MRFR - Multipath-based Routing Protocol with Fast-Recovery of Failures on MANETs

Hoai Phong Ngo[1] and Myung Kyun Kim[1]

[1] School of Electrical Engineering, University of Ulsan

Ulsan, South of Korea

[e-mail: mkkim@ulsan.ac.kr]

*Corresponding author: Myung Kyun Kim

Received June 5, 2012; revised September 13, 2012; accepted October 18, 2012; published December 27, 2012

Abstract

We propose a new multipath-based reliable routing protocol on MANETs, Multipath-based Reliable routing protocol with Fast-Recovery of failures (MRFR). For reliable message transmission, MRFR tries to find the most reliable path between a source and a destination considering the end-to-end packet reception reliability of the routes. The established path consists of a primary path that is used to transmit messages, and the secondary paths that are used to recover the path when detecting failures on the primary path. After establishing the path, the source transmits messages through the primary path. If a node detects a link failure during message transmission, it can recover the path locally by switching from the primary to the secondary path. By allowing the intermediate nodes to locally recover the route failure, the proposed protocol can handle the dynamic topological change of the MANETs efficiently. The simulation result using the QualNet simulator shows that the MRFR protocol performs better than other protocols in terms of the end-to-end message delivery ratio and fault-tolerance capability.

Keywords: MANETs, reliable routing, multipath-based routing, fast-recovery of failures

This research was supported by Basic Science Research Program through the Na-tional Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2010-0024245).

http://dx.doi.org/10.3837/tiis.2012.12.003

1. Introduction

Mobile Ad Hoc NETworks (MANETs) are collections of nodes that can communicate with each other using multi-hop wireless links without utilizing any fixed infrastructure or centralized management. Each node in the network can move and act as both a host and a router relaying packets toward a destination. Since the network topology is continuously changing in MANETs due to the frequent movement of nodes resulting in frequent broken links, discovering and maintaining effective routes to destinations are critical tasks [1]. Reliable message transmission in MANETs is an issue since wireless links are prone to failure due to node movement, and wireless communication between nodes is susceptible to all kinds of interference. For reliable message transmission in MANETs, the most reliable route between a source and a destination has to be set up during the route discovery step, and route failures due to broken links or nodes on the message transmission path must be recovered quickly with not much overhead. Multipath-based routing protocols are an effective strategy to improve reliability in the face of routing failures caused by unreliable links or frequent topological changes [1]. Considering the dynamic topological changes of wireless networks, many multipath-based routing protocols such as NDMR[2], HLAR[3], and MAODV-SIM [4] set up primary and secondary paths on demand, and switch from the primary path to the secondary path when detecting link failures on the primary path. Those protocols, however, do not consider end-to-end reliability when establishing the route, and thus, the probability of route failure is high. For reliable communication, the message transmission path has to be the most reliable among the paths between a source and a destination, and has to be recovered quickly without re-establishing the route at the source in cases of broken links or nodes in the path.

We propose a new multipath-based reliable routing protocol on MANETs, called Multipath-based Reliable routing protocol with Fast-Recovery of failures (MRFR). In MRFR, to find the most reliable path between a source and a destination, each intermediate node broadcasts RREQ (Route REQuest) packets multiple times and updates its path information each time it broadcasts the RREQ packet again. If RREQ packets are broadcasted like this, too many RREQ packets can be generated during the path establishment. MRFR uses two timers, $\Delta delayRREQ$ and $\Delta delayDEST$, for each node to be able to collect all the RREQ packets and reduce the number of RREQ broadcast packets. The established path consists of a primary path, which is used to transmit messages, and the secondary paths, which are used to recover the path when detecting failures on the primary path. After establishing the path, the source transmits messages through the primary path. If a node detects a link failure during message transmission, it can recover the path locally by switching from the primary to the secondary path. By allowing the intermediate nodes to locally recover the route failure, the proposed protocol can handle the dynamic topological changes of MANETs efficiently. We evaluated the performance of the proposed protocol using the QualNet simulator, comparing it to other protocols in terms of end-to-end packet reception probability, end-to-end delay, and fault-tolerance capability. We also evaluated the impact of the $\Delta delayRREQ$ and $\Delta delayDEST$ delay times to the performance of the protocol such as the path quality, the path set-up time, and the number of generated RREQ packets. The simulation results show that the MRFR protocol has a higher end-to-end message delivery ratio and a higher fault-tolerance capability than other protocols.

The rest of the paper is organized as follows. Section 2 describes related research on multipath-based routing protocols for reliable message transmission on MANETs. In section 3, the operation of the proposed protocol, how to set up the route between a source and a destination and how to recover the route when detecting broken links, are described. Section 4 describes the performance evaluation of the proposed protocol using simulations. Finally, the conclusion of the paper is given in section 5.

2. Related Research

Many researchers have considered multipath-based routing for reliable message transmission in MANETs to provide many alternative paths in case of link or node failures [1][2][3][4][7][9][10][13][14]. Considering the dynamic topological changes of MANETs, multipath-based protocols set up the primary and secondary paths on demand, switching from the primary to the secondary path in case of link failures on the message transmission path. Node Disjoint Multipath Routing (NDMR) [2] is a DSR [5]-based multipath routing protocol that uses node-disjoint paths. NDMR uses path accumulation in RREQ packets as does the DSR routing protocol. Based on the path information collected in RREQ packets, the destination node selects the node-disjoint paths between the source and itself. NDMR selects the shortest path as the primary path, while the secondary path is selected as the shortest path among all available path remainings, which are disjoint paths with the primary path. Using the shortest path for data transmission may be a good choice for fast delivery, but cannot be a good choice for reliability due to frequent link failures, as addressed by Pham and Perreau [6]. Ad hoc On-demand Distance Vector routing – Multipath (AODVM) [7] implements modifications on top of AODV [8] to enable multiple node disjoint paths. AODVM uses a similar method as AODV to set up a route, except that only the destination node replies to the RREQ packets to ensure selection of node-disjoint paths. AODVM has the same problem for reliable message transmission as NDMR by selecting the primary and secondary paths based on the arrival time of RREQ packets. Selecting the node-disjoint paths can incur more overhead to a destination in order to check the disjointedness of the paths, and it is sometimes difficult to find disjoint paths if the network is not dense enough. If the node-disjoint paths are used, the intermediate nodes cannot recover the route locally when detecting a broken link, which will result in an increase in the fault recovery time. The CacHing And Multiple Path (CHAMP) routing protocol [9] uses a cooperative packet caching and multipath routing method to reduce packet loss due to frequent route breakage. In the CHAMP protocol, each node maintains a small buffer for caching recently forwarded data packets. When a downstream node encounters a forwarding error in transmitting a data packet, an upstream node, which has a copy of that packet in the buffer and an alternative route, can re-transmit that packet using the alternate route. When forwarding a data packet, a node chooses the next hop neighbor that is used the least number of times, which will spread data over many routes in a round-robin fashion. In CHAMP, each node must keep copies of data packets, which is a great burden on the nodes. MAODV-SIM [4] uses the Signal Intensity Metric (SIM) as a link quality estimator, and finds multiple paths called the emergency paths from a source to a destination. The SIM values of the links are calculated based on the Received Signal Strength Indicator (RSSI) values. Using RREQ flooding, multiple routes can be established between a source and a destination. For each path, MAODV-SIM finds the smallest SIM value among all of the links in the path, and chooses a path whose smallest SIM is the highest. The problem with MAODV-SIM is that the SIM is

