRY to neighbors and sets the route-required flag link node B and C. When a node (for example, node E and D in Fig. 2(c)) which has downstream logical links receives QRY packet, it modifies value of δ in its height, based on the relative height metric of neighboring nodes. Thus, the node changes its current $NULL$ height $(-,-,-,-,i)$ to $(\tau, oid, r, \delta+1, i)$, where $(\tau, oid, r, \delta, i)$ is the minimum height of its non-$NULL$ neighbors. And it broadcasts UPD packet containing the new height to all of its neighbors. A node which received the UPD packet changes its current height to the height and make downstream logical links as shown in Fig. 2(d). In turn, node A updates its height as $(0,0,0,3,A)$ since its non-$NULL$ neighbors' minimum height is $(0,0,0,2,B)$ when it receives an UPD from node B and make downstream logical links from A to B and C. TORA establish DAG as shown in Fig. 2(e). Every node just forwards data packets to the destination node along with logical links in the DAG.

4. Proposed Hi-TORA Protocol

This section describes the proposed Hi-TORA for hierarchical routing in ad hoc networks. Hi-TORA is the protocol based on TORA and the clustering scheme mentioned in Sect. 2. Hi-TORA applies TORA to the routing among clusters and the protocol such as link-state type to the routing on the cluster. In the routing among clusters, by regarding one cluster as one node for TORA, Hi-TORA can route data packets from the source node to the destination node to communicate with each other.

4.1 Routing in Cluster

The routing table based on the hop number in each cluster is determined. The routing table is configured by periodic notification packets which are used for clustering scheme mentioned in Sect. 2.

The routing in the cluster is divided into two cases. Using clusters A, B, and C in the network we explain the two cases. We focus on the routing in cluster B. Now, suppose that cluster B neighbors clusters A and C, and clusters A and C neighbor only cluster B. One case is that the data packets sent from cluster A to C pass through cluster B. Then, in cluster B, a gateway receives the packets from the gateway of cluster A. The gateway forwards the data packets toward another gateway which is neighboring to cluster C according to the routing table of cluster B. And, the gateway which received packets sends them to the gateway of cluster C. The other case is that the destination node of data packets sent from cluster A or C exists in cluster B. Then, in cluster B, a gateway receives the data packets from the gateway of cluster A or C. The gateway forwards the data packets toward the destination node according to the routing table of cluster B.

4.2 Routing among Clusters

Hi-TORA applies TORA for the routing among clusters because TORA has high adaptability to node movement in the network. We describe how to communicate between a source node and a destination node. When a source node wants to communicate with a destination node, the source node sends QRY packets toward the destination node. In TORA, a height is set at all nodes which received the packets. Then, the height of the source node is the highest value in the network, and the height of the destination node is the lowest one. However, in Hi-TORA, a height is set at all clusters as shown in Fig. 3. That is, the height isn't set at the destination node, but is set at the clusterhead to which the destination node belongs as shown in Fig. 4. Then, the height of the source cluster to which the source node belongs is the highest value in the network, and the height of the destination cluster to which the destination node belongs is the lowest one. The data packets generated by the source node are forwarded toward the cluster with the lower height like TORA.

4.3 Problems and Their Solutions

Hi-TORA to which TORA is applied in a straightforward way has some problems because TORA isn't designed to use for a hierarchical routing. We can consider the following three cases occur.

The first case is that the source node runs away from the source cluster to which it belongs. Then, it doesn't affect anything in special in Hi-TORA. Hi-TORA takes over the idea of DAG from TORA. It is possible for every node to send the data packets to the destination node according to the logical link which the node has in the DAG. Therefore, if the source node moves to another cluster, the source cluster to which it belongs is only changed.

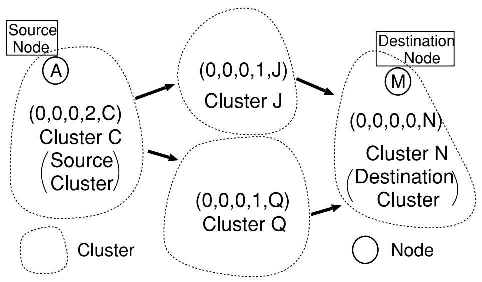

Fig. 3 Concept of the routing in Hi-TORA.

