PAPER

MESH: Multi-Eye Spiral-Hopping Routing Protocol in a Wireless Ad Hoc Network*

Yuh-Shyan CHEN† *and* **Kuan-Chang LAI**†, ***Nonmembers***

SUMMARY A mobile ad hoc network (MANET) is a characterized by multi-hop wireless links, absence of any cellular infrastructure, and frequent host mobility. Existing MANET routing protocols are divided into location-aware and non-location-aware routing protocols. In a location-aware routing protocol, location information can be exploited to facilitate routing. Our protocol, namely multi-eye spiral-hopping (MESH) routing protocol, is a location-aware routing protocol. Most promising routing protocols are constructed by the route-discovery, route-reply, and route-maintenance phases. Our MESH protocol utilizes the location-information to confine the blind-flooding region in the route-discovery phase, minimize route-reply packets in the route-reply phase, and promote the routing robustness in the route-maintenance phase. Two major contributions of this paper are introduced: (1) a *multi-eye* scheme is presented to confine route-discovery region for reducing redundant packets, and (2) a special multi-path scheme, called as *spiral-hopping* scheme, is introduced to provide on-line route-recovery capability. Extensive simulations are conducted to evaluate the protocol.

key words: *location-aware routing, mobile ad hoc network, mobile computing, routing, wireless network*

1. Introduction

Mobile ad-hoc networks (MANET) [5] consists of wireless hosts which communicate with each other, in the absence of a fixed infrastructure. Due to considerations such as radio power limitations, power consumption, and channel utilization, a mobile host may not be able to communicate directly with other hosts in a single-hop fashion. A multi-hop scenario occurs, where the packets sent by the source host are retransmitted by several intermediate hosts before reaching the destination host. In MANET, host mobility can cause frequent unpredictable topology changes, thus design of MANET routing protocol is more complicated than traditional network because it needs to have strong fault-tolerant capability.

Existing MANET routing protocols are classified into location-aware and non-location-aware as shown in Table 1, based on whether with the assistance of GPS (the location-information provider) or without. In location-aware routing protocol [6], it is assumed that a mobile host knows its current physical location, and thus such location information can be exploited to facilitate routing [8]. Basically, the location-aware routing protocol is more efficiently because the protocol fully utilizes location information to confine route-recovery region and reduce the redundant packets. Existing non-location-aware routing protocol consists of SSA [2], DSR [5], ABR [11], Multipath [9], Fisheye [3], and Query Localization [1] protocols, ···, etc. On the contrary, existing location-aware routing protocol, as we can know up to this moment, includes GRID [8], Peer-to-Peer [4], and LAR [7] protocols.

An efficient MANET routing protocol is mainly constructed by efficiency in route-discovery, route-reply and route-maintenance phases. Table 1 shows that, non-location-aware routing protocol, such as DSR [5] and Multipath [9] protocols, performs traditional blind-flooding operation in the route-discovery phase. This offers redundant route-discovery packets in the MANET. Two kinds of non-location-aware routing protocols, DSR [5] and Multipath [9], are compared in this paper. The reason is stated as follows. First, DSR protocol is used as a fundamental operation of our protocol. Table 1 shows that only the Multipath [9] protocol supports on-line route-recovery capability in all existing routing protocols. On-line route-recovery capability is achieved by maintaining alternative backup paths. Secondly, DSR [5] protocol is an on-demand routing; a mobile host initiates a route-discovery phase and then performs a route-maintenance phase if the path is established. Observe that, DSR can find alternate paths since possible alternate paths are cached temporarily in neighboring nodes. We observe that DSR [5] protocol has following shortcomings: (1) DSR is an off-line recovery protocol. If there is a failure link, the data transmission is broken before finding a new transmission path, (2) a large amount of memory cost is required to keep the whole information of the path, (3) DSR is not the hop-by-hop routing. More recently, Multipath protocol is proposed [9] with the on-line route-recovery capability. This work is done by maintaining disjoint paths from destination node to every node of primary path. If a link is failed, a backup path is replaced such that there is no route-recovery waiting time. This protocol has high fault-tolerant capability, however, main shortcoming is that destination

Manuscript received July 31, 2000.

Manuscript revised December 25, 2000.

†The authors are with the Department of Statistics, National Taipei University, Taipei, Taiwan, Republic of China.

*This work was supported by learning technology, sponsored by Ministry of Education: 89-H-FA07-1-4 and a preliminary version of this paper is presented at ICCCN'00: IEEE International Conference on Computer Communication and Network, Las Vegas, NV, Oct. 2000.

Table 1 Comparison table for existing MANET routing protocols.

	Protocol	Route-Discovery	Route-Reply	On-Line Route-Recovery
Non-Location-Aware Protocol	DSR [5]	×	×	×
	MultiPath [9]	×	×	$\vee$
Location-Aware Protocol	Peer-to-Peer [4]	$\vee$	$\vee$	×
	LAR [7]	$\vee$	$\vee$	×
	GRID [8]	$\vee$	$\vee$	×
	Ours (MESH)	$\vee$	$\vee$	$\vee$

node provides too much alternate paths, and greatly degrade network bandwidth.

Existing location-aware routing protocol consists of LAR [7], GRID [8] and GRID [8] protocols. The main contribution of LAR [7] is to propose a request-zone concept based on assumption of knowing destination node's location information. This approach can reduce redundant packets, but some disadvantages of LAR are displayed: (1) The request-zone is determined only by source and destination nodes. So it maybe leads to produce a narrow-rectangle request zone so difficult as to find a possible path. (2) LAR possibly finds a non-shortest path. (3) When a link is failed, a route-discovery process is needed to be re-performed.

More recently, a fully location-aware protocol, GRID [8] protocol, is presented to divide the MANET into grids in which each grid has a leader and only the grid leader is responsible for route discovery. The number of packets related to route discovery is insensitive to the density of mobile hosts in the searched area. Existing protocols incur more control packets as the host density increases. The robustness of path in GRID is strong and it requires less route discovery packets, however, the cost to maintain the grid leader is high. From Table 1, GRID [8] and Peer-to-Peer [4] are cluster-based routing protocol. Our scheme can be extended to be cluster-based routing protocol to possess cluster-based advantage. This paper is going to investigate only the effect of non-cluster-based routing protocol.

The main objective investigated in this paper is to develop an efficient location-aware routing protocol with on-line route-recovery capability. The motivation of this paper is to reduce the redundant packets, degrade the packet congestion, and buildup the route robustness. The work of finding accurate request-zone is facilitated by eye-nodes' location information. In this paper, we propose a multi-eye spiral-hopping (MESH) strategy with two major purposes.

- **Confine the blind flooding:** This task is completed by proposed multi-eye strategy.
- **Promote the route robustness:** This task is achieved by proposed spiral-hopping strategy.