not a good metric to measure the reliability of the paths, and the path with the smallest SIM value does not mean that it is the most reliable path. MultiPath Associativity Based Routing (MPABR) [10] uses an associativity tick as a link quality estimator, which is measured by exchanging hello messages between neighboring nodes. Each node maintains a list of current neighbors and an associativity tick count, denoting how many hello messages it has received from its neighbor. The associativity tick is a good estimator, since it reflects the real association statuses of the links over time. However, MPABR cannot always find the most reliable end-to-end route whose associativity tick value is the smallest since it is based on basic RREQ flooding. The Hybrid Location-based Ad hoc Routing protocol (HLAR) [3] uses the Expected Transmission Count (ETX) [11] as a link quality estimator. HLAR is a routing protocol based on AODV [8] and LAR [12]. In the route establishment phase, HLAR utilizes the location information to limit the number of RREQ packets generated during the search for the route. HLAR tries to find a reliable path between a source and a destination at the route establishment, and to reduce the number of control packets using location information. In cases of link failures, HLAR uses a local repair method of intermediate nodes by broadcasting the Route Repair Packet (RRP) to recover the route. In HLAR, it needs a location system like GPS in the nodes, and it is difficult to repair the route quickly after detecting link failures.

To handle link failures in the data transmission path, after detecting a link failure, most multipath-based routing protocols such as NDMR [2], AODVM [7], MP-MAODV [13], MAODV-SIM [4], and MSR [14], transmit a RERR (Route ERRor) packet to notify the source of the failure. Upon receiving the RERR packet, the source simply switches the data transmission path to one of the available secondary paths. By doing this, we can recover the path quickly and reduce the control overhead by eliminating RREQ flooding to find a new route. However, if we could recover the path locally at the intermediate nodes, it would be quicker to repair the path. HLAR allows intermediate nodes detecting a broken link to repair the data transmission path locally. When detecting a broken link toward a destination, an intermediate node consults its routing table to find a neighbor node that is closer to the destination, with routing information to the destination. If a closer neighbor is available, data packets are forwarded to that node after updating the routing table. Otherwise, the intermediate node broadcasts a RRP to find a new path to the destination. If an intermediate node fails to locally repair a broken link, it sends a RERR packet to the source node. This kind of local repair mechanism causes some amount of control traffic and takes time to recover the data transmission path due to control packet flooding in order to find a new path locally. Our approach provides a method for intermediate neighbors to find an alternative path to recover the broken path locally, without broadcasting control packets when detecting link failures.

3. MRFR – Multipath-based Routing Protocol with Fast Recovery of Failures on MANETs

This section describes the operation of the MRFR protocol proposed in this paper, which is a reliable routing protocol with fast recovery of failures on MANETs. The MRFR protocol uses ETX as a link cost metric and tries to transmit messages through the most reliable path between a source and a destination.

3.1 Link Quality Estimator - ETX

In the MRFR protocol, ETX [11] is used as a link quality estimator of MANETs. The ETX is a receiver-initiated estimator to estimate the quality of the link, which uses active monitoring. Each node broadcasts *Probe* packets periodically to calculate the ETX value of its links, and calculates the PRR of a link based on the number of *Probe* packets received successfully. ETX takes into account link asymmetry by estimating the uplink quality from a sender to a receiver, denoted as $PRR(x,y)_{forward}$, as well as the downlink quality from a receiver to a sender, denoted as $PRR(x,y)_{backward}$. $PRR(x,y)_{forward}$ is the PRR of the uplink calculated at the receiver, while $PRR(x,y)_{backward}$ is the PRR of the downlink calculated at the sender. A node *y* calculates $PRR(x,y)_{forward}$ based on the number of *Probe* packets received successfully from its neighbor *x*, and sends this value in its *Probe* packet by broadcasting to let their neighbors know this value. The ETX value of link *(x, y)* is calculated as:

$$ETX(x,y) = \frac{1}{\mathrm{PRR(x,y)_{forward}} \times \mathrm{PRR(x,y)_{backward}}} \quad (1)$$

Given a network with ETX values on the links, the end-to-end ETX of a path from a source node *S* to a destination node *D*, denoted as *e2e_ETX(S, D)*, is defined as follows:

$$e2e\_ETX(S,D) = \sum_{(x,y)\in path(S,D)} ETX(x,y) \quad (2)$$

where *path(S, D)* denotes a set of successive links in the path from node S to D such as: $path(S, D) = \{(S, X_1), (X_1, X_2), \ldots(X_{k-1}, X_k), (X_k, D)\}$. There are many paths between a source and a destination, and the path with the smaller *e2e_ETX* value represents the more reliable path. The MRFR protocol tries to find a path with the smallest *e2e_ETX* value among the paths between a source and a destination, transmitting messages through the path.

3.2 Route Discovery of MRFR

The MRFR protocol uses a multipath composed of a primary and a secondary path for message transmission, and finds a multipath before the message transmission by exchanging RREQ and RREP control packets between the source and destination. Each of the intermediate nodes maintains primary and secondary path information for the destination, and transmits the messages through the primary path, while changing from the primary to the secondary path in case of link failures on the primary path.