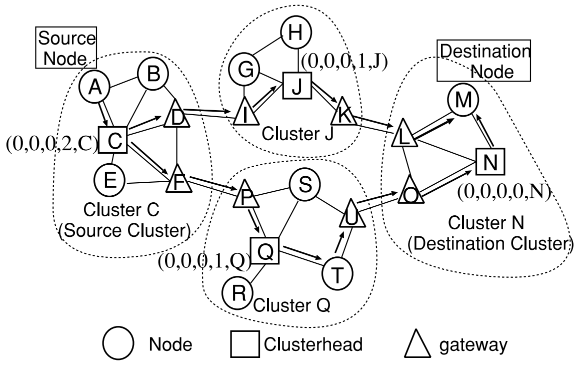

Fig. 4 Example of the routing in Hi-TORA.

The second case is that the destination node runs away from the destination cluster to which it belongs. The cluster which it enters is redefined as a new destination cluster. In this case firstly, the clusterhead in the new destination cluster sets its height to $ZERO$. Then, like UPD packets of TORA it broadcasts the new height to the neighboring clusters to find a new route to the destination node in the new destination cluster.

The last case is that the cluster ID of an intermediate cluster is changed. We explain using the network as shown in Fig. 5(a). Here, a circle denotes a cluster, a line denotes a link and an arrow denotes a logical link between two clusters. Hi-TORA establishes the DAG in the network so that every node can send data packets to the destination node E according to the direction of the logical link. However, since nodes are always moving in the network, there is some possibility that a clusterhead runs away from the cluster to which it belongs so that the change of a cluster ID often occurs. Suppose that the clusterhead in cluster D runs away from the cluster. Then, the cluster D changes the ID to F, based on the clustering algorithm (Fig. 5(b)). The updated cluster sets its height to $NULL$. And according

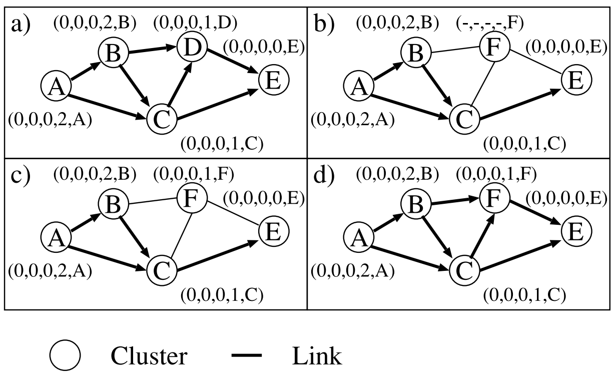

Fig. 5 Change of the cluster ID of intermediate cluster in Hi-TORA.

to the height calculation of TORA, the cluster needs to change the height from $NULL$ to $(\tau, oid, r, \delta + 1, i)$, where $(\tau, oid, r, \delta, i)$ is minimum height of all neighboring clusters. In this example, since the heights of all neighboring clusters are $(0, 0, 0, 2, B)$, $(0, 0, 0, 1, C)$ and $(0, 0, 0, 0, E)$, the minimum height of all neighboring clusters is $(0, 0, 0, 0, E)$. According to the height calculation of TORA, cluster F changes the height from $NULL$ to $(0, 0, 0, 1, F)$ (Fig. 5(c)). Finally, cluster F has logical links between all neighboring clusters as shown in Fig. 5(d). Even if the clusterhead runs away from the cluster, data packets can be delivered to the destination node by reorganizing the DAG.

5. Simulation Experiments

For the evaluation purpose, the proposed Hi-TORA is compared to TORA through simulation experiments. This section describes the simulation model and parameters, and then, the performance of Hi-TORA in terms of the number of control packets, the accuracy of packet delivery and the hop count of data packets.

5.1 Simulation Model and Parameters

We used the random waypoint model [9]. We explain the random waypoint model. It performs the following actions repeatedly. A node moves toward a given position at a specified speed on average, and then, the node stops for a specified time at the position. We define the time as the pause time. The pause time is set 0 in the experiments. And as soon as the node arrives at the position, the node selects a new destined position and starts moving to the position again.

Our simulation is based on an ad hoc network of 200 mobile nodes, where initial locations are chosen from a uniform random distributed over a square of 750 m × 750 m. The considered average moving speeds are 0 m/s, 1 m/s, 5 m/s, 10 m/s, and 20 m/s. Here, moving speed of each node is given during ±20%.