Therefore, two brief contributions of this paper are introduced: (1) a *multi-eye* scheme is proposed to accurately confine the route-discovery region for reducing route-discovery and amount of route-reply packets, and (2) a special multi-path scheme, called as *spiral-hopping* scheme, is presented to provide on-line route-maintenance capability. Our MESH protocol fully utilizes the location-information to avoid the blind-flooding region, minimize route-reply packets, and strengthen the routing robustness in the route-discovery, route-reply, and route-maintenance phases, respectively. Observe that our MESH strategy can be applied to other existing MANET routing protocols. Performance analysis illustrates that our proposed scheme outperforms existing location-aware routing schemes in MANET.

The rest of the paper is organized as follows. Section 2 presents basic idea of our protocol. Section 3 presents our routing protocol. Section 4 illustrates performance analysis. And, Sect. 5 concludes this paper.

2. Basic Idea

We begin this section by defining the ad hoc wireless network [5]. The ad hoc wireless network is modeled as an undirected graph $G = (V, E)$, where V is a set of $|V|$ nodes and E is a set of $|E|$ undirected links connecting nodes in V. Each node has a unique identifier and represents a mobile host with a wireless communication device with transmission range R, and an infinity storage space. Nodes may move around and change their speed and direction independently. An undirected link (i, j) connecting two nodes i and j is formed when the distance between i and j become less than or equal to R. Link (i, j) is removed from E when node i and j move apart, and out of their transmission ranges. Our MESH (<u>M</u>ulti-<u>E</u>ye <u>S</u>piral-<u>H</u>opping) protocol is formed by *multi-eye* and *spiral-hopping* schemes. In the following, we introduce main idea of multi-eye and spiral-hopping schemes.

2.1 Multi-Eye Scheme

The main purpose of multi-eye scheme is to determine an accurate request-zone for flooding the route-discovery packets. In LAR [7], the request-zone is determined by the source and destination nodes. In our multi-eye scheme, the request-zone is determined not only by the source and destination nodes but also by eye-nodes. The key technique of multi-eye scheme is to extract location-informations from multiple *eye-nodes*. The advantage of exploiting accurate request-zone is to reduce the useless blind-flooding packets.

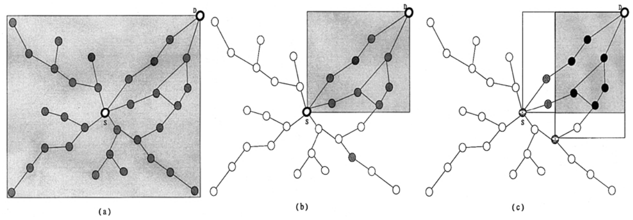


Fig. 1 Identifying the intersectant request-zone from eye-nodes.

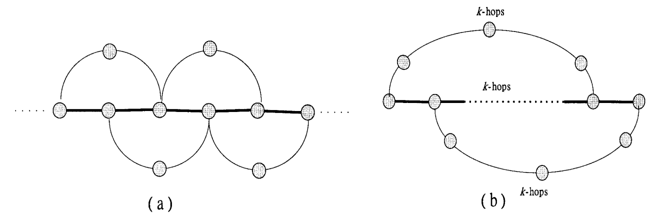


Fig. 2 An example of spiral-hopping paths (a) P_2 and (b) P_k.

Let an ad hoc network is the collection of mobile hosts/nodes, a mobile host/node is defined as *eye-node* if this mobile node is only forwarding its own location information (contains *location*, *direction*, and *speed*) and received information from other nodes. If a node receives a location information from an eye-node, this node acquires a different view of geographical information. Using geographical information allows us to determine an accurate request-zone. Attempt will be made to reduce the amount of needless packets and improve the bandwidth utilization.

Our MESH protocol acquires location information from eye-nodes and exploits an accurate request-zone. The function of eye-node is to provide destination node's location information from different eye-nodes to source. Relative geographical information will be used to estimate valuable request-zone. Ordinarily, the smaller request-zone is, the lower number of blind-flooding packets will be. As shown in Fig. 1(a), existing non-location-aware routing protocol blindly floods the route-discovery packets in MANET. This possibly leads to the 'broadcast storm' problem† seriously. On the contrary, LAR scheme, a location-aware routing protocol, can reduce the request-zone and degrade the 'broadcast storm' problem, which compared in Fig. 1(b). The request-zone of LAR scheme is determined by source and destination nodes. In our scheme, efforts will be made to find an accurate request-zone, as shown in Fig. 1(c), which is determined by many nodes.

To clarify between cluster-head and eye-node, a comparison was stated as follows. Initially, cluster-head must keep routing table but eye-node does not maintain routing table. When a cluster-head moves over the cluster, a cluster-head contention algorithm is applied to elect a new one. Therefore, there do not need the contention procedure for eye-nodes. Observe that only one cluster-head is in a cluster, but there may exist multiple eye-nodes in a cluster. Each eye-node can be seen as a visitor.

2.2 Spiral-Hopping Scheme

An interesting approach will be exploited in this paper is to identify the spiral-hopping path. The major objective of spiral-hopping scheme is to provide the on-line route-recovery capability. Among all the existing MANET routing protocols, only the Multipath [9] routing protocol supports this function, however, Multipath routing protocol incurs high network bandwidth. Therefore, it is desirable to provide a routing scheme with the on-line route-recovery capability under better bandwidth utilization.

We now formally define a generalized special multipath, the k-spiral-hopping path denoted by P_k. Given a path P, every node of path P connects to the next k-hop node of path P by a disjoint subpath, while the disjoint subpath's length equals to k. All disjoint subpaths and original path P are denoted as k-spiral-hopping path P_k. For example, if $k = 2$, then 2-spiral-hopping path P_2 is constructed as shown in Fig. 2(a). Observe that this paper focuses on discussing the effect of spiral-hopping path P_2. Furthermore, this idea can easily construct $P_k, k > 2$, to acquire high link stability, as shown in Fig. 2(b). The link stability of spiral-hopping path is achieved by maintaining backup links. If a link is failed, the failed link can be replaced immediately by the backup link. Generally speaking, the larger value of k is, the more node/link stability will be. On the contrary, the larger value of k is, the more network overhead will be incurred. Attempts to counter the trade-off between node/link stability and network overhead will be made.

To improve the node/link stability, a ping-pong effect is also considered for the node-electing scheme of our searching path. Considering a node as a candidate node of a searching-route path, the criteria of node-electing scheme is based on the node/link stability. Generally, the lower speed is, the higher node/link stability will be. However, if a node is with ping-pong moving pattern (lingering in a fixed position), then this node still preserves high node/link stability even if its speed is high. A ping-pong effect is shown as an example in Fig. 3, node 5 is with ping-pong moving pattern and higher speed than node 3, so we should choose node 5 as the candidate node in our searching path. Nodes 3, 4, 5, and 6 are said as gateway nodes and the intersectant area of transmission range of nodes 1 and 2 are

†When searching for a route, typically a route discovery packet will be sent. Every host in the searched area has the obligation to to rebroadcast the packet. In [10], it is shown that serious redundancy, contention, and collision will be incurred in a MANET with such broadcasting.