3.2.1 Propagation of a RREQ Packet

When a source node *S* wants to transmit messages to a destination node *D*, it tries to set up a path by broadcasting a RREQ packet. The RREQ packet carries (*id*, *etx*, *ttl*), where *id* denotes the ID of the path, *etx* denotes the sum of the ETX values of the links over which the RREQ packet has traversed, and *ttl* denotes the hop count value to limit the flooding area of the RREQ packet. Initially, source *S* broadcasts *RREQ[(S,D), 0, TTL]* to set a new path to the destination *D*. The RREQ packet is forwarded by intermediate nodes by rebroadcasting until they reach their destination, and the *etx* values of the links are added cumulatively to the *etx* file of the RREQ packet while being forwarded. Each node maintains a routing entry for the primary and secondary path for each path between a source and a destination: [*id*, *primaryF*, *primaryR*, *secondaryF*, *secondaryR*, ETX_{pri}, ETX_{sec}, *HC*]. *id* denotes the ID of the path,

primaryF and *primaryR* denote the forward and reverse nodes of the primary path, *secondaryF* and *secondaryR* denote the forward and reverse nodes of the secondary path, ETX_{pri} and ETX_{sec} denote the end-to-end ETX values of the primary and the secondary paths between the source and itself, and *HC* denotes the hop count from the source to itself in the primary path. If an intermediate node *B* receives *RREQ[id, etx, ttl]* from its neighbor *A*, it updates the *etx* and *ttl* values in the RREQ packet with $etx = etx + ETX(A,B)$ and $ttl = ttl - 1$, and performs the following:

(i) if it is the first RREQ packet received with path *id*, then node B creates a routing entry with [*id*, *primaryF=Null*, *primaryR=A*, *secondaryF=Null*, *secondaryR=Null*, $ETX_{pri}=etx$, $ETX_{sec}=\infty$, *HC=ttl*], rebroadcasts the updated packet, and sets *flag* = 0 and the timer Δ*delayRREQ* to collect multiple RREQ packets arriving during that time,

(ii) if a routing entry with path *id* exists, then node *B* performs the following:

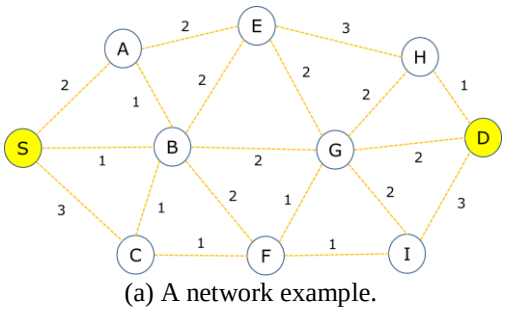
a) if $etx < ETX_{pri}$, then $secondaryR = primaryR$, $ETX_{sec} = ETX_{pri,}$ and $primaryR = A$, $ETX_{pri} = etx$, and $flag=1$,

b) if $ETX_{pri} \leq etx < ETX_{sec}$, then $secondaryR = A$, $ETX_{sec} = etx$,

c) otherwise, the RREQ packet is dropped,

(iii) if the Δ*delayRREQ* timer = 0 and *flag* = 1, then it rebroadcasts *RREQ[id, ETX_{pri}, HC]* and sets *flag* = 0.

This process continues until the RREQ packet arrives at the destination or the *ttl* value becomes 0. The MRFR protocol allows each node to broadcast the RREQ packet multiple times to obtain the primary and secondary paths with smaller *e2e_ETX* values from the source to the node. However, this can cause a large number of RREQ packets to be generated, so the MRFR protocol uses the Δ*delayRREQ* timer and *flag* to collect multiple RREQ packets for a path and to control the number of RREQ packets. The Δ*delayRREQ* time is the delay time for a node to collect all of the RREQ packets from its neighbors, and choose the best primary reverse node among them. A suitable value of Δ*delayRREQ* is necessary. If it is too small, then the intermediate node cannot receive all of the RREQ packets, if it is too high, then it will increase the path set-up time. We evaluated the impact of the Δ*delayRREQ* delay to the performance of MRFR such as the e2e_ETX of the path set up, the path establishment delay, and the number of RREQ packets generated and described in Section 4.



(a) A network example.

	S	A	B	C	E	F	G	H	I	D
primaryR	-	S	S	B	B	B	B	G	F	G
secondaryR	-	B	A	S	A	C	F	E	G	H
ETX_{pri}	-	2	1	2	3	3	3	5	4	5
ETX_{sec}	-	2	3	3	4	3	4	6	5	6

(b) Routing table entries of the intermediate nodes.

Fig. 1. Propagation of RREQ packets.

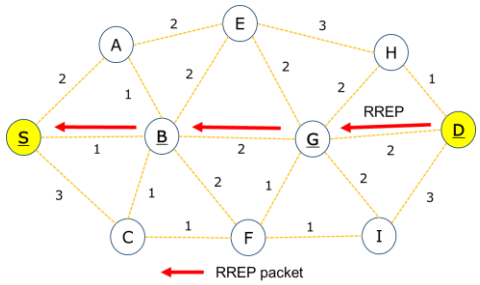
Let us consider the propagation of RREQ packets using the example in **Fig. 1**. In **Fig 1-(a)**, the number in each link is the ETX value of the link, and *S* and *D* are the source and destination. At first, source *S* broadcasts *RREQ[(S,D), 0, 5]* to find a path toward *D*. The RREQ packet is broadcasted again by the intermediate nodes until they reach the destination node, and the RREQ packets carry the e2e_ETX value of the path over which they traverse. Each intermediate node waits for the Δ*delayRREQ* time after receiving the first RREQ packet, and receives all of the RREQ packets arriving during that time. Each node selects the primary and secondary reverse nodes among its neighbors based on the *etx* values of the RREQ packets received from the neighbors. The neighbor which transmitted the RREQ packet with the smallest *etx* value is the primary reverse node, and the one which transmitted the RREQ packet with the second smallest *etx* value is the second reverse node. For example, node *G* in **Fig. 1** can receive the following RREQ packets from its neighbors:

from node *B* (through path *S-B-G*): *RREQ[(S,D), 3, 3]*,
from node *E* (through path *S-B-E-G*): *RREQ[(S,D), 5, 2]*,
from node *F* (through path *S-B-F-G*): *RREQ[(S,D), 4, 2]*,
from node *H* (through path *S-B-E-H-G*): *RREQ[(S,D), 7, 1]*,
from node *I* (through path *S-B- G-I-G*): *RREQ[(S,D), 7, 1]*.

Among them, node *G* chooses node *B* as the primary reverse node, and node *F* as the secondary reverse node, considering the *ETX* values of the RREQ packets. This process continues until the RREQ packets arrive at their destination. **Fig. 1-(b)** shows the reverse entries and the ETX values of the routing table entries in the intermediate nodes after the RREQ flooding is finished. *primaryR* and *secondaryR* denote the primary and secondary reverse nodes of each node, and ETX_{pri} and ETX_{sec} are the ETX values of the primary and secondary paths from the source to itself, respectively.

3.2.2 Establishment of the primary path

If the destination node *D* receives the first RREQ packet, then it sets the Δ*delayDEST* timer and waits for the RREQ packets arriving during that time. The Δ*delayDEST* time is the delay for the destination to collect enough number of the RREQ packets from its neighbors and is set to a little larger than Δ*delayRREQ*. We evaluated the impact of the Δ*delayDEST* delay to the performance of MRFR such as the *e2e_ETX* of the path set up, the path establishment delay, and the number of RREQ packets generated and described in Section 4. If the Δ*delayDEST* timer is expired, it selects its primary and secondary reverse nodes and transmits the *RREP[id]* packet to its primary reverse node. The *RREP* packet is transmitted through the primary reverse path that is set up during the propagation of *RREQ* packets. If an intermediate node receives the *RREP* packet, then it sets up its primary forward node, *primaryF*, for the path of *id*. This process is repeated until the *RREP* packet arrives at the source node. Through this process, the primary path from the source to the destination is set up. **Fig. 2** shows the primary path set-up in the example of the network in **Fig. 1**.



