A Multipath Routing Method with Dynamic ID for Reduction of Routing Load in Ad Hoc Networks

Tomoya Okazaki, Eitaro Kohno, Tomoyuki Ohta, and Yoshiaki Kakuda

Graduate School of Information Sciences, Hiroshima City University,
3-4-1 Ozuka-Higashi Asaminami-ku, Hiroshima, 731-3194, Japan
{okazaki@nsw.info,kouno@,ohta@,kakuda@}hiroshima-cu.ac.jp

Abstract. In recent years, ad hoc networks have attracted a great deal of attention. Ad hoc networks consist of nodes with wireless communication devices without any base stations or fixed infrastructures. Most routing protocols of ad hoc networks form a single-path. Single-path routing protocols need to repair routes each time the route has broken. This route repair generates a lot of control packets, and an increase in end-to-end packet delay. In order to compensate for these drawbacks of the single path routing, multipath routing schemes have been proposed. AOMDV (Ad hoc On-demand Multipath Distance Vector routing) is one multipath routing scheme. AOMDV constructs routes by flooding Route Request (RREQ) messages. When the number of nodes increases in the network, the routing load of AOMDV, which is defined as the ratio of the number of control packets to the number of delivered packets, may increase immensely. On the other hand, DART (Dynamic Address RouTing) has been proposed for a large scale network. In DART, the dynamic routing address has a tree-based logical structure related to connectivity between adjacent nodes. In this paper, we propose a multipath routing scheme to solve the above problems of AOMDV. The proposed scheme is an extension of DART for dealing with multiple paths. The proposed scheme aims to reduce the routing load and adapt to large ad hoc networks. We evaluated its performance by comparing it with AODV and AOMDV through simulation experiments. Performance metrics are the number of control packets and the routing load. Simulation results indicate the proposed scheme can reduce the number of control packets and the routing load.

Keywords: Ad hoc networks, routing load, Dynamic ID.

1 Introduction

Ad hoc networks are autonomous distributed networks without any base station, since they consist of nodes with wireless communication which relay packets. When the node cannot perform direct communication with other nodes, the intermediate nodes can relay packets of the source node to communicate with the destination node.

J. Zheng, D. Simplot-Ryl, V.C.M. Leung (Eds.): ADHOCNETS 2010, LNICST 49, pp. 114–129, 2010.

AODV (Ad hoc On-demand Distance Vector) [7][6] is a popular routing method for creating single paths. AODV discovers the shortest (single) path between the source and the destination nodes. With a single path routing method such as AODV, when route breaks occur in the network, it is necessary to repair the route in each case. These route repair procedures pose an escalation both in the number of control packets and in the delay to the destination. Multiple path routing [1] is one of the solutions for these problems. AOMDV (Ad hoc On-demand Multipath Distance Vector) [5] routing is one of the methods to create multiple paths. AOMDV is an extension of AODV. AOMDV discovers multiple paths by flooding RREQ (Route Request) messages. When the number of nodes in the network increases, the number of the control packets escalates. Therefore the increase in the number of control packets suppresses the effective bandwidth of the network. Meanwhile, DART (Dynamic Address RouTing) [2] has been proposed as the method which can apply to networks that have many nodes. DART constructs a single path using a dynamic routing address based on assigning addresses to node using a binary tree.

In this paper, we propose a new multiple-path routing method to extend DART to solve the existing problems mentioned above. We have designed the proposed method to reduce the number of control packets in networks with many nodes. Additionally, we have implemented our method on the network simulator, QualNet [8], and have evaluated by comparison with AODV and AOMDV in terms of the number of control packets and the routing load.

The rest of the paper includes the following: In Section 2, we mention the ad hoc networks and their existing problems. We explain our proposed method in Section 3. Section 4 illustrates the experiments and the results. Finally, we conclude our paper in Section 5.

2 Existing Routing Schemes and Their Problems

AODV and DSR (Dynamic Source Routing) [3] are popular single-path routing methods. They are categorized as reactive routing protocols, and the source node broadcasts RREQ (Route Request) messages to discover and establish a route to the destination node. While route information is partially stored on each node in AODV, it is stored only on the source node in DSR.

