Global Journal of Computer Science and Technology
Network, Web & Security
Volume 12 Issue 17 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Node Disjoint Multipath Routing Approach for Controlling Congestion in Manets

By Mr. Abhishek Bande & Mr. Gaurav Deshmukh

University of Pune, Maharashtra India

Abstract - Mobile Ad hoc Networks are highly dynamic networks. Quality of Service (QoS) routing in such networks is usually limited by the network breakage due to either node mobility or energy depletion of the mobile nodes. Node-disjoint routing becomes inessential technique in communication of packets among various nodes in networks. Meanwhile AODV (Ad Hoc On-demand Multipath Distance Vector) creates single-path route between a pair of source and destination nodes. Some researches has done so far to make multipath node-disjoint routing based on AODV protocol. But however their overhead and end-to-end delay are relatively high, while the detail of their code is not available too. In an ad hoc network, identification of all node-disjoint paths between a given pair of nodes is a challenging task. The phenomena that a protocol is not able to identify all node-disjoint paths that exist between a given pair of nodes is called path diminution. In this paper, we discuss that path diminution is unavoidable when a protocol discovers multiple node-disjoint paths in a single route discovery and working of node disjoint multipath protocol.

Keywords : *routing; ad hoc networks; path diminution; node-disjoint multipath routing; single route discovery, node disjoint routing.*

GJCST-E Classification : *C.2.2*

Strictly as per the compliance and regulations of:

Node Disjoint Multipath Routing Approach for Controlling Congestion in Manets

Mr. Abhishek Bande [α] & Mr. Gaurav Deshmukh [σ]

Abstract - Mobile Ad hoc Networks are highly dynamic networks. Quality of Service (QoS) routing in such networks is usually limited by the network breakage due to either node mobility or energy depletion of the mobile nodes. Node-disjoint routing becomes inessential technique in communication of packets among various nodes in networks. Meanwhile AODV (Ad Hoc On-demand Multipath Distance Vector) creates single-path route between a pair of source and destination nodes. Some researches has done so far to make multipath node-disjoint routing based on AODV protocol. But however their overhead and end-to-end delay are relatively high, while the detail of their code is not available too. In an ad hoc network, identification of all node-disjoint paths between a given pair of nodes is a challenging task. The phenomena that a protocol is not able to identify all node-disjoint paths that exist between a given pair of nodes is called path diminution. In this paper, we discuss that path diminution is unavoidable when a protocol discovers multiple node-disjoint paths in a single route discovery and working of node disjoint multipath protocol.

Keywords : routing; ad hoc networks; path diminution; node-disjoint multipath routing; single route discovery, node disjoint routing.

I. Introduction

Mobile ad hoc network is a type of wireless that is composed of wireless mobile nodes. Each mobile node dynamically changes the network topology without relying on a wired backbone network or a fixed base station. Mobile nodes in MANETs are constrained by their limited power, processing, memory resources and high degree of mobility. In such networks, the wireless mobile nodes may dynamically join or leave the network topology. In MANETs, many routing protocols have been suggested to communicate between mobile nodes. And pertinent routing protocols are used in various network environment and application.

Multipath routing protocols search node-disjoint, link disjoint or non-disjoint routes during the rout discovery Process. Node-disjoint routes have completely disjoint routes where there are no nodes or links in common. Link-disjoint routes have no links in common but may have nodes in common. Non-disjoint routes may use nodes or links in common. Following are the figures which explains the what are the node disjoint paths and link disjoint paths.