Each node has a radio transmission range of radius 100 m. At any time, each node can communicate with all the nodes within its transmission range. Propagation delay through each link including both data packet transmission time and processing time is 1 ms. And also, in the simulation the bandwidth and packet loss between any two nodes are not taken into consideration.

Each node periodically sends hello packets to nodes within its transmission range for a period of 200 ms to exchange its state and node ID. And each node broadcasts periodic notification packets to inform all nodes within the cluster to which the node belongs of the information which the node has. The duration of each simulation is 65 seconds. No data is collected for the first 5 seconds to avoid measurements during the transient period and to ensure an initial hierarchical structure which is generated by the clustering scheme.

The simulation scenario proceeds as follows. First, one SD pair is selected by choosing a source node and a destination node in the network. At 5 seconds, the source node starts to generate the route to the destination node. After the route is generated, the source node starts to send data packets to the destination node for a period of 200 ms until 65 seconds. To maintain the route, TORA requires control packets for the routing and Hi-TORA needs control packets for the routing and the clustering. In the first experiment, we measure the control packets to compare the performance of TORA and Hi-TORA.

The source node can send 300 data packets at maximum from 5 seconds to 65 seconds. In the second experiment, we measure the number of data packets which the destination node received to evaluate the accuracy of packet delivery. And then, in the third experiment, we measure the hop count of data packets by analyzing them. Through these three experiments, we show the effectiveness of Hi-TORA.

Finally, in Hi-TORA the ranges of the cluster size (L, U) are $(10, 25)$, $(20, 50)$ and $(30, 75)$, respectively. The simulation experiments were performed at 5 times for combination of each parameter.

5.2 Results for Control Packets

Table 1 shows the average number of control packets for routing of TORA versus moving speed of nodes. Table 2 shows the average number of control packets for routing and clustering of Hi-TORA versus moving

Table 1 Average number of control packets of TORA.

V (m/s)	A
0	3419.60
1	20564.20
5	108421.60
10	194460.20
20	373946.20

V: moving speed of node
A: average number of control packets

Table 2 Average number of control packets of Hi-TORA.

	Cluster Range					
	10-25		20-50		30-75	
V (m/s)	B	C	B	C	B	C
0	72.20	234223.40	34.20	204655.80	18.00	215148.40
1	224.00	243414.60	56.40	211626.60	24.60	211571.60
5	1932.40	323456.00	363.40	252834.20	84.60	230652.60
10	6518.00	380812.00	1251.80	268497.80	907.00	235619.00
20	9614.80	434067.00	2559.00	281446.00	1465.20	257470.80

V: moving speed of node B: average number of control packets for routing
C: average number of control packets for clustering

speed of a node and the cluster size. In both tables, each number shows the average number of control packets which either TORA or Hi-TORA requires to maintain the route for one SD pair in the network during 60 seconds specified above.

As shown in Tables 1 and 2, when we focus on control packets only for routing, control packets of Hi-TORA are much smaller than those of TORA because of hierarchy. However, Hi-TORA uses lots of control packets for clustering. In fact, because there are lots of SD pairs in the network, we expect if the larger the number of SD pairs increase, the more control packets for routing are becoming greater. However, control packets for clustering do not depend on the number of SD pairs. To make the boundary clear, we calculate the minimum number of SD pairs which shows that Hi-TORA is more effective than TORA.

We denote the number of SD pairs, the number of control packets for routing of TORA, the number of control packets for routing of Hi-TORA as N, T_p, HIT_p. Then, the numbers of control packets for routing of TORA and Hi-TORA require $N \times T_p$ and $N \times HIT_p$, respectively. And we denote the number of control packets for clustering of Hi-TORA as HI_p. Therefore, the total number of control packets of Hi-TORA is $N \times HIT_p + HI_p$. Finally, we denote the minimum number of SD pairs to make Hi-TORA effective as N_{min}.