(a) Propagation of a RREP packet.

	S	A	B	C	E	F	G	H	I	D
primaryF	B	-	G	-	-	-	D	-	-	-
secondaryF	-	-	-	-	-	-	-	-	-	-
primaryR	-	S	S	B	B	B	B	G	F	G
secondaryR	-	B	A	S	A	C	F	E	G	H

(b) Routing table entries of the intermediate nodes.

Fig. 2. Estabishment of the primary path.

Each node in the primary path (node S, B, G, and D in **Fig. 2**) has a flag, *pri_flag*, which is set to 1, to denote it in the primary path.

3.2.3 Establishment of the secondary path

After setting up the primary path, the MRFR protocol tries to set up a secondary path which is used to recover the message transmission path in case of failures on the links on the primary path. To set up a secondary path, the destination node transmits the *RREP2[id]* packet to its secondary reverse node after a Δ*delayRREP* time from the time when it has transmitted the *RREP* packet. If the destination node has no secondary reverse node, then it transmits the *RREP2[id]* packet to its primary reverse node. The *RREP2* packet is forwarded through the intermediate nodes toward the source node. We have two options to set up the secondary path as follows.

[Option 1]

When a node *A* receives a *RREP2[id]* packet from its neighbor node *B*,

(i) if *flag_RREP2* == 1, then it sets *secondaryF* to *NULL* and forwards the *RREP2[id]* to its primary reverse node, and exits,

(ii) *flag_RREP2* = 1

(iii) if node *A* is not in the primary path (*pri_flag == 0*) of the path *id*, then

  a) it sets its primary forward node, *primaryF*, to *B*, and

  b) it forwards the *RREP2[id]* to its primary reverse node

(iv) if node *A* is in the primary path (*pri_flag == 1*) of the path *id*, then

  a) if *B* == *primaryF* of *A*, that is, link (*A*,*B*) is in the primary path, then it sets *secondaryF* to *NULL*, else it sets *secondaryF* to *B*, and

  b) it forwards the *RREP2[id]* to its secondary reverse node if *secondaryR != NULL*, otherwise it forwards the *RREP2[id]* to its primary reverse node,

(v) this process continues until the RREP2 packet arrives at the source node.

[Option 2]
When a node *A* receives a *RREP2[id]* packet from its neighbor node *B*,

(i) if node A is in the primary path (*pri_flag == 1*) of the path *id*, then

a) if *B* == *primaryF* of *A*, then exit,

b) otherwise, it sets its secondary forward node, *secondaryF*, to B, and transmits the *PRI_NOTIFY[id]* packet to node *B*,

(ii) if node A is not in the primary path (*pri_flag == 0*) of the path *id*, then

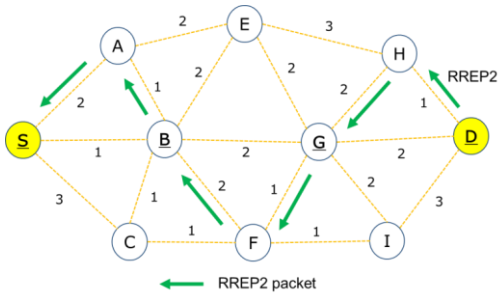
a) it sets its primary forward node, *primaryF*, to B, and forwards the *RREP2[id]* to its primary reverse node,

(iii) this process continues until the *RREP2* packet arrives at the source node.

When a node *A* receives the *PRI_NOTIFY[id]* packet, it transmits the *RREP2[id]* packet to its secondary reverse node if *secondaryR != NULL*,; otherwise, it stops transmitting the *RREP2* packet.

In Option 1, the *RREP2* packet is forwarded through intermediate nodes, including the primary nodes. In the case of the primary nodes, they forward the *RREP2* packet to its secondary reverse node, and the other nodes which are not in the primary path forward the *RREP2* packet to its primary reverse node. *flag_RREP2* is used to avoid forming a loop in the secondary path. When a primary node receives the *RREP2* packet, it sets *flag_RREP2* to 1, and if it receives the *RREP2* packet again, which is the case of forming a loop, it resets the secondary path. In the case of Option 2, the *RREP2* packet is forwarded only through intermediate nodes, except the primary nodes. If a primary node receives a *RREP2* packet, then it sets its secondary forward node and replies with a *PRI_NOTIFY* packet. If an intermediate node receives the *PRI_NOTIFY* packet, then it forwards the *RREP2* packet to its secondary reverse node. This process allows the intermediate nodes in the primary path to set up a node-disjoint secondary path from itself to the destination node.

Fig. 3 and Fig. **4** show examples of setting up the secondary path after setting up the primary path in the example of **Fig. 2,** according to Option1 and Option2, respectively.



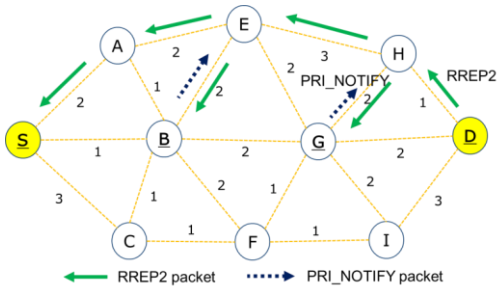
(a) Propagation of RREP2 packets.

	S	A	B	C	E	F	G	H	I	D
primaryF	B	B	G	-	-	G	D	D	-	-
secondaryF	A	-	F	-	-	-	H	-	-	-
primaryR	-	S	S	B	B	B	B	G	F	G
secondaryR	-	B	A	S	A	C	F	E	G	H

(b) Routing table entries of the intermediate nodes.

Fig. 3. Estabishment of the secondary path using Option 1.

Fig. 3 shows an example of setting up the secondary path using Option 1. While forwarding the RREP2 packets from the destination to the source, new nodes, *A* and *H* in **Fig. 3**, are included in the secondary path, and the secondary path (the secondary forward node) is added in nodes *S*, *B*, and *G* in the primary path. As the example in **Fig. 3** shows, the MRFR-option1 protocol provides many alternative routes to go around the link of the primary path in case of failures. For example, route *S-A-B* can be used to go around link *S-B*, route *B-F-G* can be used to go around link *B-G*, and route *G-H-D* can be used to go around link *G-D*.



(a) Propagation of RREP2 packets.