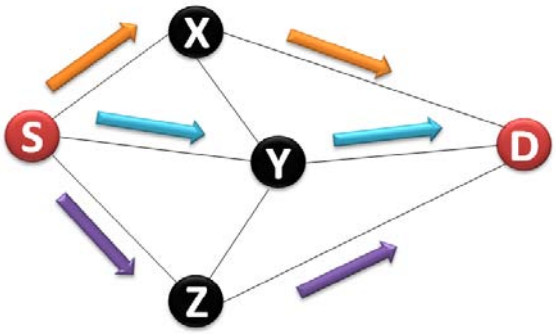


Fig (a) : Node Disjoint

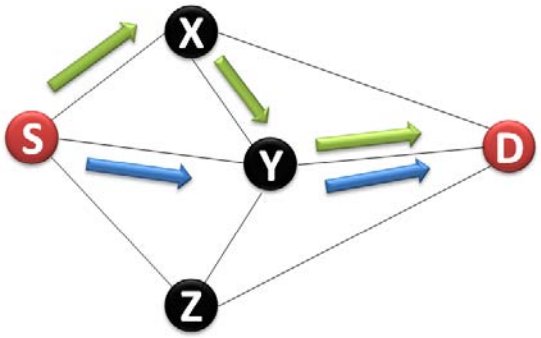


Fig (b) : Link Disjoint

If a node or link fails (and it is used by the main and backup route) in non-disjoint and link-disjoint routes, then main and backup routes will be disconnected at the same time. However in node-disjoint routes, main routes and backup routes use completely different nodes or links. Therefore, even though main route will be disconnected, data transmission may be available through the backup route.

In single-path routing protocols, route maintenance may be Performed after route fail. Therefore, data transmission will be stopped while the new route is established, causing data transmission delay. On the other hand, multipath routing protocols perform the route maintenance process even if only one route fails among the multiple routes. To perform the route maintenance process before all routes fail, the network must always maintain multiple routes. This can reduce data transmission delays caused by link failure.

Other than the source and the destination, node-disjoint paths have no node in common, while link-

Author α σ : Department of Computer Engineering, University of Pune, Maharashtra India. E-mails : abhishek.bande2008@gmail.com, deshmukhgaurav9@gmail.com

disjoint paths do not share any link. Clearly, node-disjoint paths are also link-disjoint paths. Node-disjoint paths are desirable where node resources are Scarce or when nodes are susceptible to failure. On the other hand, link-disjoint paths are preferred where link resources are scarce or when only links are susceptible to failure. Using link-disjoint paths, one can address the issue of fault-tolerance. However, link-disjoint paths cannot be used for simultaneous data transfer because doing so will overload the nodes that are common in more than one path. Using node-disjoint paths, one can address fault-tolerance as well as load sharing.

II. Related Work

Multipath routing establishes multiple routes between source and destination nodes. For fault tolerance, even if one route failure occurs, source nodes can maintain connections by using other routes. So multiple routing protocols can reduce data transmission failures and delay times that are caused by route disconnection. Multipath routing protocols search node-disjoint, link disjoint or non-disjoint routes during the route discovery process. Node-disjoint routes have completely disjoint routes where there are no nodes or links in common. Link-disjoint routes have no links in common but may have nodes in common. Non-disjoint routes may use nodes or links in common. If a node or link fails (and it is used by the main and backup route) in non-disjoint and link-disjoint routes, then main and backup routes will be disconnected at the same time. However in node-disjoint routes, main routes and backup routes use completely different nodes or links. Therefore, even though main route will be disconnected, data transmission may be available through the backup route. In single-path routing protocols, route maintenance may be performed after route fail. Therefore, data transmission will be stopped while the new route is established, causing data transmission delay. On the other hand, multipath routing protocols perform the route maintenance process even if only one route fails among the multiple routes. To perform the route maintenance process before all routes fail, the network must always maintain multiple routes. This can reduce data transmission delays caused by link failure.

Several implementation of multipath routing are based on AODV; typical examples are AOMDV, AODVM, AODV-BR and MP-AODV protocols. The AOMDV [2] protocol establishes loop-free link-disjoint paths in the network. When intermediate nodes receive the RREQ packet from the source node, AOMDV stores all RREQ packets, unlike conventional AODV, which discards duplicates. So, each node maintains a firsthop-list where information from additional field called firsthop in RREQ packet to indicate the neighbor node of the source nodes. If firsthop of received RREQ packet is duplicated from its own firsthop-list, the RREQ packet is discarded. On the other hand, the RREQ packet is not duplicated from previous RREQ packets. Then the node updates the nexthop, hopcount and advertised-hopcount in routing table. At the destination, RREP packets are sent from each received RREQ packet. The multiple routes are made by RREP packets that are follow the reverse routes that have been setup already in intermediate nodes [6].

For the AODVM protocol, intermediate nodes are not allowed to send a RREP packet directly to the source node. Also, intermediate nodes do not discard the duplicate RREQ packets. But the intermediate nodes record all received RREQ packets in routing table. The destination node sends an RREP for all the received RREQ packets. An intermediate node forwards a received RREP packet to the neighbor in the routing table. Whenever a node overhears one of its neighbors broadcasting RREP packet and it removes that neighbor from its routing table, because nodes cannot participate in more than one route.