Meanwhile, AOMDV and SMR (Split Multipath Routing) [4] protocol are proposed to create multiple paths. AOMDV and SMR are extensions of AODV and DSR, respectively. Because these protocols use broadcasting and flooding of control packets, the number of control packets can escalate. As a result, the routing load, that is the proportion of the number of control packets to the number of data packets, will increase. This cause can become serious in large scale ad hoc networks.

On the other hand, DART has been proposed for large scale ad hoc networks. DART introduces the routing address as the indicator of node location, and is utilized for limiting the size of the routing table and the number of control packets. The routing address space has a binary tree structure, which is essential for packet forwarding. Each node stores a unique and static IP address and the dynamic routing address. DART has the serious problem that it is necessary that the start node (the root of the binary tree structure) must be established preliminarily.

3 Proposed Method

3.1 Overview

AOMDV, the existing multipath routing scheme, has the control packet inflation drawback which was mentioned in the previous section. Also, DART has three problems: the need to establish the starting node of the routing address, the overlapping of the routing addresses, and the lack of robustness of the single path. To solve AOMDV and DART, we propose a new multipath routing method. Our method is an extension of DART, and we aim to limit the number of control packets and the routing load for large scale networks. Therefore we extend DART to the following:

1. We introduce Dynamic IDs which are assigned from the source node.
2. The routing table is extended to consist of multiple paths.

Each node stores the following information:

IP address
: The IP address is uniquely assigned at the start. The IP address will not change in our scheme.

Dynamic ID
: Dynamic ID is expressed as the l bit binary numbers $a_{l-1}, a_{l-2}, \cdots, a_0$, where $a_i (i = 0, \cdots, l-1)$ indicate 0 or 1. The binary tree, which is referred to as the ID tree, consists of Dynamic IDs. Figure 1 shows the ID tree with Dynamic IDs made up three bits each. Leaves of the ID tree indicate Dynamic IDs which are assigned to each node. The vertices, which are illustrated as dotted lines, represent the subtree (the set of nodes) below that node. Also, the vertical position of the subtree is referred to as the level. The network topology which corresponds to Figure 1 is depicted in Figure 2. The sets of nodes in the dotted rectangle areas correspond to the subtrees of Figure 1 .

Routing table
: The routing table has routing entries. Each entry has the information of the subtrees and their sets of nodes, and consists of the following:
  - level
  - subtree
  - IP address to the next hop node
  - the hop count to subtree
  - the number of ID tree.

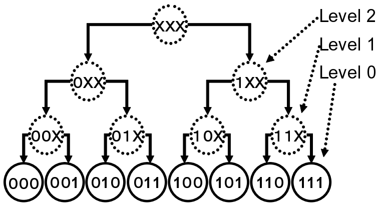


Fig. 1. Example of ID tree with three bit Dynamic ID

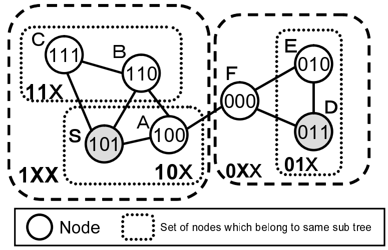


Fig. 2. Example of network topology for proposed method

level	sub tree	ID tree number	next hop	hop count
2	0XX	F	A	2
2	0XX	F	B	2
1	11X	F	C	1
1	11X	F	B	1
1	11X	F	A	2
0	100	F	A	1

Fig. 3. Example of the routing table for proposed method

level	sub tree	next hop	hop count
2	0XX	A	2
1	11X	B	1
0	100	A	1

Fig. 4. Example of the routing table for DART

Figures 3 and 4 show an example of a routing table using the proposed method and DART. While DART has only single next hops to each subtree, our proposed method has multiple next hops to each subtree. This extension is mandatory to make multiple paths.

Number of ID trees

In our scheme, the Dynamic ID assignment starts from the source node which has an unassigned Dynamic ID. Therefore, the network can have multiple ID trees. The ID tree numbers are used to distinguish each ID tree. The ID tree numbers consist of IP addresses of the source and the destination nodes.