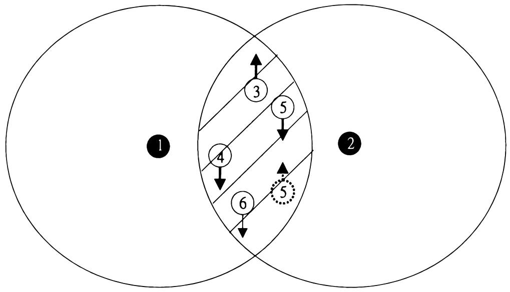


Fig. 3 An example of ping-pong effect.

also denoted as gateway area.

3. Our MESH Routing Protocol

In this section, the data structure of routing table and beacon packet are initially introduced. A MANET routing protocol is divided into, (1) route-discovery phase, (2) route-reply phase, and (3) route-maintenance phase. Each phase of MESH protocol is described below.

3.1 Routing Table and Beacon Packet

Routing table exists in every mobile host to record routing information. The beacon message is used to dynamically preserve the stable structure of spiral-hopping path. Our MESH scheme is a hop-by-hop routing protocol, so next-hop information is kept in routing table. This information is updated only if a link is failure. We now formally define the data structure of routing table below.

- **RoutingTable.**_Destination_ field records the destination node's IP address, which is filled in the route-reply phase.
- **RoutingTable.**_PrimaryNextHop_ field records the next-hop information for the primary path.
- **RoutingTable.**_BackupNextHop_ field records the next-hop information for the backup path.
- **RoutingTable.**$Click_{i,j}$ field is used to record stability status for each gateway node.
- **RoutingTable.**$TwoHopPath_{i,j}$ field records the two-hop path.
- **RoutingTable.**_BranchNode_ field is TRUE indicated this node is a branch node.

Note that the fields _Destination, PrimaryNextHop,_ and _BackupNextHop_ are all respectively identified in the route-reply phase (as described in Sect. 3.3). In addition, $Click_{i,j}$, $TwoHopPath_{i,j}$, and _BranchNode_ fields are maintained in the route-discovery phase (described in Sect. 3.2.2) to reflect the connecting status of neighboring nodes. The connecting status of neighboring nodes is maintained by using beacon packet. Let beacon packet be denoted as **Beacon**(_HopNumber, NodeList_), where _HopNumber_ is life time of the beacon packet, and _NodeList_, $|NodeList| \leq HopNumber$, is a node list which represents the traversal path of the beacon packet. We always set _HopNumber_ to be 2, since we investigate the effect of constructing P_2 in this paper. Further, we can construct a P_k by setting _HopNumber_ be k. As a comment, beacon packet helps us to construct a stable spiral-hopping path.

3.2 Route-Discovery Phase

Our MESH protocol has two major important features. One feature is to identify an intersection area from various request-zone. The second feature is to construct our spiral-hopping path in the intersectant request-zones. This work is done by three steps.

- **Step 1:** _Exploiting Intersectant-Request-Zone Operation:_ An intersectant request-zone is identified to reduce the total amount of route-discovery packets.
- **Step 2:** _Probing Operation:_ A probing operation provides information to maintain a stable spiral-hopping path.
- **Step 3:** _Route-Discovery Operation:_ A DSR-like route-discovery operation builds a spiral-hopping path on the identified intersectant-request-zone.

3.2.1 Step 1: Exploiting Intersectant-Request-Zone Operation

This section identifies an intersectant request-zone by utilizing the eye-nodes' location informations. As mentioned in Sect. 2.1, the smaller request-zone can obviously reduces the amount of route-discovery packets. However, the smaller request-zone must be supported by the correct information. Basically, a smaller request-zone supports limitative region, but is not guaranteed to finding a feasible path. Efforts will be made to counter the trade-off of exploiting smaller request-zone and finding a feasible path.

Let **EyeInfo(**_Destination, EyeNode_**)** packet be denoted as packet recording relative location information from eye-node, where _Destination_ and _EyeNode_ keep the location information of destination node and eye-node, respectively. The recognized intersectant-request-zone operation is derived below.

S1. Each possible destination node periodically floods a **EyeInfor(**_Destination,_ NULL**)** packet in MANET in which _Destination_ is the destination node's location information.

S2. If an eye-node received a **EyeInfo(**_Destination,_ NULL**)** packet, then added current eye-node's location information into **EyeInfo** and then reforward **EyeInfo(**_Destination, EyeNode)_ packet into MANET.

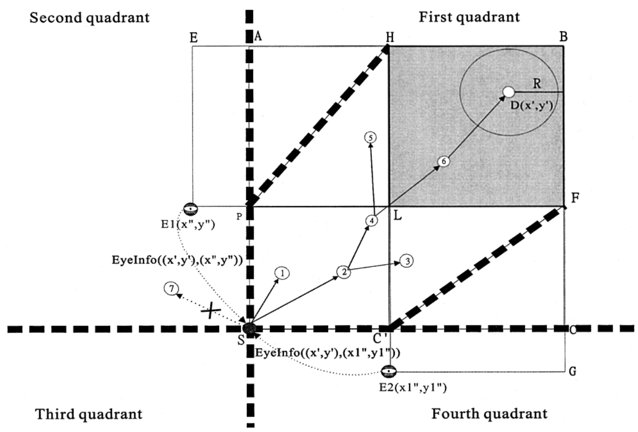


Fig. 4 Identifying the intersectant request-zone from eye-nodes.

S3. Assume a source node's location be (0,0) (here we use xy-coordinate to facilitate our presentation; in fact, device such as GPS receivers can provide 3-D locations in logitude, latitute, and altitude). Each **EyeInfo(***Destination, EyeNode)* packet can determine a request-zone by LAR scheme [7]. Source node receiving two or more **EyeInfor** packets from second and fourth quadrants (as shown in Fig. 4), then an intersectant request-zone is recognized by intersectant area of request-zones. Note that only packets from second and fourth quadrants are adopted to identify a smaller intersectant request-zone. The smaller intersectant request-zone reduces blind-flooding area of route-discovery packets. Only packet in request-zone is allowed to re-forward.

For instance as shown in Fig. 4, let source and destination nodes be denoted as S and D, where D located in $(x\prime, y\prime)$. Assume that there are two eye-nodes E_1 and E_2 located in (x'', y'') (in second quadrant) and (x_1'', y_1'') (in fourth quadrant), respectively. When E_1 and E_2 received node D's $(x\prime, y\prime)$, **EyeInfo**$((x\prime, y\prime),(x'', y''))$ and **EyeInfo**$((x\prime, y\prime),(x_1'', y_1''))$ are reforwarded to source node S. From nodes E_1 and E_2, the request zones are rectangles (E_1, F, B, E) and (E_2, G, B, H). It is worth mentioning that our route-discovery phase can be considered as using multiple times of LAR-scheme. The intersectant region of rectangles (E_1, F, B, E) and (E_2, G, B, H) is (B, H, L, F). Observe that intersection area (B, H, L, F) is smaller than the original request zones (E_1, F, B, E) and (E_2, G, B, H). Keep in mind, only packet in (B, H, L, F) is valuable for the searching path. However, we still need to construct a path from source to the region (B, H, L, F). Therefore, a possible request-zone is suggested in Fig. 4. Of course, there may exist other request-zones only if we can connect to region (B, H, L, F). Observe that our scheme allows us to reduce route-discovery packets in rectangles (A, H, L, P) and (L, F, C, C'), compared to LAR scheme.