For the AODV-BR protocol, neighbor nodes overhear the RREP packets for establishing and maintaining the backup routes during the route initiation process. If part of the main route is broken, nodes broadcast error packets to neighbor nodes. When neighbor nodes receive the error packet, they establish an alternate route using information about the overheard RREP packets previously. AOMDV has the overhead of storing multiple next hops and hop counts and the first hop list for each destination. By overhearing the neighbor's packets, AODVM may not establish alternate routes depending on the path along which the RREP packets are sent. Moreover, to speak strictly, AODV-BR is not a multipath routing protocol, because it only maintains bypass routes when the main route is broken by using the neighbor nodes around the main routes. MP-AODV protocol uses the modified RREQ and RREP packet that has additional 1bit flag 'F'. This flag distinguishes the packet into the main route (RREQ, RREP) or backup route (RREQ_2, RREP_2) route discovery processes. Unlike a conventional AODV, intermediate nodes that receive the RREP packet increment the RREQ ID value in the seen table. By incrementing the RREQ ID value, the protocol ensures that a backup route will not use any nodes that belong to the main route. When a source node receives the RREP packet, the main route is established, and the source node starts data transmission and broadcasts the RREQ_2 packet (a packet with a RREQ ID value of two) for simultaneously searching a backup route. RREQ_2 is a packet for establishing a backup route, and its flag bit F is set to one. When the RREQ_2 packets are delivered to the intermediate nodes, the RREQ ID values in the seen table are compared with the RREQ ID values in the RREQ_2 packets. If they are identical, the nodes discard the RREQ_2 packet. If not, the nodes forward the RREQ_2 packet continuously.

When nodes belonging to the main route receive the RREP packet, the RREQ ID value in the RREQ_2 packet and the RREQ ID value in the seen table are identical because the protocol has already increased the RREQ ID value in the seen table during the previous route discovery process. After this process, the intermediate nodes belonging to the main route do not join in the backup routes.

MP-AODV has high control overhead and end-to-end delay, because it uses at last five control packets to establish two node-disjoint route.

III. Multipath Routing Protocol

We call the proposed protocol Vertex Disjoint Multipath Routing (VDMR). Like other on-demand protocols, VDMR is also based on the request–reply paradigm. We describe the protocol in two phases: *route discovery* and *route maintenance*.

a) Route discovery

If a source node, *S*, wishes to communicate with a destination node, *D*, and it does not have a route to the destination, it initiates a route discovery. To initiate a route discovery, node S broadcasts a route request (RREQ). The transmitted RREQ is heard by all nodes which are in the transmission range of the source. The RREQ carries the following information in its header: *Source Address, Destination Address, Source Seq No, Path Traversed*. A node that is neither the source nor the destination of an RREQ is called an intermediate node. The processing of an RREQ at a node will differ depending upon whether the node is the source/destination of the RREQ, or an intermediate node. An intermediate node maintains an *RREQ Cache* where it stores information about the RREQs forwarded earlier. Note that two or more RREQs are said to copies of one another if they have the same *Source Address, Destination Address, Source Seq No*.

If node *i* has already forwarded a copy of the RREQ or if the address of node itself is present on the *Path Traversed* of the RREQ, then it discards the RREQ. When an RREQ reaches node *D*, it stores the RREQ in *RREQ Cache*. Node *D* collects all copies of an RREQ received and saved in *RREQ Cache* before the expiry of a timeout. Upon expiry of the timeout, destination *D* reads all RREQs cached in its *RREQ Cache* and computes there from a maximal set of node-disjoint paths (using the heuristics described in Appendix B) by inspecting the *Path Traversed* in each of the RREQs. Node *D* then sends multiple route replies3 (RREP), one for each node-disjoint path. It also stores the RREPs in its *RREP Cache*. An RREP contains the following information in its header: *Source Address, Destination Address, Source Seq No, Reverse Path*_, where, *Reverse Path* of the RREP is *Path Traversed* of the RREQ in reverse direction. Note that the multiple RREPs differ in *Reverse Path*. Upon receiving an RREP, an intermediate node updates its routing table and unicasts the RREP to the next node along the path. When node *S* receives an RREP, it stores the path to node *D* in its *Route Cache*.