	S	A	B	C	E	F	G	H	I	D
primaryF	B	E	G	-	H	G	D	D	-	-
secondaryF	A	-	E	-	-	-	H	-	-	-
primaryR	-	S	S	B	B	B	B	G	F	G
secondaryR	-	B	A	S	A	C	F	E	G	H

(b) Routing table entries of the intermediate nodes.

Fig. 4. Estabishment of the secondary path using Option 2.

Fig. 4 shows an example of setting up the secondary path using Option 2. While forwarding the RREP2 packets from the destination to the source, new nodes *A*, *E*, and *H* in Fig. 4 are included in the secondary path, and the secondary path (the secondary forward node) is added in nodes *S*, *B*, and *G* in the primary path. As the example in **Fig. 4** shows, the MRFR-option2 protocol also provides many alternative routes to go around the link of the primary path in case of link failures. However, unlike Option 1, MRFR-option2 provides disjoint paths from each intermediate node to the destination node to go around the link of the primary path. For example, the route *S-A-E-H-D* can be used to go around link *S-B*, route *B-E-H-D* can be used to go around link *B-G*, and route *G-H-D* can be used to go around link *G-D*.

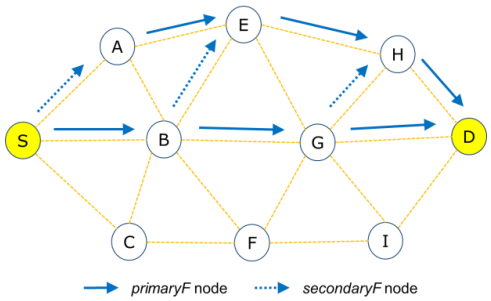
3.3 Route Maintenance: Handling Link Failures

Message transmission using MRFR is simple. After setting up the primary and secondary paths with *path_ID=id* to a destination node *D*, a source node *S* transmits messages to *D* through the primary path. All of the messages from *S* to *D* carry the *path_ID*. If a node receives a message to transmit with *path_ID=id*, it then finds a routing entry with *path_ID=id*, and transmits the message to its primary forward node. When node *A* on the primary path with *path_ID = id* detects a link failure on its primary forward node *B*, it performs the following process to recover the message transmission path:

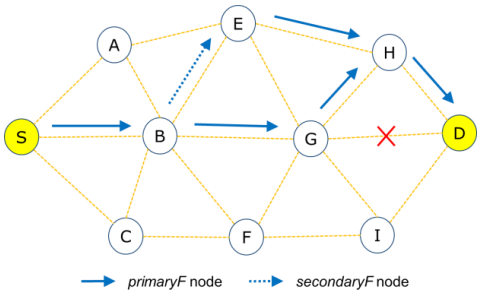
(i) If node *A* has a secondary path (*secondaryF* ≠ NULL) for the path *id*, then *primaryF = secondaryF* and *secondary = NULL*, that is, the secondary path becomes the primary path.

(ii) If node *A* does not have a secondary path (*secondaryF* == NULL) for the path *id*, then it transmits a *RERR[id]* (Route ERRor) packet to the primary reverse node. The *RERR[id]* packet is forwarded through the primary reverse path until it arrives at a node having a secondary path. If a node *B*, which has a secondary path for the path *id*, receives the *RERR[id]* packet, it recovers the message transmission path by changing the primary path to the secondary path, where *primaryF = secondaryF* and *secondary = NULL*.

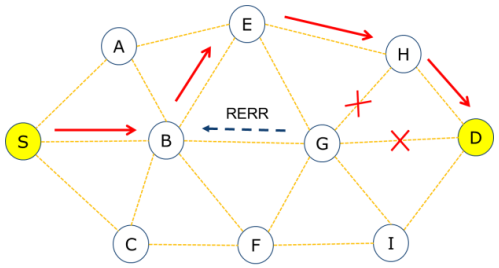
Let us look at an example of handling the link failures of the MRFR. **Fig. 5-(a)** shows the primary and secondary paths established between the source *S* and destination *D* using Option 2 of the MRFR (**Fig. 4**). After setting up the path, the messages sent by the source are transmitted through the primary path *S-B-G-D*. During message transmission, when link *G-D* is broken, node *G* detects the failure and recovers the path by changing its primary forward node from *D* to *H* (**Fig. 5-(b)**). If link *G-H* is broken in **Fig. 5(b)**, node *G* detects the failure and transmits a *RERR[id]* packet to its primary reverse node *B* because it has no secondary forward node. Node *B*, receiving the *RERR* packet, recovers the message transmission path by changing its primary forward node from *G* to *E* (**Fig. 5-(c)**). This process is repeated until there is no path remaining between *S* and *D*. If the source *S* receives the *RERR* packet but has no secondary forward node, then it tries to set up a new path to *D*.



(a) Path establishment between S and D.



(b) Path recovery after G-D link failure.



(c) Path recovery after G-H link failure.

Fig. 5. Handling link failures in MRFR-option2.

4. Performance Evaluation

The performance of the proposed protocol is evaluated by simulation using the *Qualnet* 5.0 simulator [15], and is compared with the *NDMR* [2], *MAODV-SIM* [4], and *HLAR* [3] protocols in terms of the packet delivery ratio, end-to-end delay, delay jitter, protocol overhead, and the recovery time of link failures. On a network having 1500 m × 1500 m dimensions with various numbers of nodes from 75 to 200, we created message flows from a random source to a random destination one by one, up to 10 flows. After setting up the message transmission path, the source of each flow transmits its messages periodically. In MANETs, each node can move. We used the random waypoint mobility model, and the minimum and maximum movement speeds were set from 0 to 25 m/s. We also evaluated the performance of the protocols depending on the different levels of message traffic: light traffic (1 flow), medium traffic (4-5 flows), and high traffic (7-10 flows). As we mentioned in Section 3, the Δ*delayRREQ* and Δ*delayDEST* delay times can affect the performance of the MRFR protocol. To get the proper values of Δ*delayRREQ* and Δ*delayDEST*, we have conducted the simulation to show the impact of Δ*delayRREQ* and Δ*delayDEST* to the performance of our protocol (shown in Section 4.4). Based on the simulation result, we set the Δ*delayRREQ* and Δ*delayDEST* delay times to 15ms and 25ms, respectively. The following **Table 1** shows the parameters used in our simulation.