The relation between the number of control packets of TORA and Hi-TORA to obtain N_{min} is shown by Eq. (1).

$$N \times T_p > N \times HIT_p + HI_p. \tag{1}$$

And then,

$$N > \frac{HI_p}{T_p - HIT_p} \tag{2}$$

is derived. Therefore, we can hold Eq. (3) to minimize N.

$$N_{min} = \left\lfloor \frac{HI_p}{T_p - HIT_p} \right\rfloor + 1 \tag{3}$$

N_{min} was calculated by utilizing the mean number of control packets of Tables 1 and 2 and each moving speed of nodes. The result is shown in Table 3. As shown in Table 3, whenever nodes are moving, Hi-TORA is more effective than TORA in case that there are only a few SD pairs. It is because the number of control packets for clustering dominates overhead in Hi-TORA and does not increase proportionally to moving speed of nodes, while it costs much overhead for each SD pair to generate a route and maintain the route in TORA and the number of control packets increases almost proportionally to moving speed of nodes.

Table 3 Minimum number of SD pairs in which Hi-TORA is effective in the network.

	Cluster Range		
V(m/s)	10-25	20-50	30-75
0	70	61	64
1	12	11	11
5	4	3	3
10	3	2	2
20	2	1	1

V: moving speed of node

We expect that the number of SD pairs increases as the size of a network becomes large. The number of control packets for routing of Hi-TORA is becoming smaller than that of TORA along with the increase of the number of SD pairs. Therefore, Hi-TORA is more effective than TORA as the size of a network is larger. Consequently, we can say that Hi-TORA is even more effective than TORA in case that nodes move at faster speeds and the size of a network becomes large.

In Table 2, the number of control packets decreases along with the increase of the upper bound U and the lower bound L of a cluster size. This fact is mainly caused by division and merger of clusters. In addition to control packets used for routing, Hi-TORA requires control packets for division and merger of clusters. In case that U and L are small, there are high possibilities that clusters are reconfigured frequently due to frequent division and merger of clusters. On the contrary, in case that U and L become large, the number of divisions and merger of clusters decreases because the number of clusters in the network decreases and also the number of control packets for division and merger of clusters decreases. However, the more the size of clusters increases, the more the number of control packets for division and merger of clusters gradually decreases and the more the number of control packets which nodes broadcast within their cluster increases in a square or-

Table 4 Number of data packets which destination node received from source node.

		Hi-TORA		
		Cluster Range		
V(m/s)	TORA	10-25	20-50	30-75
0	299.80	299.00	299.00	299.20
1	172.80	286.80	298.00	298.00
5	141.80	229.40	283.00	263.80
10	180.40	181.00	193.00	274.80
20	77.20	132.20	171.20	261.60

V: moving speed of node

Table 5 Number of hop counts of data packets.

		Hi-TORA		
		Cluster Range		
V(m/s)	TORA	10-25	20-50	30-75
0	8.00	9.40	10.40	10.60
1	7.66	10.92	9.42	9.97
5	7.87	9.55	9.35	9.14
10	5.74	7.08	6.66	6.37
20	3.63	4.72	4.97	5.40

V: moving speed of node

der of the number of nodes within each cluster. We expect the number of control packets in the extreme case that only one cluster covers the entire network is $12000000 = |V|^2 \times 300$ where $|V| = 200$. This increasing ratio of the number of control packets only for broadcast dominates in the increasing ratio of the total number of control packets when U and L exceed a threshold of them. This threshold depends on complex combination of several parameters concerning network environments.

5.3 Results for Accuracy of Data Packet Delivery

Table 4 shows the number of data packets which a destination node received from a source node to evaluate the accuracy of packet delivery. The result shows that the number of data packets decreases as moving speed of nodes is much faster and the range of cluster size is larger in both TORA and Hi-TORA. However, the decreasing ratio for Hi-TORA is smaller than that for TORA even when the range of cluster size is larger.

It is because when the range of cluster size is larger, there is high possibility that both a source node and a destination node belong to the same cluster. The routing within each cluster has high connectivity in comparison with the routing among clusters. Especially, in case that the range of cluster size is $(30, 75)$, the number of data packets which the destination node could receive still remain in almost the same high level even when moving speed of nodes is fast.

We can definitely conclude that Hi-TORA is even more effective than TORA with respect to the accuracy of data packet delivery.

5.4 Results for Hop Count

Table 5 shows the hop count of data packets. Regarding hop counts, Hi-TORA is not superior to TORA for the following reason. In Hi-TORA, each cluster is autonomously constructed and maintained so that data packets are delivered within the cluster along a shortest path between gateways. However, connection of such shortest paths in neighboring clusters does not always guarantee the shortestness between a gateway in a cluster and another gateway in its neighboring cluster.