b) Route maintenance

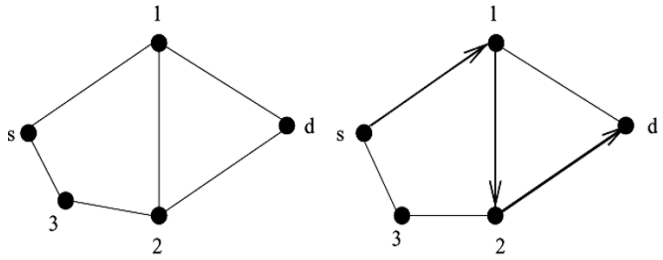
If a link failure occurs while transmitting data packets, then a node sensing link failure generates a route error (RERR) message. An RERR contains the route to node *S*. The RERR message informs upstream nodes about the link failure. All the nodes update their routing tables by deleting the entry of the path along which a link failure occurred. When an RERR reaches node *S*, it also updates its routing table by deleting the entry of the path that has failed. Then, it looks up its routing table for a path. If it finds a path, it starts sending data packets along the path. Otherwise node *S* initiates a new route discovery. In what follows, we discuss that a protocol may or may not be able to discover all node-disjoint paths that exist in the network between a given pair of nodes.

c) Path diminution

By 'path diminution'4 we mean that the number of node-disjoint paths discovered by a protocol is less than the number of node-disjoint paths that exist in the network between a given pair of nodes. There are two reasons of path diminution: RREQ forwarding policy and computation of disjointness at the destination. We are, here, concerned with the occurrence of path diminution due to an RREQ forwarding policy A policy which is generally adopted in single path routing protocols such as Dynamic Source Routing (DSR) (Johnson and Maltz, 1996) and Ad hoc On-demand Distance Vector Routing (AODV) (Perkins and Royer, 1999) is that only the first copy of an RREQ is forwarded at an intermediate node, while other copies are discarded. This policy works well for finding *a* path from a given source to a destination. However, if one wishes to use it in a protocol to discover multiple node-disjoint paths, the protocol may or may not be able to discover all node-disjoint paths that exist between a given pair of nodes.

d) Example

Consider a network shown in Figure (c). There are two node-disjoint paths between *s* and *d*. If node 2 receives a copy of an RREQ from node 3 before it receives a copy from node 1, then it broadcasts this copy of the RREQ to its neighbors. The destination receives two copies of the RREQ with *Traversed Path* <3, 2> and <1>. When the destination computes disjointness, it finds two node-disjoint paths.



(a) (b)

Fig (c) : A network with 2 node-disjoint paths from *s* to *d*

However, if node 2 receives a copy of an RREQ from node 1 before it receives a copy from node 3, node 2 broadcasts the copy of the RREQ to its neighbors. It discards the copy of RREQ received from node 3. Two copies of the RREQ with *Traversed Path* <1, 2> and <1> reach the destination. When the destination computes the disjointness, it finds only one path. Although, there exist two node-disjoint paths between *s* and *d*, the protocol is not able to find them.

e) An expensive solution

One method for mitigating path diminution can be described as follows. An intermediate node discards only those copies of an RREQ that cause loops. The node forwards all other copies of an RREQ after appending its own address to the *Path Traversed* of the copy of the RREQ. The destination collects all copies of the RREQ before computing disjointness. Assuming that an algorithm for computing the maximum set of node-disjoint paths is available, one will be able to discover all node-disjoint paths. The number of RREQs that an intermediate node has to transmit can be as large as $(n - 3)!$ 5 Further, the number of copies of an RREQ that may reach the destination can be as large as $(n - 2)!$ 6 As a result, this scheme requires an exponential amount of computational and communication overheads. Clearly, this solution is not acceptable in an ad hoc network where resources of nodes are limited.

f) Mitigating path diminution

We propose three schemes to mitigate path diminution. In a fundamental sense, these schemes play upon the number of RREQs that each intermediate node forwards and the manner in which it selects the RREQs that are forwarded. However, the features that are common among all proposed schemes are as follows.

- To prevent loops, an intermediate node checks whether its own address is already present in *Path Traversed* of the RREQ. If that is so, it discards the RREQ. Otherwise, it forwards the RREQ according to a stated policy.
- Before forwarding a copy of an RREQ, an intermediate node appends its own address to *Path Traversed* of the RREQ.
- To keep a record of the RREQs forwarded, a node maintains a *RREQ Cache*. Before forwarding a copy of the RREQ, a node stores the RREQ in its *RREQ Cache*.