To obtain $PRR(x,y)_{forward}$ and $PRR(x,y)_{backward}$, node x and y broadcast *Probe* packets periodically (period = 1s in our simulation), and calculate these values based on the number of *Probe* packets received successfully from its neighbor during a fixed interval w (w = 10 s in our simulation). Thus, $PRR(x,y)_{forward}$ at time t is calculated at node y as follows:

$$\mathrm{PRR(x,y)}_{forward} = \frac{count(t-w,t)}{N} \tag{3}$$

Here, N (=10) is the total number of *Probe* packets transmitted by node x during [t-w, t], and *count(t-w, t)* is the number of *Probe* packets received successfully from node x during [t-w, t]. Node y sends the $PRR(*,y)_{forward}$ values of its neighbors in its *Probe* packets to inform the neighbors of the value. Each node computes *ETX(x,y)* based on the equation in Section 3.1 using $PRR(x,y)_{backward}$ calculated by itself, and $PRR(x,y)_{forward}$ received from node y. We compared the performance of the *MRFR* with the *NDMR* [2], *MAODV-SIM* [4], and *HLAR* [3] protocols in terms of the packet delivery ratio, end-to-end delay and delay jitter, protocol overhead, and recovery time of link failures.

Table 1. Simulation parameters.

Simulation time	1000s
Dimension	1500m × 1500m
Transmission range	250m
Packet size	256B
Number of data packets transmitted after setting up the path	500
Data packet interval	1s
Time for collecting RREQs at intermediate node (ΔdelayRREQ)	15ms
Time for collecting RREQs at destination node (ΔdelayDEST)	25ms
Waiting time for transmitting RREP2 (ΔdelayRREP)	40ms
MAC protocol	802.11 DCF
Mobility pattern	Random way-point
Min/Max speed	0-25m/s

4.1 End-to-end packet delivery ratio

Fig. 6 shows the packet delivery ratios of the protocols in terms of the movement speed of the nodes from 0 to 25 m/s. It shows that as the speed of the node increases, the packet delivery ratio decreases. Overall, the *MRFR-2* protocol (Option 2 of *MRFR*) shows a higher packet delivery ratio than *HLAR*, *NDMR*, and *MAODV-SIM*. More specifically, in a worst case scenario (10 flows and node speed is 25 m/s), while *MRFR-2* maintains the packet delivery ratio at a value of 0.8, the *HLAR*, *NDMR* and *MAODV-SIM* protocols only maintain the values of 0.66, 0.64, 0.71, respectively. This result denotes that the *MRFR* protocol uses the more reliable path than other protocols to transmit the messages.

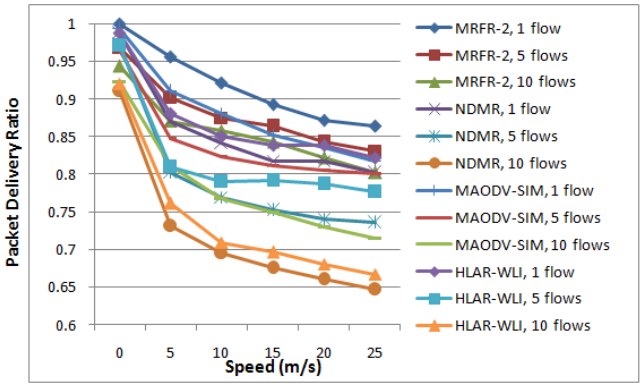
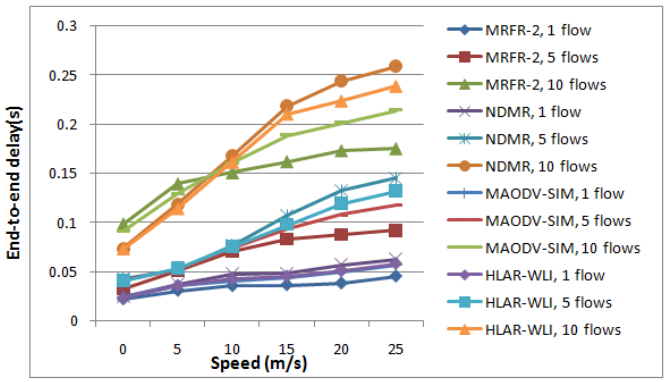


Fig. 6. End-to-end packet delivery ratio in terms of the speed of the node.

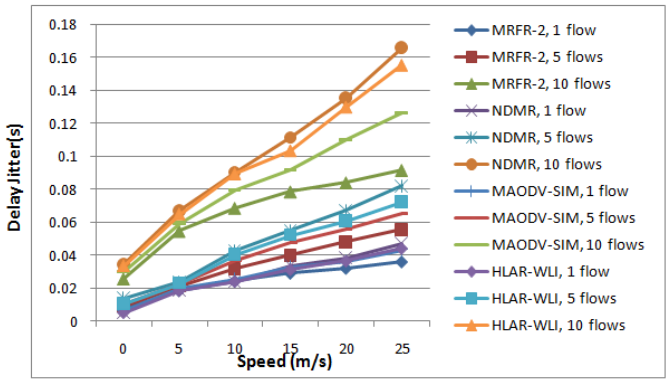
4.2 End-to-end delay and delay jitter of messages

Fig. 7 shows the average end-to-end delay and delay jitter of the messages according to the speed of the nodes from 0 to 25 m/s. When the node speed is small, the end-to-end delays and delay jitters of the compared protocols are almost the same. However, as the node speed increases, the average end-to-end delay of *MRFR-2* was a little smaller than the *NDMR*, *MAODV-SIM*, and *HLAR* protocols. The end-to-end delay of a message is the delay from the time when the source node transmits the message to the time when the destination node receives the message. If there is a link failure in the message transmission path when a

message is transmitted, the end-to-end delay of the message increases because the message can arrive at the destination after recovering the message transmission path. When the node mobility is small, the probability of link failures is also small. Thus, the end-to-end delay and delay jitter are shown to be almost the same in the compared protocols. However, if the node mobility increases, the probability of link failures becomes high, which results in an increase in the message delay and delay jitter. From the result shown in **Fig. 7**, we can see that the MRFR recovers the message transmission path fast in the case of link failures. We can also see the fast-recovery capability of the MRFR protocol in the simulation result of the recovery time and path lifetime shown in the next subsection.



(a) Average end-to-end delay



(b) Average delay jitter

Fig. 7. Average end-to-end delay and delay jitter of messages in terms of the speed of the node.

4.3 Fault-tolerance capability of the MRFR

To analyze the recovery capability of the failures, we compared the path recovery time and the path lifetime of the protocols. Most of the routing protocols on MANETs are reactive routing protocols, where a routing path is established between a source and a destination before message transmission, and the source tries to set up a new path if the path is broken. The path recovery time is defined as the time when the next message transmitted from the source arrives at the destination after a link or node failure happens in the message transmission path. Multipath-based routing protocols provide redundant paths to recover the

routing path without re-establishing the path in case of failures. However, if there is no available path in the routing path, the source tries to set up a new path by flooding with RREQ packets. We also defined the path lifetime as the time it takes for the source to set up the next path after establishing a routing path.