We use the example as shown in Fig. 6 to explain

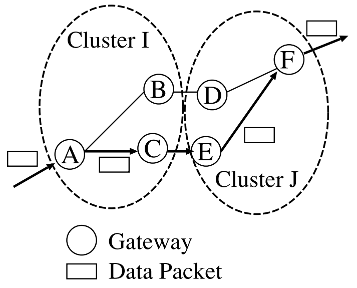

Fig. 6 Problems of hierarchical routing.

the above reason. Now, nodes $A - F$ are gateways. Suppose that node A received a data packet from a gateway in a neighboring cluster. Node A forwards the data packet toward the neighboring cluster according to the routing among clusters because the destined node of the data packet dose not exist in cluster I. Now, node A decides to forward it to node C because the distance between nodes A and C is shorter than the distance between nodes A and B. After that, the data packet is forwarded in the order of nodes C, E, and F. And then, node F forwards it to the neighboring cluster. However, there is some possibility that it is forwarded in the order of nodes A, B, D, and F, which is superior to the case that it is forwarded in the order of nodes A, C, E and F with respect to total hop counts in clusters I and J.

In ad hoc networks, accuracy of data packets delivery is more important than delay of data packets delivery. Since accuracy of data packets in Hi-TORA is even higher than that in TORA specially in high mobility cases, and the ratio of hop counts in Hi-TORA to that in TORA is less than 1.5 in all cases, we can say that the hop counts in Hi-TORA is in an acceptable level.

Next, as moving speed of nodes is faster, the hop count of data packets becomes shorter. This is because there is high possibility that the source node and the destination node become to be in the neighborhood due to frequent node movement for a long time. In the experimental environment, when a source node and a destination node on both ends of the field communicate with each other, it takes 8 hops from the source node to the destination node even in the most cases. Because, when moving speeds of nodes are 0 m/s and

5 m/s, nodes do not move much, there is high possibility that the distance between the source node and the destination node still remain to be far and therefore, data packets require a lot of hops to be delivered from the source node to the destination node.

6. Related Work

DSDV [14], DSR [13] and ZRP [6] are a kind of table driven protocols, on-demand protocols and hierarchical routing protocols, respectively. They provide only a single route between a source node and a destination node. In ad hoc networks, connectivity is the most important factor. We expect that Hi-TORA and TORA [12] have better performance than DSDV, DSR and ZRP with respect to the connectivity because Hi-TORA and TORA provide multiple routes between a source node and a destination node.

Some clustering schemes applied to the hierarchical routing have been proposed. In LCC scheme [10], a cluster consists of one clusterhead and its neighboring nodes called clustermembers. During node movement, each node recognizes the cluster to which the node belongs and the role of the node in the cluster as follows. When a node finds that its neighboring node is a clusterhead, the node belongs to the cluster which the clusterhead manages. When a node finds that more than one neighboring node of the node are clusterheads, it becomes a gateway shared with the clusters which these clusterheads manage. When a clusterhead finds that its neighboring node is another clusterhead, one of these clusterheads (that is, a clusterhead with the largest IP address) maintains the role of clusterhead and the others abandon their roles.

In the experimental evaluation, we compare our clustering scheme used in Hi-TORA with the LCC scheme with respect to the number of changes of clusterhead and cluster ID in [1]. The number of changes of clusterhead is an important factor to evaluate the clustering schemes in ad hoc networks. If the number is small, it means that the cluster is not frequently reconfigured. On the other hand, if the number is large, it means that the cluster is frequently reconfigured. In [1], the number of changes of clusterhead and cluster ID of our proposed scheme is much smaller than that of the LCC scheme. In case that the number of changes of clusterhead and cluster ID is large, it means that the clustering is not stable. In the hierarchical routing, the route between a source node to a destination node is made based on clusterheads and cluster IDs. If the clustering is not stable, we expect that the route between the source node and the destination node is changed frequently and a lot of redundant control packets are required to maintain the route and re-create another route. The other clustering schemes [8], [11] utilize hop counts from clusterheads. The similar discussions hold for hierarchical routing based on these clustering schemes.

As a result, we can say that Hi-TORA based on our clustering scheme is much effective in ad hoc networks.