We now discuss the features of the proposed schemes that are different.

i. *Path diminution in node-disjoint multipath routing*

1. *All Disjoint Copies (ADC).* An intermediate node forwards all node-disjoint copies of an RREQ and discards the copies which are not node-disjoint. Upon receiving a copy of an RREQ, an intermediate node checks whether *Path Traversed* of the copy of the RREQ is disjoint with those already forwarded. If that is so, it forwards the copy of the RREQ. Otherwise, it discards the copy of the RREQ.
2. *Two Disjoint Copies (2DC).* An intermediate node forwards the first copy of an RREQ. It also forwards another copy of the RREQ, if any, provided its *Path Traversed* is node-disjoint with that already forwarded. Other copies of the RREQ are discarded. To keep a record of *number of copies* forwarded with disjoint *Path Traversed*, an intermediate node maintains a counter *rreq Count*. The variable *rreq Count* is initially set to 0 and is incremented each time when the node forwards a copy of the RREQ. If *rreq Count* reaches 2, the node discards subsequent RREQs (if any).
3. *At most one Copy per Neighbour (OCN).* An intermediate node forwards at most one copy from each neighbour. It discards the duplicate copy from a neighbour. Upon receiving a copy of an RREQ, an intermediate node checks the previous hopID in *Path Traversed* of the RREQ. An intermediate node forwards the copy of the RREQ if and only if the previous hopID does not match with the previous hopID of any of the copy of the RREQ stored in its *RREQ Cache*. Let us return to the example shown in Figure 1(a). Suppose each intermediate node forwards 2 disjoint copies of an RREQ. Node *D* will receive 3 copies of the RREQ with *Path Traversed* <1>, <1, 2> and <3, 2>, respectively.7 Upon computing disjointness, the destination identifies two node-disjoint paths with *Path Traversed* <1> and <3, 2>. The destination sends two RREPs one along each node disjoint paths. In other words, if 2DC is used as RREQ forwarding policy, the protocol will be able to find both node-disjoint paths in the example network. There is no guarantee that any of these schemes will always discover all node-disjoint paths between a given pair of nodes. However, these schemes are adopted to introduce a diversity in the *Path Traversed* of the copies of an RREQ that reach the destination. In other words, adoption of the policies discussed above can enhance the chances of forwarding those copies of

an RREQ which have the potential of obtaining disjoint *PathTraversed* at the destination node.

ii. *Node Disjoint Multipath Routing Considering Link and Node Stability*

The main aim of the proposed work is to find the multiple node disjoint routes from source to a given destination Also it keeps track of the route bandwidth which can be further used by the source to select the optimal routes. From the factors Link Expiration Time (LET) [19] and Drain Rate (DR) [22] it is inferred that the Link Stability:

a) Depends directly on Mobility factor
b) Depends inversely on the energy factor

Hence, Link Stability Degree (LSD) is defined as:

$$LSD = \text{Mobility factor} / \text{Energy factor} \quad (3)$$

It defines the degree of the stability of the link. Higher the value of LSD, higher is the stability of the link and greater is the duration of its existence. Thus, a route having all the links with LSD > LSDthr is the feasible. We choose the Dynamic Source Routing (DSR) [5] protocol as a candidate protocol. Modifications are made to the Route Request (RREQ) and Route Reply (RREP) packets to enable the discovery of link stable node disjoint paths. The proposed scheme has three phases: Route Discovery, Route Selection and Route Maintenance. The various phases are described as follows:

a. *Route Discovery*

The source node when needs to send packet to some destination node, starts the route discovery procedure by sending the Route Request packet to all its neighbors .In this strategy , the source is not allowed to maintain route cache for a long time, as network conditions change very frequently in terms of position and energy levels of the nodes. Thus, when a node needs route to the destination, it initiates a Route Request packet, which is broadcasted to all the neighbors which satisfy the broadcasting condition. Route Request packet of NDMLNR is shown in figure.

SA	DA	Type	ID	TTL	Hops	Bandwidth	LSD	Path	Velocity	Direction	Position

Fig. (d) : RREQ packet

Type (T) field: It indicates the type of packet.

SA (Source Address) field: It carries the source address of node.

ID field: unique identification number generated by source to identify the packet.

DA (Destination Address) field: It carries the destination address of node.

Time to Live (TTL) field: It is used to limit the life time of packet, initially, by default it contains zero.