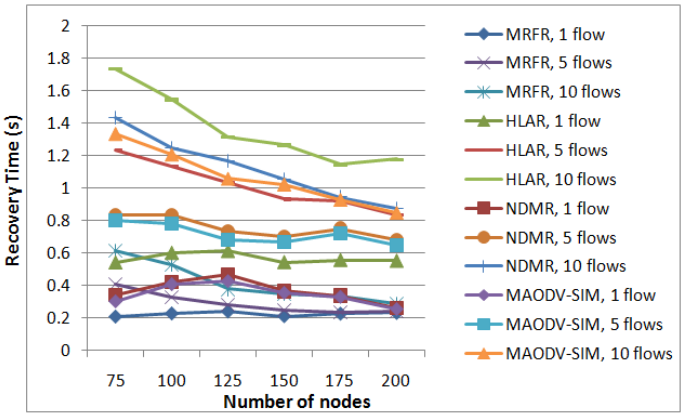


Fig. 8. The average path recovery time of the protocols.

To obtain the path recovery time of the protocols, after setting up a path in each protocol, we make some nodes in the primary path faulty one by one at specified times until the source cannot recover the path without reestablishing a new path. To make a fair comparison among the protocols, we set the location of a faulty node (the hop distance between a source and a faulty node) to be the same for all compared protocols. **Fig. 8** shows the average path recovery times of the compared protocols. As shown in the figure, the path recovery time of the MRFR (MRFR-option 2) is much smaller than other protocols. For example, in the case of 10 flows and 75 nodes, the average path recovery time of the MRFR is 604 ms, which is much smaller than that of the other protocols, 1,734 ms (HLAR), 1,437 ms (NMDR), and 1,345 ms (MAODV-SIM).

The multipath-based protocols can recover link or node failures locally without re-establishing the path by flooding RREQ packets at the source. The path lifetime denotes how long the established path can be valid by recovering the link or node failures locally. To obtain the path lifetime of the protocols, on a network with 100 nodes and with node speed from 5 m/s to 30 m/s, we set up a path between a given source and destination pair. Then we calculated the path lifetime, which is the period of time from the starting time of the path set-up to the time when the source tries to set up a new path by broadcasting a RREQ packet. **Fig. 9** shows the path lifetime of the protocols in terms of the mobility speed of the nodes. As the mobility speed of the nodes goes up, the probability of link failure will increase, which will affect the path lifetime. As we can see in the figure, the path lifetime decreases as the movement speed of the nodes increases for each protocol. For a given node mobility speed, however, the path lifetimes of our protocols (MRFR-1 and MRFR-2) show higher path lifetimes than those of the protocols we compared.

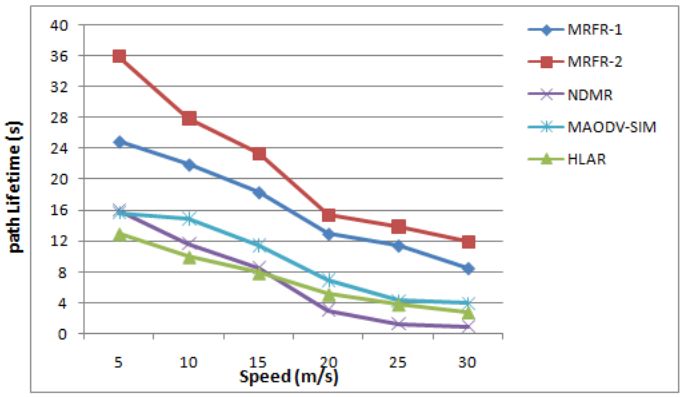


Fig. 9. The path lifetimes of the protocols in terms of node speed.

4.4 Impact of Δ*delayRREQ* and Δ*delayDEST* parameters to the performance of MRFR

In this part, we conducted experiments on 125-nodes network with 5-flows data transmission to evaluate the impact of Δ*delayRREQ* and Δ*delayDEST* parameters to the route quality (*e2e_ETX* of the route), route establishment delay, the number of RREQ control packets broadcasted and. The result is calculated on average for 5-flows. In MRFR, after broadcasting the firstly arrived RREQ packet, each node collects the RREQ packets during the ΔdelayRREQ time and selects the best route from the source to itself. By doing this, the MRFR can reduce the number of RREQ packets generated during the route establishment. The Δ*delayRREQ* delay can affect the established route quality (*e2e_ETX* of the route), the route establishement delay, and the number of generated RREQ packets. If the Δ*delayRREQ* delay is large, we can set up a path with small e2e_ETX and reduce the number of RREQ packets, but the route establishment delay will be increased. Otherwise, the reverse phenomenon will happen. To get a proper value of Δ*delayRREQ*, we performed the following simulation.

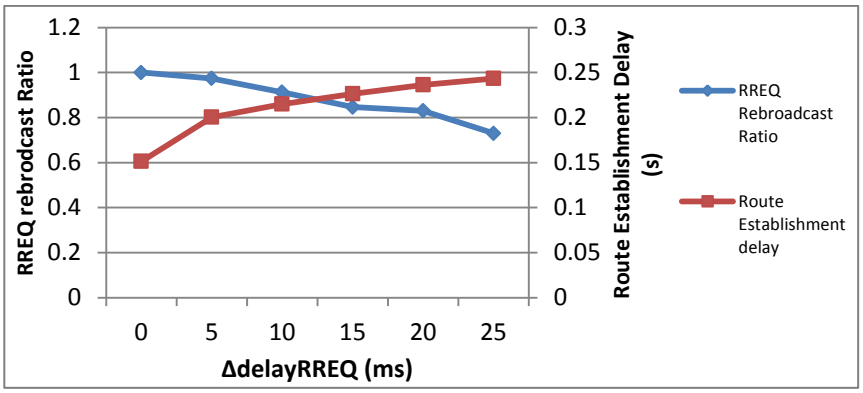


Fig. 10. The impact of Δ*delayRREQ*.

In a network with 125 nodes and 5 flows, we measured the route establishment delay and the RREQ rebroadcasting ratio while the value of Δ*delayDEST* is fixed to 25ms and the value of

Δ*delayRREQ* is changed from 0 ms to 25ms. The RREQ rebroadcasting ratio is calculated as the ratio between the number of RREQ packets rebroadcasted and the total number of RREQ packets that are arrived with a smaller ETX value than the previous ones. **Fig. 10** shows the impact of Δ*delayRREQ* parameter to the MRFR performance in terms of the primary path establishment delay and the number of RREQ packets broadcasted. As showed in **Fig. 10**, when we set Δ*delayRREQ* to 0, which means that every RREQ with a smaller ETX value will be broadcasted again immediately, therefore the rebroadcasting ratio is equal to 1. When we increase Δ*delayRREQ*, the RREQ rebroadcasting ratio has been decreased, while the route establishment delay was increased.