7. Conclusion

We have proposed a hierarchical routing called Hi-TORA in ad hoc networks. As simulation results show, Hi-TORA can significantly reduce control packets as compared to TORA with increasing of the number of SD pairs because of adaptive hierarchy of the clustering scheme in [4]. In the future work, we would like to evaluate the proposed Hi-TORA from the various points of view and implement it in the real network.

References

[1] T. Ohta, S. Inoue, Y. Kakuda, and K. Ishida, "An adaptive multihop clustering scheme for ad hoc networks with high mobility," IEICE Trans. Fundamentals, vol.E86-A, no.7, pp.1689–1697, July 2003.
[2] T. Ohta, S. Inoue, and Y. Kakuda, "An adaptive multihop clustering scheme for highly mobile ad hoc networks," Proc. 7th IEEE Int. Symp. Autonomous Decentralized Systems (ISADS2003), pp.293–300, 2003.
[3] T. Ohta, M. Fujimoto, S. Inoue, and Y. Kakuda, "Hi-TORA: Hierarchical routing protocol in ad hoc networks," Proc. 7th IEEE Int. Symp. High Assurance Systems Engineering (HASE2002), pp.143–148, 2002.
[4] T. Ohta, S. Inoue, Y. Kakuda, K. Ishida, and K. Maeda, "An adaptive maintenance of hierarchical structure in ad hoc networks and its evaluation," Proc. 1st Int. Workshop on Assurance in Distributed Systems and Networks (ADSN2002), pp.7–13, 2002.
[5] T. Ohta, K. Ishida, Y. Kakuda, S. Inoue, and K. Maeda, "Maintenance algorithm for hierarchical structure in large ad hoc networks," Proc. Int. Conf. Fundamentals of Electronics, Communications and Computer Sciences (ICFS2002), pp.S3-12–S3-17, 2002.
[6] C.E. Perkins, Ad hoc networking, pp.221–253, Addison-Wesley, 2001.
[7] C.E. Perkins and E.M. Royer, "Ad hoc on-demand distance vector routing," Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications, pp.90–100, 1999.
[8] S. Basagni, "Distributed clustering for ad hoc networks," Proc. Int. Symp. Parallel Architectures, Algorithms, and Networks (I-SPAN'99), pp.310–315, 1999.
[9] 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, pp.85–97, 1998.
[10] C. Chiang, H. Wu, W. Liu, and M. Gerla, "Routing in clustered multihop, mobile wireless networks with fading channel," Proc. IEEE Singapore Int. Conf. Networks (SICON), pp.197–211, 1997.
[11] C.R. Lin and M. Gerla, "Adaptive clustering for wireless networks," IEEE J. Sel. Areas Commun., vol.15, no.7, pp.1265–1275, 1997.
[12] V.D. Park and M.S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," Proc. IEEE INFOCOM'97, pp.1405–1413, 1997.
[13] D. Johnson and D.A. Maltz, "Dynamic source routing in ad hoc wireless network," in Mobile Computing, vol.353, ed. T. Imielinski and H. Korth, pp.153–181, Kluwer Academic

Publishers, 1996.
[14] C.E. Perkins and P. Bhagwat, "Highly dynamic destination sequenced distance-vector routing (DSDV) for mobile computers," Proc. SIGCOMM'94, pp.234–244, 1994.

Tomoyuki Ohta received the B.E., and M.E. degrees from Hiroshima City University, Japan, in 1998 and 2000, respectively. He had been a Ph.D. candidate at the Graduate School of Hiroshima City University from 2000 to 2002. He is currently a research associate in the Department of Computer Engineering, Faculty of Information Sciences, Hiroshima City University, since 2002. His current research interests include mobile communication systems. He is a member of IEEE (U.S.A.).

Munehiko Fujimoto received the B.E. degree from Hiroshima City University, Japan, in 2002. He is currently working toward the M.E. degree at Hiroshima City University. His current research interests include mobile communication systems.

Shinji Inoue received the M.E. degree from Hiroshima University, Japan, in 1988. He has been a research associate in the Department of Computer Engineering, Faculty of Information Sciences, Hiroshima City University since 1994. His research interests include multi-agent systems and fault-tolerant systems. He is a member of IPSJ (Japan).

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 Department of Computer Engineering, Faculty of Information Sciences, Hiroshima City University, since 1998. His current research interests include network software engineering and assurance 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.