3.2.2 Step 2: Probing Operation

The purpose of a probing operation is to maintain the mobility status of neighboring nodes. This work is achieved by maintaining a *branch node*. A node is said as *branch node* if there at least exist two disjoint paths from same node to the branch node, and a node is said as *gateway node* if this node can communicate with two non-neighboring nodes. The major work of the probing operation is to distributively identify the branch node in MANET. In addition, a probing strategy is proposed to identify the branch node and further to eliminate the ping-pong effect.

To make ping-pong effect into account, some extra information is kept in the routing table. Consider a node N, assume that there are m two-hop non-neighboring nodes $N_{\alpha=1\ldots m}$. Consider that there are n gateway nodes between nodes N and node N_α. For simplicity, we use gateway node G_β to denote one of n gateway nodes, where $1 \leq \beta \leq n$. Therefore, the counter $Click_{\alpha,\beta}$ is maintained in nodes N and N_α for the gateway node G_β, which reflects the connecting status of the node G_β with nodes N and N_α. Observe that $Click_{\alpha,\beta}$ counter increases periodically if gateway node G_β is still located in intersection area. Basically, the lager value of the counter implies this gateway node having high stability. Following the above notations, we formally define the detail operation below.

S1. Each node floods a **Beacon** (*HopNumber, NodeList*) within **Beacon.HopNumber** (=2) hops. The field *NodeList* records the node list where this node list denotes the traversal path of beacon packet. This information is also kept in **RoutingTable.***TwoHopPath.*

S2. If this node receives two $\mathbf{Beacon_1}$(*HopNumber, NodeList*) and $\mathbf{Beacon_2}$ (*HopNumber, NodeList*). Let $\mathbf{Beacon}_i.NodeList[k]$ denote the k-th element of *NodeList* in $\mathbf{Beacon}_i$, where $i = 1$ or 2. If $\mathbf{Beacon_1}.NodeList[1] = \mathbf{Beacon_2}.NodeList[2]$, then increase **RoutingTable.**$Click_{\alpha,\beta}$. This implies that the gateway node $NodeList[1]$, which is node G_β, is still located in gateway area. The larger of $Click_{\alpha,\beta}$ is, the higher stability will be achieved.

S3. If there is any no Beacon packet sending from $NodeList[1]$, which is node G_β, for a period of time (time-out condition), then reset **RoutingTable.**$Click_{\alpha,\beta} = 0$.

S4. If there exists two distinct $\mathbf{Beacon_1}$(*HopNumber, NodeList*) and $\mathbf{Beacon_2}$(*HopNumber, NodeList*) and if $\mathbf{Beacon_1}.NodeList[1] = \mathbf{Beacon_2}.NodeList$ [1], then implies that *NodeList* [1] is a branch node such that we set **RoutingTable.**$BranchNode$ = TRUE.

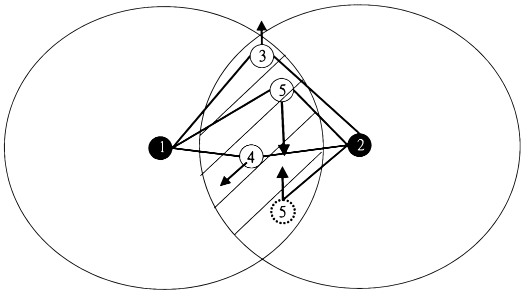


Fig. 5 The probing operation.

The beacon packet is transmitted periodically, there is no existed any synchronous method or timing to guarantee that the beacon packet transmitted from all nodes at the same time. However, it is still success working if a node received an old beacon packet sent from the same node. Furthermore, a sequence number can be added in the beacon packet to avoid using the old beacon packet seriously.

Note that all existing routing protocols discard duplicate packets sent from the same node. On the contrary, our scheme receives duplicate packets from same node, since it implies that there exist multiple paths between two-hops neighboring nodes. This information records in **RoutingTable.***TwoHopPath*, and will be used in the route-reply phase to determine the stable path. The connection status reflects the fact of the node stability. Large value of $Click_{\alpha,\beta}$ implies that corresponding gateway node has high stability. This work is completed by S1 and S2 steps to maintain the value of $Click_{\alpha,\beta}$. In S3 step, $Click_{\alpha,\beta}$ counter is reset to 0 if gateway node is out of the gateway area. Observe that our strategy can eliminate the ping-pong effect. Our path selection strategy is done in route-discovery phase by determining a path with maximum value of $Click_{\alpha,\beta}$. Moreover, the beacon packet identifies the branch node, in S4 step, if the packets are coming from same node. Figure 5 is illustrated an example, node 1 receives beacon packets from node 2 through gateway nodes 3, 4, and 5, so node 1 is a branch node. The stable status is kept in nodes 1 and 2 for gateway nodes 3, 4, and 5.

3.2.3 Step 3: Route-Discovery Operation

Before describing the route-discovery operation, we define the route-discovery packet, which is denoted as **RoutePequest(***EyeInfo, PacketType, TTL, HopCount, DestinationIP, BranchNodeCounter, IntermediateNodeIP***)**, where *EyeInfo* is the eye-nodes' information, *PacketType* represents the packet type (RREQ for Route REQuest packet and RREP for Route REPly packet), *TTL* denotes the packet lifetime, *HopCount* records the hop number from source node to current node, *DestinationIP* represents the IP of the destination host, *BranchNodeCounter* indicates the number of branch nodes in the route path, and *IntermediateNodeIP* records all nodes in the current traversal path.

After branch node is determined, a DSR-like route-discovery operation [5] is performed to obtain the maximum number of branch nodes. Our route-discovery operation is formally developed below.

S1. Source node initiates a **RouteRequest(***EyeInfo, PacketType*=RREQ*, TTL, HopCount*=0*, DestinationIP, BranchNodeCounter*=0*, IntermediateNodeIP*={NULL}**)** packet.

S2. If the receiving node *E* is in the intersectant request-zone by *EyeInfo* and *E* ≠*DestinationIP*, then repeatedly performs S3 step within packet lifetime.

S3. If node *E* is a branch node (**RoutingTable.***BranchNode* == TRUE), then *BranchNodeCounter*++, and floods **RouteRequest (***EyeInfo*, RREQ*, TTL−−, HopCount++, DestinationIP, BranchNodeCounter++,* {old *IntermediateNodeIP,E*}**)** into MANET**.**

S4. Destination node waits for a period of time in order to receive multiple paths from source node. A shortest-path with maximum number of *BranchNodeCounter* is selected as a final path.