3.2 Procedures

```
Procedures of our scheme consist of the following:
  1. Routing
     i. Assignment of the Dynamic ID
     ii. Update of routing table
  2. Packet Forwarding
     i. Node lookup
     ii. Selection of the Next Hop
     iii. Detour using an alternate path
```

Our proposed method differs with DART in that it assigns a Dynamic ID. Additionally, routing and packet forwarding are extended from DART to create multiple paths. Other procedures follow the method established in DART.

The following subsections describe each procedure in detail.

Routing. The routing procedure consists of three sub-procedures: the Dynamic ID assignment, the maintenance of the border node table, and the routing table update. The Dynamic ID assignment sub-procedures are performed using the following three control messages:

– RTUP (RouTe UPdate)
– IDREQ (dynamic ID REQuest)
– IDREP (dynamic ID REPly)
– IDN (dynamic ID Nortification).

RTUP carries the routing information using the routing table in the source node. The routing table has route entries which includes the next hop from the source node. RTUP is used for the Dynamic ID assignment and the update of the routing table sub-procedures. When a node assigns a Dynamic ID, the node broadcasts RTUP to its neighboring nodes. When the demand to communicate with the destination node occurs on the source node, the Dynamic ID assignment sub-procedure is invoked. At the same time, the source node which is not assigned a Dynamic ID becomes the start node of the Dynamic ID assignment sub-procedures and the start node of the ID subtree. The Dynamic ID assignment sub-procedure performs differently depending on the type of assigning node. That is, whether it is the start node or not. Therefore, we explain the Dynamic ID assignment sub-procedure in each case.

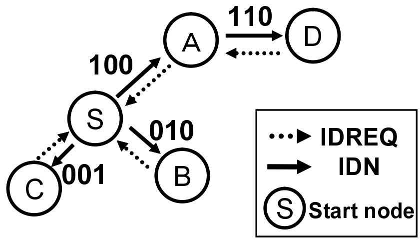
The case in which the assigning node is the start node

Suppose the start node assigns the Dynamic ID to neighboring unassigned nodes. At first, the start node assigns itself to its own Dynamic ID as $[00 \cdots 0]$. Next, the start node broadcasts RTUP to its neighboring nodes. Unassigned nodes receive a RTUP broadcast, then unicast an IDREQ to the start node. When the start node receives IDREQ, the start node selects a Dynamic ID to assign for the IDREQ sender after waiting a certain period of time. At this point, the start node assigns the subtree which has the highest level value within the ID tree of the source node, too. If the level that is assigned to the node which generates IDREQ is m, the Dynamic ID is differs from the start node only at the mth bit. The start node unicasts the IDN message with a new Dynamic ID to the IDREQ generator node.

Figure 5(a) shows an example network. In Figure 5(a), the start node S is going to assign Dynamic ID to nodes A, B, and C. Figures 5(b) and 5(c) show the routing table of node S and the sequence of the procedures, respectively. Suppose node S receives IDREQs in the sequence of C, B, and A. Node S assigns [100], [010], [001] to the Dynamic ID of A, B, and C, which are different in regard to 2nd, 1st, and 0 bit from the Dynamic ID of the node S, respectively.

The case in which the assigning node is any nodes but the start node

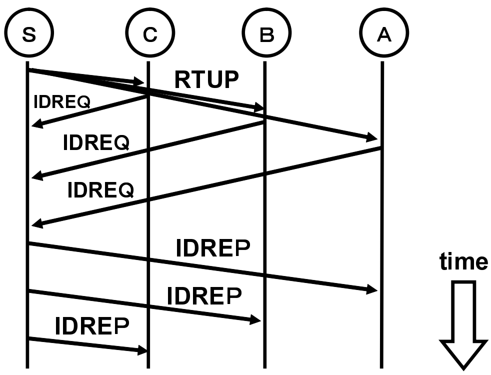
Suppose nodes except the start node assign the Dynamic ID to neighboring unassigned nodes. When the unassigned node receives RTUP from the assigned node, the assignment sub-procedure will be invoked. The unassigned



(a) Dynamic ID assignment.

level	sub tree	next hop
2	1XX	none
1	01X	none
0	001	none

(b) The routing table of node S.



(c) Sequence.