Fig. 11 shows the impact of Δ*delayDEST* parameter to the primary path quality and the route establishment delay. In this test, we fix Δ*delayRREQ* to 5ms and change Δ*delayDEST* from 0ms to 30ms. In MRFR, when the destination node receives the first RREQ, it sets the Δ*delayDEST* time to wait for collecting multiple RREQ packets during that time. After this time is expired, it chooses the best primary and secondary path and replies to the source. **Fig. 11** shows that when we vary this parameter from 0ms to 30ms, the route establishment delay is increased from 0.13 seconds to 0.21 seconds. On the other hand, the quality of the established route becomes better (smaller *e2e_ETX* value) as the Δ*delayDEST* increases. However, after 15ms (10ms more than Δ*delayRREQ* delay), the *e2e_ETX* almost remains the same even though we increase the Δ*delayDEST* more. From this result, we can see that 10 ms more than the Δ*delayRREQ* time would be a proper value for the Δ*delayDEST* time.

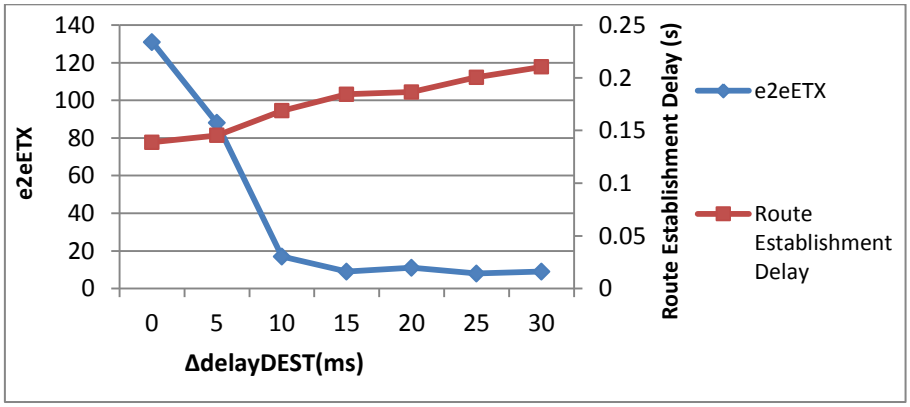


Fig. 11. The impact of Δ*delayDEST*.

5. Conclusion

In this paper, we proposed a multipath-based reliable routing protocol with fast-recovery of failures on MANETs. For reliable message transmission, the proposed protocol sets up a multipath considering the end-to-end packet reception reliability, which consists of a primary path and additional paths which can be used for recovering link or node failures. After setting up a path, the source transmits its data through the primary path, while additional paths are used to recover the packet transmission path without re-establishing the path by flooding RREQ packets at the source. The performance of the proposed protocol is evaluated using the QualNet simulator, and compared with the NDMR, HLAR, and MAODV-SIM protocols. The performance was measured in terms of the end-to-end packet reception probability, end-to-end delay and delay jitter of packets, path recovery time, and path

lifetime. We also experimented the impact of the Δ*delayRREQ* and Δ*delayDEST* delay times to the performance of our protocol in terms of the route establishment delay, number of control packets, and route quality. Simulations demonstrated that the proposed protocol has higher packet reception probability than other protocols. In terms of its fault-tolerance capability, the protocol has shown a smaller path recovery time and longer path lifetime than other protocols to which it was compared.

References

[1] Aristotelis Tsirigos and Zygmunt J. Haas, "Multipath routing in the presence of frequent topological changes," *IEEE Communication Magazine*, pp. 132–138, Nov. 2001. Article (CrossRef Link)

[2] Xuefei Li and Laurie Cuthbert, "Stable node-disjoint multipath routing with low overhead in mobile ad hoc network," *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 84–191, Oct. 2004. Article (CrossRef Link)

[3] Al-Rabayah and Robert Malaney, "A new hybrid location-based ad hoc routing protocol," *Global Communications Conference, Exhibition & Industry Forum*, pp. 1–6, Dec. 2010. Article (CrossRef Link)

[4] Bastien Mainaud, Mariem Zekri and Hossam Afifi, "Improving routing reliability on wireless sensors network with emergency paths," *Proc. of 28th International Conference on Distributed Computing Systems Workshops,* pp.545–550, Jun. 2008. Article (CrossRef Link)

[5] David B. Johnson, David A. Maltz and Josh Broch, "The Dynamic Source Routing (DSR) protocol for mobile ad hoc networks for IPv4," *IETF Internet RFC 4728*, Feb. 2007. Article (CrossRef Link)

[6] Peter P. Pham and Sylvie Perreau, "Performance analysis of reactive shortest path and multipath routing mechanism with load balance," *International Conference on Computer Communications*, pp. 251−259, Mar. 2003. Article (CrossRef Link)

[7] Zhenqiang Ye, Srikanth V. Krishnamurthy and Satish K. Tripathi, "A framework for reliable routing in mobile ad hoc networks," *INFOCOM 2003, Joint International Conference on Computer Communications,* pp. 270−280, 2003. Article (CrossRef Link)

[8] Charles E. Perkins, Elizabeth M. Belding-Royer and Samir R.Das, "Ad Hoc On-Demand Distance Vector (AODV) Routing,", *IETF Internet RFC 3561*, Jul. 2003. Article (CrossRef Link)

[9] Alvin C. Valera, Winston K.G. Seah and S.V. Rao, "Improving protocol robustness in ad hoc networks through cooperative packet caching and shortest multipath routing," *IEEE Transactions on Mobile Computing,* vol. 4, no. 5, pp.443−457, September, 2005. Article (CrossRef Link)

[10] Patrick McCarthy and Dan Grigoras, "Multipath associativity based routing," *Second Annual Conference on Wireless On-demand Network Systems and Services*, pp. 60−99, Jan. 2005. Article (CrossRef Link)

[11] Nouha Baccour, Anis Koubaa, Maissa Ben Jamaa, Habib Youssef, Marco Zuniga and Mario Alves, "A comparative simulation study of link quality estimator in wireless sensor network," *IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 1–10, September, 2009. Article (CrossRef Link)

[12] Young-Bae Ko and Nitin H. Vaidya, "Location-Aided Routing (LAR) in mobile ad hoc networks," *ACM Wireless Networks Journal*, vol. 6, no. 4, pp. 307–321, 2000. Article (CrossRef Link)

[13] Hong Tang, Fei Xue and Peng Huang, "MP-MAODV: A MAODV-Based Multipath Routing Algorithm," *Proc. of IFIP International Conference on Network and Parallel Computing*, pp. 296–301, 2008. Article (CrossRef Link)

[14] Lianfang Zhang, Zenghua Zhao, Yantai Shu, Lei Wang and Oliver W.W. Yang, "Load balancing of multipath source routing in ad hoc Networks," *International Conference on Computer Communications*, pp. 3197–3201, 2002. Article (CrossRef Link)

[15] Qualnet simulator, http://www.scalable-networks.com/content/.