Our route-discovery operation is partially like the route-discovery operation in DSR protocol [5], except for keeping *BranchNodeCounter* counter. This field is to accumulate the number of branch nodes, which is maintained in S3 step, for any possible paths. A shortest-path with the maximum number of branch nodes among all of the possible paths will be determined as a final route path. A path with maximum number of branch nodes indicates that this path is with higher stability. Figure 6 gives a scenario of route-discovery operation. Source node is node 1, and two possible paths are reached to node 27. One contains eight branch nodes and another one has six branch nodes. Therefore, the path with eight branch nodes is our final path.

3.3 Route-Reply Phase

We now define the route-reply packet **RouteReply(***PacketType, TTL, DestinationIP, IntermediateNodeIP***)**, where *PacketType* represents the packet type, *DestinationIP* represents the IP of the destination node, and *IntermediateNodeIP* is the node list recording the identified path.

S1. After destination node determining a path, then repeatedly sends a **RouteReply(***PacketType* = RREP*, TTL, DestinationIP, IntermediateNodeIP***)** packet back according to *IntermediateNodeIP.*

S2. For each node receiving **RouteReply** packet, set **RoutingTable.***PrimaryNextHop* as the last node

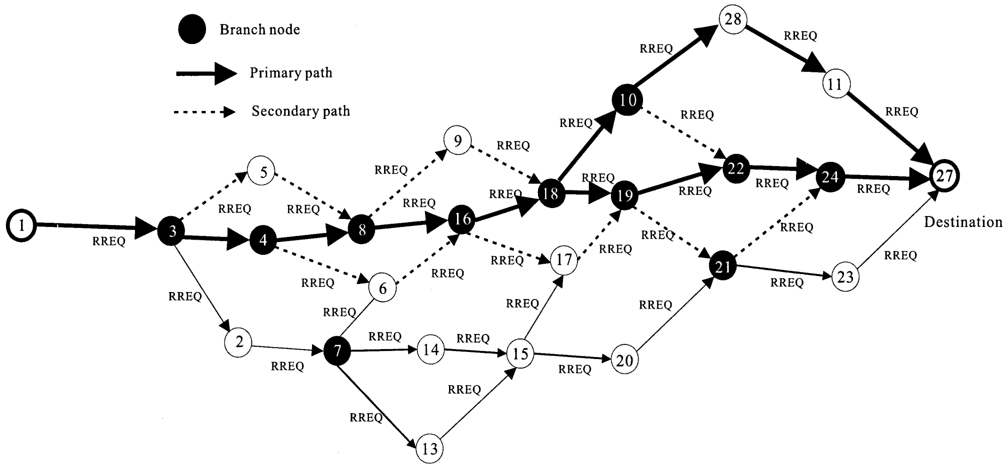


Fig. 6 Our route-discovery operation.

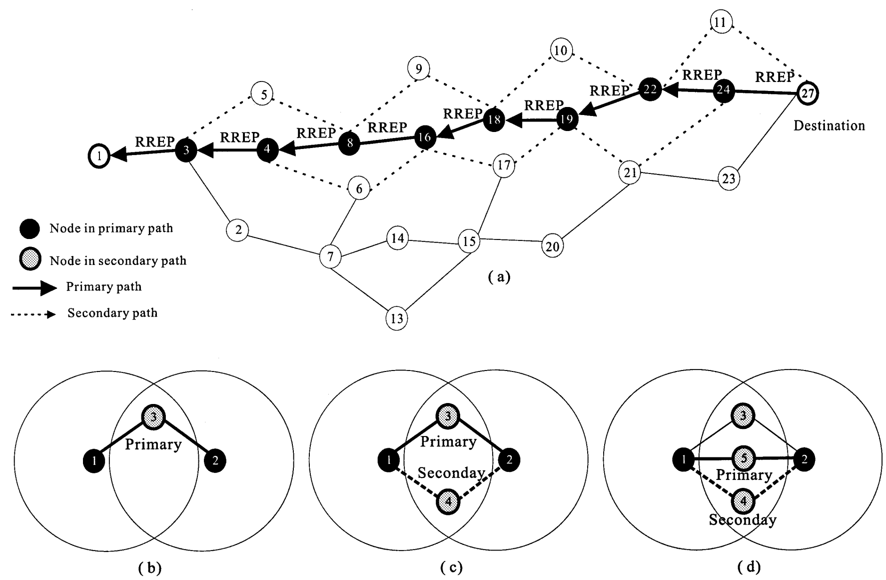


Fig. 7 Our route-reply operation.

in the *IntermediateNodeIP* path. Consider a pair of two-hop neighboring nodes α and β, and there exists a gateway area between these two-hop neighboring nodes. Let $\gamma_j, 1 \le j \le k$, denote gateway nodes in gateway area between α and β. Assume that the **RouteReply** packet is sent back from β to γ_j and then from γ_j to α. Therefore, three cases are considered.

1) If $k = 1$, then **RoutingTable.***PrimaryNextHop* = γ_1 (Fig. 7(b)).
2) If $k = 2$, then **RoutingTable.***PrimaryNextHop* = γ_1 and **RoutingTable.***SecondNextHop* = γ_2. (Fig. 7(c))
3) If $k > 2$, then **RoutingTable.***PrimaryNextHop* = γ_{m_1} and **RoutingTable.***SecondNextHop* = γ_{m_2}, where **RoutingTable.**$Click_{i,m_1}$ and **RoutingTable.**$Click_{i,m_1}$ have the largest values among **RoutingTable.**$Click_{i,j}$, $1 \le j, m_1, m_2 \le k$ (Fig. 7(d)).

S3. Set **RoutingTable.***Destination* to be *DestinationIP*.

Figure 7(a) displays an example of route-reply operation. It is guaranteed that a most stable spiral-hopping path is obtained.

3.4 Route-Maintenance Phase

Our main contribution of MESH protocol is to provide an on-line route-recovery capability. Route error packet, which is used to transmit error message, is denoted as **RouteError(***NodeIP, SourceIP***)**, where *NodeIP* represents detecting-error node and *SourceIP* is the source node. The on-line recovery capability is done by using a backup path to take over the failed path. This work is done by setting **RoutingTable.***PrimaryNextHop* to be **Rout-**

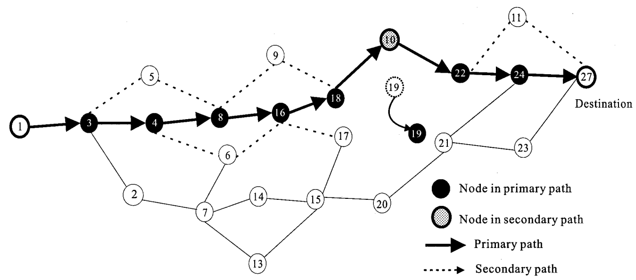


Fig. 8 Our route-maintenance operation.