Fig. 5. Example of Dynamic ID assignment by the start node

node sends IDREQ by unicast to the assigned node. The node, which receives IDREQ, searches subtrees which do not have the next hop registered, and selects the highest level subtree. If the level value is m, the node which receives IDREQ assigns a new Dynamic ID for the unassigned to node. To do this, the node which receives IDREQ sends IDN with the newly assigned Dynamic ID to the generator of IDREQ.

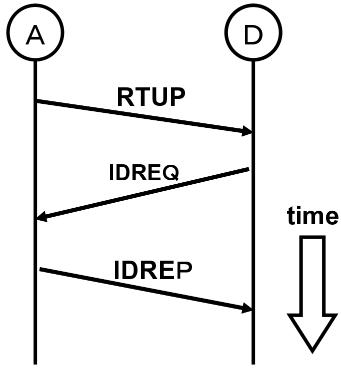
In Figure 5(a), suppose node A is going to assign a Dynamic ID to node D. Figures 6(a) and 6(b) show the routing table of node A and the sequence of the procedures, respectively. Suppose node A has the Dynamic ID [100]. When node D receives RTUP, it sends IDREQ to node A. When node A receives IDREQ from node D, it searches the subtrees which did not have the next hop registered. According to Figure 6(a), the level 2 subtree is already registered, but the level 1 subtree does have not the next hop registered. Therefore, node A assigns the Dynamic ID [110] to node D. To do this, node A sends IDREP to node D.

Update of routing table

A Node updates its own routing table using the information included in RTUP from other nodes. Suppose node B is going to update its own routing table using RTUP from node A, where the Dynamic ID length of nodes A

level	sub tree	next hop
2	0XX	S
1	11X	none
0	101	none

(a) The routing table of node A.



(b) Sequence.

Fig. 6. Dymamic ID update by nodes except the start node

and B is l bits. Also, m is smaller than l. Node B updates the entry within the its own routing table as follows:

Routing entries from level $l - 1$ **to** m
Update the value of the next hop to node A. Update node B's hop count to that of RUTP plus one.

Routing entries level $m - 1$
Update the next hop to node A, and the value of hop count to one.

Routing entries from level $m - 2$ **to** 0
Routing entries will not be updated.

We explain the sub-procedure through an example in Figure 7. Suppose node B receives RTUP from node A, and is going to update its own routing table, where the Dynamic ID length is four bits, and the Dynamic ID of nodes A and B are [0110] and [0100]. Figure 7 shows the routing tables of nodes A and B. The upper table shows the routing table prior to being updated, and the lower table is after the update. The shadowed entries are updated entries. The common prefix length of the Dynamic ID between nodes A and B is two. Node B registers node A to the entries with level 3 and 2. The hop counts of the entries with level 3 or 2 ($= 1 + 1$) is set to 2 because of the hop count of levels 3 and 2 on node A. Node B registers node A to the entry with level 1. At this time, the hop count of this entry will be set to 1. The level 1 entry will not be updated. Because our proposed method establishes multiple paths, the node registers multiple next hops to one subtree.

Packet Forwarding. The data packet forwarding sub-procedure is divided into three parts: the node lookup, next hop selection, establishing detours using alternate paths.

Node Lookup
Node lookup is the sub-procedure to seek the Dynamic ID that corresponds with the IP address of a node. In the node lookup sub-procedure, a look

Node B ID[0100]

level	sub tree	next hop	hop count
3	1XXX	S	1
2	00XX		
1	011X	C	1
0	0101	D	1

Node A ID[0110]

level	sub tree	next hop	hop count
3	1XXX	E	1
2	00XX		
1	010X	F	1
0	0111	C	1

Update

Node B ID[0100]

level	sub tree	next hop	hop count
3	1XXX	S	1
3	1XXX	A	2
2	00XX	A	2
1	011X	C	1
1	011X	A	1
0	0101	D	1

Fig. 7. Update of the routing table

up table is required. For the realization of the lookup table in our method, we proposed distributed lookup tables. A distributed lookup table consists of a pair made up of the IP address and the Dynamic ID. The pairs will be distributed to each node. A node which is assigned or re-assigned the Dynamic ID adds the pair to the distributed lookup tables.