Hop field: It carries the hop count; the value of hop count is incremented by one for each node through which packet passes. Initially, by default this field contains zero value.

LSD field: when packet passes through a node, its LSD value with the node from which it has received this packet is updated in the LSD field. Initially, by default this field contains zero value.

Bandwidth field: carries the cumulative bandwidth of the links through which it passes; initially, by default this field contains zero value.

Path field: It carries the path accumulations, when packet passes through a node; its address is appended at end of this field.

The node's current velocity, direction and position are updated at each node in the respective fields before forwarding the RREQ packet.

Every node maintains a Neighbor Information Table (NIT), to keep track of multiple RREQs. With following entries Source Address, Destination Address, Hops, LSD, ID and bandwidth.

SA	DA	ID	Hops	LSD	Bandwidth

Fig (e) : Neighbor Information Table (NIT)

As RREQ reaches a node it enters its information in the NIT. It makes all the entries for the requests till Wait Period. At the end of the Wait Period, it accepts the request with the highest value in LSD field. It adds the value of the link bandwidth to the Bandwidth field of the RREQ packet. If two RREQs have same LSD values, the one with lesser value of hop count is selected. In case, hops are also same, one with higher bandwidth is selected. In the worst case, RREQ is selected on First-come-first -serve basis. This prevents loops and unnecessary flooding of RREQ packets. None of the intermediate nodes is allowed to send RREP if it has the current route to the destination. As doing this may lead to those paths which do not fulfill current QoS requirements.

b. *Route Maintenance*

In case, LSD of a node falls below LSDthr, it informs its predecessor node of the node failure by sending the NODEOFF message. Once a node receives such a message, it sends the ROUTEDISABLE message to the source node. Source can then reroute the packets to the backup routes. If no backup route exists, the source then starts the route discovery procedure again.

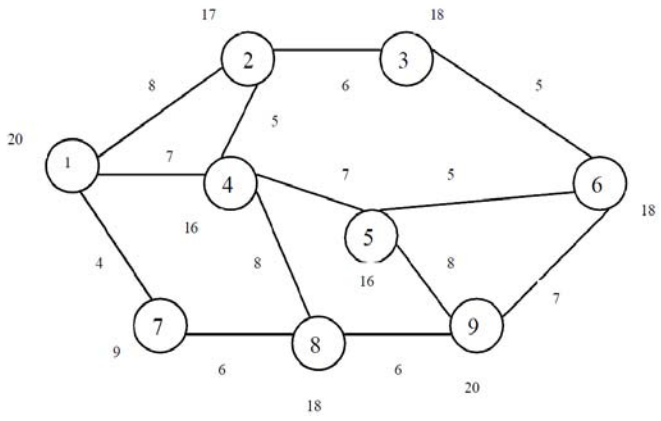


Fig (f) : An example network

Let us illustrate our technique with the following example network shown in figure (e). Suppose node 1 is the source node and node 6 is the destination. Let LSD equals to 15. Let B equals to 5 mbps. To send the packet, node 1 checks its neighbors (2.4.7) for their LSD value Out of these node 7 has value 9<15. So, node 1 sends the packets only to nodes 2 and 4. Node 2 receives this packet for the first time, makes entry in its NIT for the RREQ packet as (1, 6, 1, 1, 20, and 8) and starts Wait Time, 5 secs here. Node 2 now checks its neighbors, updates the path field as, 1-2 and the bandwidth field to 8 and forwards RREQ to both 4 and 3. At node 4, it may receive two RREQ packets during Wait Time. One from node 1 directly and the other via node 2. It has two entries in its NIT (1,6,1,1.20,8) and (1,6,1,2,17,13). At this moment it selects the one from node 1 with higher LSD value, 20. It updates the path field of the RREQ packet as 1-4 and the bandwidth field to 7. It forwards the packet to both its neighbors, 5 and 8, with LSD values 16 and 18 respectively. Node 3 has only one neighbor, 6 which satisfies the LSD value and hence, it updates RREQ path field as 1-2-3 and the bandwidth field to 14 and forwards the packet to node 6. Node 6 now receives a path from source node 1. It appends its own ID to it. Thus, first path is 1-2-3-6 and bandwidth of this path is 17. Node 5 after receiving the RREQ packet with path 1-4, checks for its neighbors and forwards RREQ with updated path field to 1-4-5 and bandwidth field to14 to nodes 9 and 6 Node 6 now receives another path,1-4-5.It appends its ID to it, to get the path, 1-4-5-6 with bandwidth 19. Node 8 after receiving the RREQ packet forwards it to its neighbor, 9, after updating path field to 1-4-8 and bandwidth field to 15 Node 9 can receive two packets in its wait time, one from node 5 and the other from node 8. It updates its NIT as (1,6,1,3,16,22) and (1,6,1,3,18,21). To select from the one, it chooses one from node 8 as its LSD value is higher, 18. It then forwards the request after updating the path field as 1-4-8-9 and bandwidth field to 21. Node 6 again receives another path 1-4-8-9.It appends its ID to this path to get 1-4-8-9-6 with bandwidth 28.Now node 6 receives two paths 1-4-5-6 and 1-4-8-9-6 with node 4 as common node. It selects the one with higher bandwidth i.e. Path, 1-4-8-9-6 with bandwidth 28.