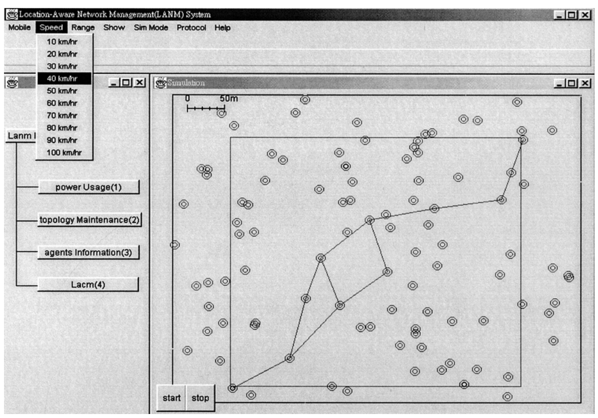


Fig. 9 Our simulation platform.

ingTable.$BackupNextHop$. Figure 8 displays an example when node 19 leaves out, a secondary path is replaced and continues to transmit data.

Note that our P_2 can tolerate multiple non-consecutive faults. However, if multiple consecutive faults occur, then detecting-error node initiates a **RouteError(***NodeIP, SourceIP***)** to source node in order to re-construct other P_2. Furthermore, if there exist consecutive faults, we may construct P_k to tolerate consecutive multi-faults.

4. Performance Evaluation

We have developed a simulator using Java. The core function of the simulator is a discrete event-driven engine designed to simulate systems that can be modeled by processes communicating through signals. We build a simulation platform, as illustrated in Fig. 9. The parameters in our simulation platform are given as follows.

- The number of mobile hosts is from 25 to 100.
- The mobile speed is from 0–10 km/hour to 0–90 km/hour.
- The transmission radius is from 50 to 150 meters
- The routing-protocol selection is DSR, LAR, and MESH.
- Data transmission rate is 2 Mb/sec.
- Route-discovery packet size is 2 k.
- Message length is ranging from 1 k to 30 k.

The simulation platform is simulated in $500 \times 500\,\text{m}^2$ area. To simulate host mobility, each host is simulated by generating a series of turns. In each turn, a direction, a velocity, and a time interval uniformly generated. The direction is uniformly distributed from $0°$ to $360°$, and time interval uniformly distributed from 1 to 100 seconds. The velocity is randomly chosen from 0 to V km/hour, where $V = 10 \ldots 90$. Observe that all moving nodes will not be protruded from the simulation area. This simulation platform keeps the same number of all nodes in the simulation area if we fixed the total number of mobile hosts. This work is simply achieved by a turn action. If a moving node will protrude from the simulation area, we change the direction and velocity of the turn action. The performance metrics to be observed are shown in:

- *REachability (RE)*: the number of destination host receiving the data message divided by the total number of destination host that are reachable, directly or indirectly, from the source host.
- *ReBroadcast (RB)*: the number of route-discovery packets for all mobile hosts in MANET.
- *A*verage *L*atency (*AL*): the interval from the time the unicasting was initiated to the time the last host finishing its unicasting.

It is worth mentioning that an efficient routing protocol is achieved by with high reachability (*RE*), low rebroadcast (*RB*), and low average latency (*AL*). In the following, we show our simulation results of *RE*, *RB*, and *AL* from several prospects.

4.1 Performance of Reachability (*RE*) vs. Mobility

The simulation results of DSR, LAR, and MESH routing protocols are shown in Fig. 10 to reflect the performance of RE vs. mobility. The average *RE* is obtained by calculating average value of all estimated *RE* values. Two types of effects are discussed.

1A) Effects of Number of Mobile Hosts: Each value in Fig. 10(a) is obtained by assuming the transmission radius is 100 meters, and the number of mobile hosts is ranging from 25 to 100. A higher *RE* indicates that a better scheme will be. Figure 10(a) shows that our MESH scheme has higher *RE* than other schemes have even in various number of mobile host and mobility. For example, average *RE* of MESH, LAR, and DSR are 67%, 58%, and 57%, where mobility is 0–30 km/hour. To see the effect of number of mobile host, two cases can be observed. With low mobility (at 0–10 km/hour), the more number of mobile host is, the higher *RE* will be. With high mobility (at 0–10 km/hour), the more number of mobile host is, the lower *RE* will be. This is because that the probability of re-construct a route path is increasing with high number of mobile hosts and high mobility.

1B) Effects of Transmission Radius: Each value

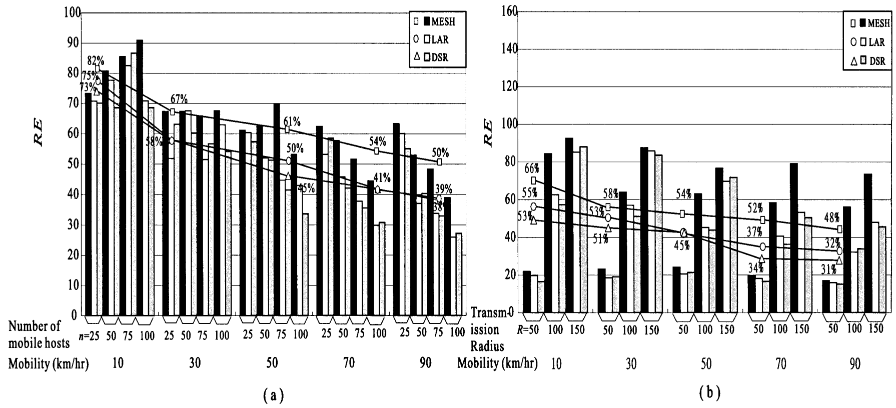


Fig. 10 Performance of reachability (RE) vs. effect of (a) number of mobile hosts, and (b) transmission radius.

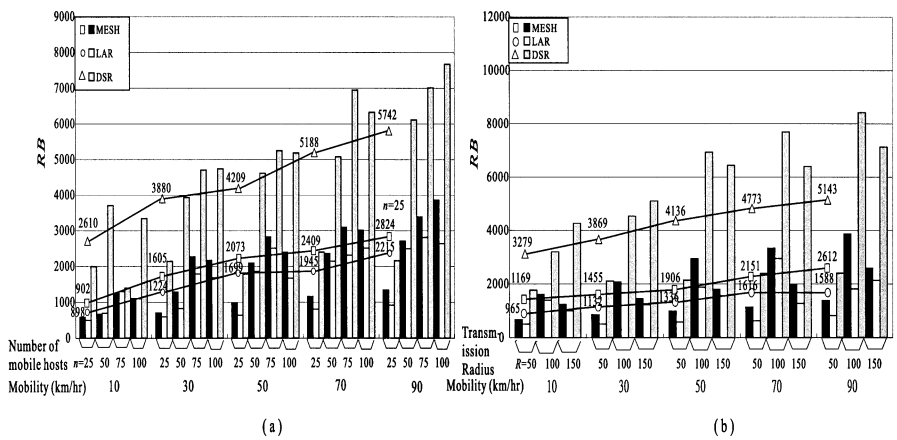


Fig. 11 Performance of rebroadcast (RB) vs. effect of (a) number of mobile hosts, and (b) transmission radius.

in Fig. 10(b) is obtained by assuming that the number of mobile hosts is 75. Average RE of LAR, DSR and MESH are obtained when transmission radius is varying from 50 to 100. Observe that our scheme has lower RE than other schemes have under various transmission radius, as shown in Fig. 10(b). For instance, we see that the average RE of MESH, LAR, and DSR are 58%, 53%, and 51%, where mobility is 0–30 km/hour. Observe that the average RE of MESH is improved 8% than LAR and DSR schemes have, since the spiral-hopping path with on-line route-recovery capability, where value 8% is obtained by calculating the average value of all average RE under various speeds which from 10–90 km/hr.