Next Hop Selection

When a node forwards data packets, it decides the next hop according to the subtree of the destination node. When the length of the Dynamic ID is l and the common prefix length between the sender and destination node is m, the node can get the subtree level of the destination node by the calculation $l - m - 1$. The node transfers data packets to the next hop node which is indicated by the calculated result. Those sub-procedures will be repeated until the node reaches the destination nodes.

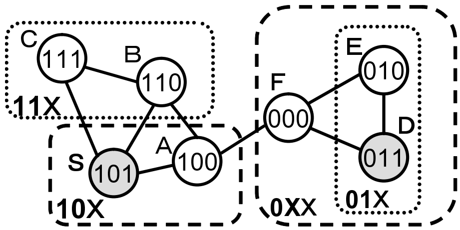
Suppose node S sends data packets to the node D in Figure 8. Node S has the routing table shown in Figure 8(a). The Dynamic ID of nodes S and D are [101] and [011], respectively. In this case, the common prefix length between nodes S and D are 0. Node A calculates the level that belongs to node D as $3 - 0 - 1 = 2$. Node S transmits the data packets to the next hop node A which are registered in the entry as level 2.

Establishing Detours Using Alternate Paths. When a node detects failure of the packet transmission due to link breakage, it detours the data packets. When the node sends the data packets, it replicates the data to the buffers before sending the data packets.

Multiple ID Trees and Countermeasures. In the proposed method, it is possible that the network has multiple ID trees. The multiple trees can be created when simultaneous requests are made to multiple unassigned nodes. When the

level	sub tree	next hop	hop count
2	0XX	A[100]	2
1	11X	B[110]	1
1	11X	C[111]	1
0	100	A[100]	1

(a) The routing table of node S.



(b) Topology.

Fig. 8. Example of data packet forwarding for proposed method

network has multiple ID trees, the data packets need to traverse the multiple ID trees. To do that, we introduced the border node table and the traversal transfer sub-procedure among multiple ID trees.

Border node tables and their maintenance

To maintain the border information, we employ the BDN (BorDer node Notification) message. Figure 9 shows an example in which the network has multiple trees. In Figure 9, " 000" on the nodes A and B indicates the start nodes for each ID tree where each ID tree is distinguished by the IP address of its start node. When a node receives RTUP from the neighboring node, it compares the ID tree identifier in the RTUP to that in its own routing table. If two ID tree identifiers differ, the node which receives RTUP recognizes it set as the border node. The border node sends BDN by unicast to the source node which has the same ID tree identifier. At this time, nodes which relay the BDN and the border node sends BDN to the assigning node for a Dynamic ID.

When the source node receives BDN, it registers the generator of BDN to its border node table. At the same time, the relay nodes of BDN register the address of the border node to their border node tables.

Traversal transfer among multiple ID trees

When the destination node locates another ID tree and it has no information about the border node, it can get the IP address of the border node using the following messages:

– BNREQ(Border Node REQuest)
– BNREP(Border Node REPly).

The source node sends BNREQ to the start node of its own ID tree. The start node that receives BNREQ seeks its border node table. When the start node finds the IP address of the border node, it sends BNREP by unicast to the source node.

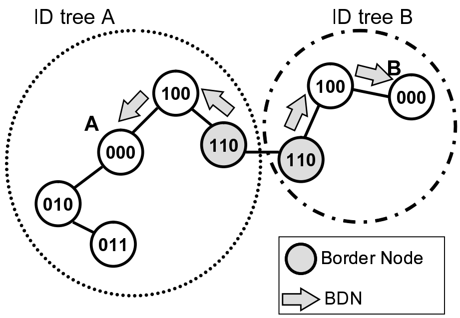


Fig. 9. Existance of multiple start nodes

4 Experiments and Discussion

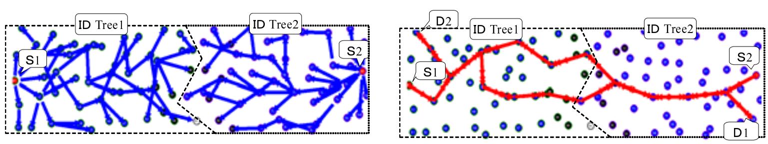
4.1 Implementation of the Proposed Method and Prelimimary Experiments