IV. Conclusion

In this paper, we proposed a routing protocol that establishes two node-disjoint routes between source and destination nodes based on AODV protocol for MANETs. NMN-AODV uses three control packets for establishes two routes, but MP-AODV uses five control packets. Thus NMNAODV has law overhead to MP-AODV. In addition, two routes will not break at the same time because the protocol uses node-disjoint multiple routes that are not duplicated between main and backup routes. NMN-AODV establishes two node-disjoint faster than MP-AODV because NMNAODV starts to establish backup route faster than MP-AODV. Thus nd-to-end delay is lawyer than MP-AODV. Also this protocol sends the data immediately after the main route is found by separating the main route and backup route discovery process to reduce the data transmission delay. In the future work, we will compare NMN-AODV with other multipath routing protocols based on AODV such as AOMDV, AODVM and AODV-BR.

References Références Referencias

1. W. Cheng, A. Y. Teymorian, L. Ma, X. Cheng, X. Lu, and Z. Lu. Underwater localization in sparse 3d acoustic sensor networks. In *IEEE INFOCOM*, 2008.
2. D. Goldenberg, A. Krishnamurthy, W. Maness, Y. R. Yang, A. Young, A. S. Morse, A. Savvides, and B. Anderson. Network localization in partially localizable networks. In *IEEE INFOCOM*, 2005.
3. D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. *Mobile Computing*, pages 153–181, 1996.
4. S.-J. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *ICC*, 2001.
5. S. Li and Z. Wu. Node-disjoint parallel multipath routing in wireless sensor networks. In *ICESS*, 2005.
6. C. E. Perkins and E. M. Royer, "Ad hoc On-Demand Distance Vector Routing (AODV) ", IETF RFC 3561, 2003.
7. Z. Hass and R. Pearlmann, "Zone routing Protocol", IETF Internet Draft, 1999.
8. RFC2386.
9. S.Chen and K.Nahrstedt, "An Overview of Quality-of-Service Routing for the next Generation High -Speed Networks: Problems and Solutions", IEEE Network Magazine, vol12, 1998, pp. 64 -79.
10. M. K. Marina and S. R. Das, "On-Demand MultiPath Distance Vector Routing in Ad hoc Networks", Proceedings of the Ninth International Conference on Network Protocols (ICNP}, IEEE Computer Society Press, 2001, pp. 14-23.

11. X. Hong, K. Xu, M. Gerla, "Scalable Routing Protocols for Mobile Ad Hoc Networks," IEEE Network, Vol. 16, No. 4, pp. 11-21, 2002.
12. L. Blazevic, S. Giordano, and J. Le Boudec, "Self Organized Terminode Routing," Cluster Computing, Springer Science Business Media, pp. 205-218, 2002.
13. Y. Ge, G. Wang, Q. Zhang, and M. Guo, "Multipath Routing with Reliable Nodes in Large-Scale Mobile Ad-Hoc Networks," IEICE Transactions on Information and Systems, Vol. E92-D, No. 9, pp. 1675-1682, Sept. 2009.
14. M.T.Toussaint, "Multipath Routing in Mobile Ad Hoc Networks", *TUDelft/ TNO Traineeship Report*.
15. Chang-Woo Ahn, Sang-Hwa Chung, Tae-Hun Kim, Su- Young Kang, "A Node-Disjoint Multipath Routing Protocol Based on AODV in Mobile Ad-hoc Networks", International Conference on Information Technology, IEEE, 2010.

This page is intentionally left blank