4.2 Performance of Rebroadcast (RB) vs. Mobility

The simulation results are shown in Fig. 11 to illustrate the performance of RB vs. mobility. The average RB is obtained by calculated average value of all estimated RB values. Two types of effects are discussed.

2A) Effects of Number of Mobile Hosts: The simulation assumption is the same as the *1A* case. A lower RB implies that a better scheme will be. Figure 11(a) shows that our MESH scheme has lower RB than DSR scheme but higher than LAR scheme's RB. For instance, average RB of MESH, LAR, and DSR schemes are 2073, 1699, and 4209, where mobility is 0–50 km/hour. This justifies that smaller request-zone is, less number of rebroadcast packets will be. With low mobility (0–10/30/50 km/hr), it is possible to obtain larger RB with less number of mobile hosts. With high mobility (0–90 km/hr), the more number of mobile host is, the more RB will be obtained. However, average RB of our MESH scheme is 22% higher than LAR scheme has. This is because that MESH scheme needs to broadcast extra packets to maintain the stability.

2B) Effects of Transmission Radius: The simulation assumption is the same as the *1B* case. Figure 11(b) illustrates that our scheme has lower RB than DSR scheme but higher than LAR scheme's RB under various transmission radius. For instance, average RB of MESH, LAR, and DSR is 1906, 1336, and 4136, where mobility is 0–50 km/hour. Observe that aver-

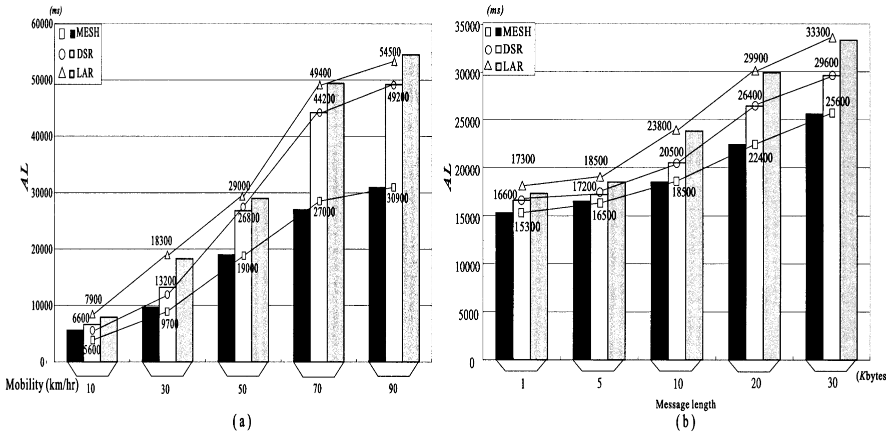


Fig. 12 Performance of average latency AL vs. (a) mobility and (b) message length.

age RB of our MESH scheme is 19% higher than LAR scheme has under various transmission radius.

As a conclusion, comparing Fig. 10 and Fig. 11, we observe that our MESH scheme is to provide a better average reachability though our MESH has about 20% extra number of broadcast packets than LAR scheme.

4.3 Performance of Average Latency (AL) vs. Mobility and Load

The simulation results of DSR, LAR, and MESH routing protocols are shown in Fig. 12 which reflect the effects of average latency. To estimate the performance of AL, two types of effects are discussed.

3A) Effects of Mobility: Figure 12(a) shows AL when message length is 10 k. Our MESH scheme incurs lower latency than DSR and LAR schemes. It reflects the fact that our scheme has better performance than other schemes at various mobility. For instance, average latency of MESH, LAR, DSR are 19000, 26800, and 29000 ms, where mobility is 50 km/hr.

3B) Effects of Number of Message Length: Figure 12(b) shows AL when transmission radius is 100 km and message length is ranging from 1 k–30 k. Our MESH scheme incurs lower latency than DSR and LAR schemes. It offers the fact that our scheme has better performance than other schemes at various message length. For instance, average latency of MESH, LAR, DSR are 18500, 20500, and 23800 ms, where message length is 10 kbyte.

Apparently, it is desirable to have high RE as well as high AL. Generally, the higher RE is, higher AL will be.

4.4 Performance of Effect of P_k

To illustrate the path stability of P_k, we make a simulation comparison of using P_2 or P_3 to be our spiral-hopping-path. For simplicity, let P_2-scheme and P_3-scheme denote as our scheme by using P_2 and P_3, respectively. To estimate the performance of P_2-scheme and P_3-scheme, three types of effects are discussed.

4A) Effects of RE: Figure 13 shows RE when message length is 10 k. We observe that high RE will be obtained if we use P_2-scheme. For instance, as shown in Fig. 13(a) under various of number of mobile hosts, average RE of P_2-scheme and P_3-scheme are 70% and 68%, where the mobility is 50 km/hr. Figure 13(b) shows the average RE under various of number of transmission radius. For instance, P_2-scheme and P_3-scheme are 56% and 52%, where the mobility is 50 km/hr.

4B) Effects of RB: Figure 14(a) shows RB when transmission radius is 100 m and message length is ranging from 1 k–30 k. We also observe that the average RB of P_3-scheme is still higher than P_2-scheme. For instance, average RB of P_2-scheme and P_3-scheme are 2073 and 2539, where the mobility is 50 km/hr.

4C) Effects of AL: Figure 14(b) shows AL when transmission radius is 100 m and number of mobile host is 75. We also observe that the average AL of P_3-scheme is higher than P_2-scheme. For instance, average AL of P_2-scheme and P_3-scheme are 18500 and 20200, where message length is 10 k.

Observe that the performance of P_3-scheme is ineffective compared with P_2-scheme due to the success ratio of constructing P_3 is lower than the ratio of constructing P_2 in our simulation platform. By the similar reason, the success ratio of constructing P_k, $k > 3$, is lower than the ratio of constructing P_3, therefore, the performance of P_k-scheme, $k > 3$, is ineffective compared with P_3-scheme. Form the simulation result, it is can be seen that no better performance will be obtained if our scheme adopted the P_k, where $k \geq 3$. That is why our scheme uses P_2 to be our spiral-hopping-path. To conclude this section, it is always beneficial to adopt our proposed scheme, which is evaluated by our simulation result.