We have implemented our proposed method on the prototype system using the simulator, QualNet 4.5. In the prototype system, we use the global lookup table instead of the distributed lookup table. This is the same as in literature [2]. Using the prototype system, we confirmed the procedures for three main points: the assignment of Dynamic ID, the detour to the alternate paths, and the traversal among multiple ID trees as follows:

Traversal among multiple ID trees

Figure 10 shows the traversal between two ID trees. Figures 10(a) and 10(b) show the flow of IDN messages and the stream of data packets.

According to Figure 10, our prototype system indicates that the our design performs correctly.



(a) ID trees with two different start nodes.

(b) Data packet forwarding between S1-D1 and S2-D2.

Fig. 10. Forwarding data packets using two disjoint start nodes

We have performed the preliminary experiments using the prototype. In our preliminary experiments, we compared our prototype with AODV in terms of the data packet delivery ratio and the amount of received packets.

Table 1. Parameters for Preliminary Experiments

Simulator	QualNet ver.4.5
The number of data SD pairs	30
Data packet size[byte]	512
Interval of data [sec]	0.25
Radio area [m]	250
MAC layer protocol	IEEE802.11
Maximum bandwidth [Mbps]	2
Simulation time[sec]	500
Node maximum moving speeds [m/s]	0

Table 2. The number of nodes and field sizes

The number of nodes	100	200	300	400	500
Field size [m^2]	1550× 1550	2200× 2200	2700× 2700	3120× 3120	3500× 3500

Tables 1 and 2 show the parameters of the preliminary experiments, and the number of nodes and field sizes. The field sizes are set so that the average number of neighboring nodes is set to be around 12. This parameter is set in order to prevent the occurance having orphan nodes. For the application layer protocol, we employed CBR /UDP (Constant Bit Rate / User Datagram Protocol). The length of the Dynamic ID is 32 bits.

The following simulation performed; 1 second from the start of the simulation, the first source node starts transmission, and the second source node transmits data after the transmission of the first node. Afterwards, the other source nodes start transmission every 1 second. This is done to confirm the performance as it gives enough time to exchange route information among nodes which have been assigned their Dynamic IDs. When the node is assigned its Dynamic ID, it broadcasts RTUP every 1 second up to 10 times. When the RTUP is broadcasted more than 10 times, there is no significant change in performance.

Figure 11 shows the 30 runs average of the data packet delivery ratio versus the number of nodes for the proposed method and AODV. The x-axis and y-axis show the number of nodes and the data packet delivery ratio, respectively. The error bar indicates the level of error with 95 % accuracy of each average. According to Figure 11, the proposed method is almost identical to AODV.

Figure 12 shows the 30 runs average of the amount of received data packets versus the number of nodes for the proposed method and AODV. The x-axis and y-axis show the number of nodes and the amount of received data packets, respectively. The error bar indicates the level of error with 95 % accuracy of each average. According to Figure 12, the proposed method is almost identical to AODV.

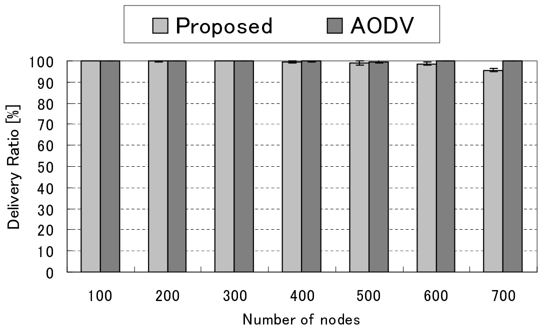


Fig. 11. Packet delivery ratio

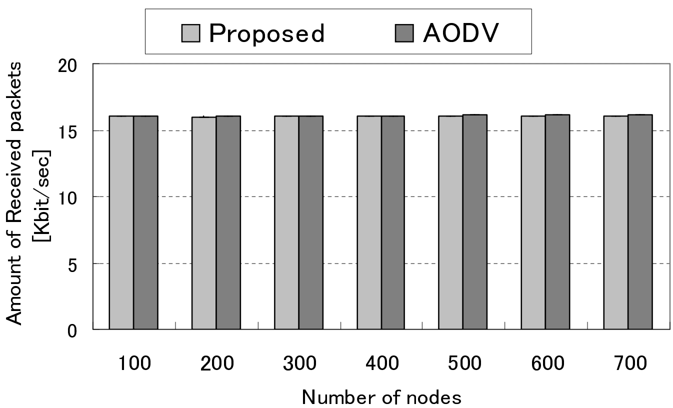


Fig. 12. The amount of received packets

4.2 Experiments to Evaluation

We performed simulation experiments to evaluate our prototype implementation. We define the term "routing load" as the proportion of the number of control packets in the network to the received data packets of the destination nodes. In the experiment, we compared our method to AODV and AOMDV in terms of the number of control packets and the routing load.

Number of Control Packets. The parameters of the experiments are shown in Table 3. Table 4 shows the number of nodes and the field sizes. Figure 13 depicts the 30 runs average of the number of control packets in the network for the proposed method and AODV. The x-axis and the y-axis show the number of nodes and the number of control packets. The error bar indicates the level of error with 95 % accuracy of each average. According to Figure 13 , the proposed method has a smaller number of control packets than that of AODV. In addition,

Table 3. Parameteres for experiments of the number of control packets

Simulator	QualNet ver.4.5
The number of SD pairs	30
Data packet size [byte]	512
Interval of data packets [sec]	0.25
Radio Area[m]	250
MAC Layer protocol	IEEE802.11
Maximum Bandwidth[Mbps]	2
Simulation time [sec]	500
Maximum node speed [m/s]	0

Table 4. The relation between number of nodes and field sizes

The number of nodes	100	200	300	400	500
Filed size [m^2]	1550× 1550	2200× 2200	2700× 2700	3120× 3120	3500× 3500

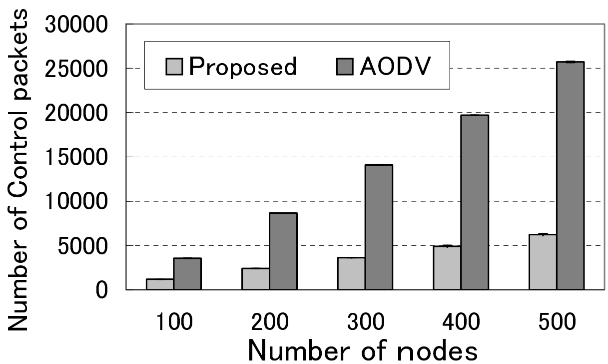


Fig. 13. The number of control packets

as the number of nodes increases, the gap between our method and AODV widens. While AODV uses flooding to discover routes, our method does not. The gap expresses the difference between the proposed method and AODV.

The Routing Load. Then, we measured the routing load under the conditions as given in Table 5. The number of nodes and the field sizes are the same as in preliminary experiments with exception of the number of source and destination node pairs. Figure 6 shows the 30 runs average of the results. According to Figure 6, our proposed method has the smallest routing load among our proposed method, AODV, and AOMDV. While AODV and AOMDV use flooding to discover routes, our method does not. The gap expresses the difference among the proposed method, AODV, and AOMDV.

Table 5. Parameters for experiment of the routing load

The number of nodes	100
Field Size[m^2]	2200×600
The number of SD pairs	25

Table 6. Routing Load

	Routing Load
Proposed	0.482
AOMDV	1.340
AODV	1.583

End-to-end Delay. We also have investigated the end-to-end delay between the source and destination nodes. Table 7 shows the results of the average time of the end-to-end delay for our proposed method, AOMDV, and AODV. The results for each method in Table 7 are the average of 30 runs each. The results of our proposed method and AODV are from our simulation experiments. The results of AOMDV are based on the data from literature [5]. Figure 14 shows the results of the end-to-end average delay versus the number of nodes. The results in Figure 14 are also the average of 30 runs. Table 8 shows the average number of hops between the source and destination nodes. The results in Table 8 show that the average number of hop counts for our proposed method is longer than that of AODV by about one hop(s).

Table 7. The average time of the end-to-end delay for our proposed method, AOMDV, and AODV

	End-to-end delay (sec)	confidence interval
Proposed	0.099	0.005
AODV	0.075	0.004
AOMDV	0.041	0.008

Table 8. The average number of hops between the source and destination nodes

Number of nodes	Proposed	AODV
100	4.82	4.10
200	6.82	5.62
300	8.57	6.90
400	9.57	7.91
500	10.98	10.27

According to Tables 7 and 8, the end-to-end delay time of the proposed method is the longest among our proposed method, AODV, and AOMDV. This is because our method constructs tree-based multiple-paths. In contrast, AODV

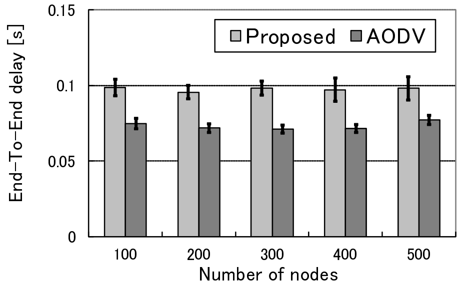


Fig. 14. The average time of the end-to-end delay vs the number of nodes for our proposed method and AODV

constructs the shortest path. Also, since AOMDV is the modified method of AODV, the end-to-end delay time becomes the quasi-shortest path. Furthermore, AOMDV compensates for unstable wireless links by using alternate paths. As a result, AOMDV has the shortest end-to-end delay. However, the values of the end-to-end delay time of the proposed method, AODV and AOMDV are small. When the route breaks occur frequently in the networks, this trend will be possible to change. Further study will be necessary to investigate the performances for our proposed method when the route breaks occur frequently.

5 Conclusion and Future Work

In this paper, we have proposed a new multipath routing method to decrease the routing load based on DART. Its prototype system has been implemented. In addition, we have evaluated our proposed method by conducting simulation experiments of our method, AODV, and AOMDV. According to the results, our method can establish multiple paths with smaller routing loads than AODV and AOMDV.

As for future work, we plan to extend our proposal to node mobility, and to investigate the potential scope of application of our method.

Acknowledgment

This research is supported by the Ministry of Education, Science, Sports and Culture of Japan under Grant-in-Aid for Scientific Research (B) (No.21300028). This research is also supported by the National Institute of Information and Communications Technology, Japan under Early-concept Grants for Exploratory Research on New-generation Network (No.145-9), and the Ministry of Education, Science, Sports and Culture of Japan under Grant-in-Aid for Scientific Research (C) (No.22500065).

References

1. Abidi, S., Erfani, S.: A multipath routing survey for mobile ad-hoc networks. In: Proceedings of the 3rd IEEE Consumer Communications and Networksing Conference (CCNC 2006), vol. 2, pp. 984–988 (2006)
2. Eriksson, J., Faloutsos, M., Krishnamurthy, S.V.: DART: Dynamic address routing for scalable ad hoc and mesh networks. IEEE/ACM Transactions on Networking 15(1), 119–132 (2007)
3. Johnson, D.B., Maltz, D.A.: Dynamic source routing in ad hoc wireless networks. In: Mobile Computing, vol. 353, pp. 153–181. Kluwer Academic Publishers, Dordrecht (1996)
4. Lee, S.-J., Gerla, M.: Split multipath routing with maximally disjoint paths in ad hoc networks. In: Proceedings of IEEE International Conference on Communications (ICC 2001), vol. 10, pp. 3201–3205 (2001)
5. Marina, M.K., Das, S.R.: Marina and Samir R. Das. Ad hoc on-demand multipath distance vector routing. SIGMOBILE Mobile Computing and Communication Review 6(3), 92–93 (2002)
6. Perkins, C., Belding-Royer, E., Das, S.: Ad hoc On-demand Distance Vector (AODV) routing. RFC 3561 (Experimental) (July 2003)
7. Perkins, C.E., Royer, E.M.: Ad-hoc on-demand distance vector routing. In: Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications, pp. 90–100. IEEE Computer Society Press, Los Alamitos (February 1999)
8. Scalable Network Technologies, Inc., QualNet network simulator