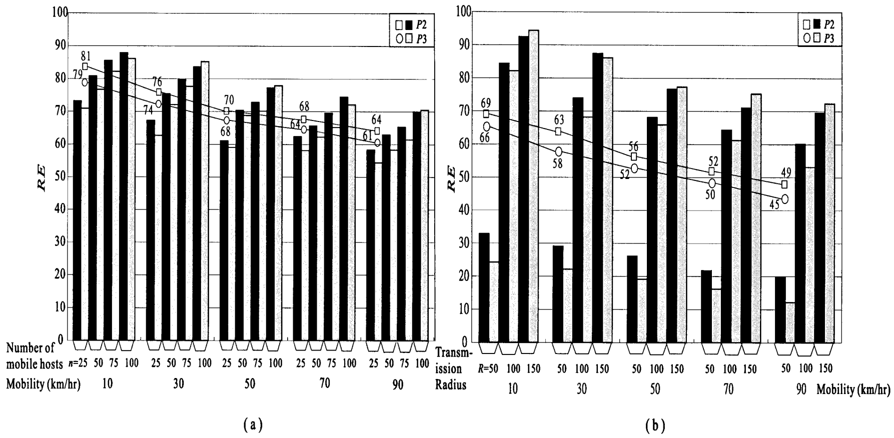


Fig. 13 Performance of average latency *AL* vs. mobility with (a) number of mobile hosts and (b) transmission radius.

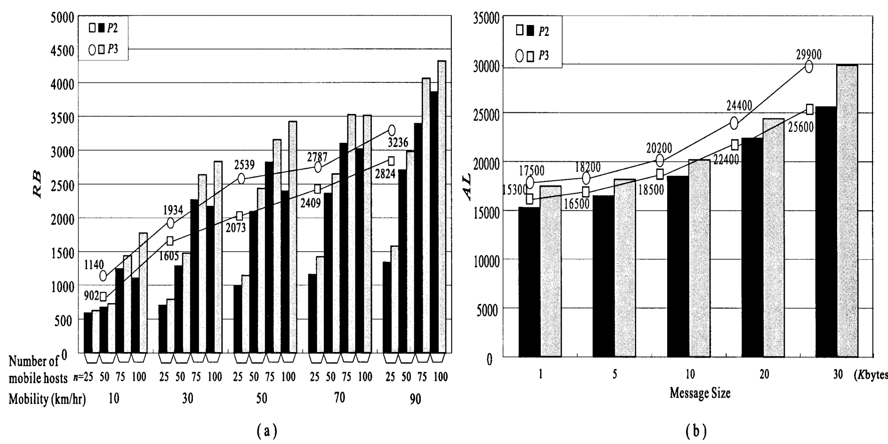


Fig. 14 Performance of (a) *RB* vs. mobility, (b) average latency *AL* vs. message length.

5. Conclusions

This paper addresses a location-aware routing protocol in a mobile ad hoc network. Our proposed MESH protocol is a fully location-aware routing protocol. Our MESH protocol uses the location-information to avoid the blind-flooding region in the route-discovery phase, minimize route-reply packets in the route-reply phase, and promote the routing robustness in the route-maintenance phase. Two major contributions of this paper are introduced: (1) a *multi-eye* strategy is proposed to more accurately confine the route-discovery region for reducing route-discovery and route-reply packets in amount, and (2) a special multi-path strategy, called as *spiral-hopping* strategy, is presented to provide the on-line route-maintenance capability. The performance analysis illustrates that our proposed scheme outperforms existing location-aware routing schemes in MANET. Observe that this paper only considers a one-to-one communication problem, we will further investigate the many one-to-one communication pattern in MANET based on our spiral-hopping strategy. Work is currently underway to develop a multicast protocol based on MESH strategy in the wireless ad hoc network.

References

[1] R. Castaneda and S.R. Das, "Query localization techniques for on-demand routing protocols in ad hoc networks," Proc. Fifth ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99), pp.186–194, Aug. 1999.

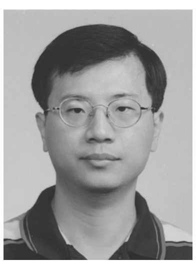
[2] R. Dube, C.D. Rais, K.-Y. Wang, and S.K. Tripathi, "Signal stability-based adaptive routing (ssa) for ad hoc mobile networks," IEEE Personal Communications, pp.36–45, Feb. 1997.

[3] A. Iwata, C.-C. Chiang, G. Pei, M. Gerla, and T.-W. Chen, "Scalable routing strategies for ad hoc wireless networks," IEEE J. Sel. Areas Commun., vol.17, no.8, pp.1369–1379, 1999.

[4] M. Joa-Ng and I.-T. Lu, "A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks," IEEE J. Sel. Areas Commun., vol.17, no.8, pp.1415–1425, Aug. 1999.

[5] D.B. Johnson and D.A. Maltz, "Dynamic source routing

in ad hoc wireless networks," Mobile Computing, ed. T. Imielinski and H. Korth, Chapter 5, pp.81–153, 1996.
[6] E.D. Kaplan, "Understanding gps: Principles and applications," Artech House, Boston, MA, 1996.
[7] Y.-B. Ko and N.H. Vaidya, "Location-aided routing (lar) in mobile ad hoc networks," Proc. Fourth ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98), pp.67–75, Aug. 1998.
[8] W.-H. Liao, Y.-C. Tseng, and J.-P. Sheu, "Grid: A fully location-aware routing protocol for mobile ad hoc networks," Telecommunication Systems, 2000, (to appear).
[9] A. Nasipuri and S.R. Das, "On-demand multipath routing for mobile ad hoc networks," IEEE Eighth International Conference on Computer Communication and Network (IC3N'99), pp.24–30, 1999.
[10] S.Y. Ni, Y.C. Tseng, Y.S. Chen, and J.P. Sheu, "The broadcast storm problem in a mobile ad hoc network," Proc. Fifth ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99), pp.151–162, Aug. 1999.
[11] C.-K. Toh, "Associativity-based routing for ad-hoc mobile networks," Wireless Pers. Commun., vol.4, no.2, pp.1–36, March 1996.



Yuh-Shyan Chen received the B.S. degree in computer science from Tamkang University, Taiwan, Republic of China, in June 1988 and the M.S. and Ph.D. degrees in Computer Science and Information Engineering from the National Central University, Taiwan, Republic of China, in June 1991 and January 1996, respectively. He joined the faculty of Department of Computer Science and Information Engineering at Chung-Hua University, Taiwan, Republic of China, as an associate professor in February 1996. He joined the Department of Statistic, National Taipei University in August 2000. Dr. Chen served as a Program Committee Member in the Int'l Conf. on Parallel and Distributed Systems, 2001, the Int'l Conf. on Computer Communications and Networks, 2001. He was a Workshop Co-Chair of the 2001 Mobile Computing Workshop, and Guest Editor of Journal of Internet technology, special issue on "Wireless Internet Applications and Systems." His paper wins the 2001 IEEE 15th ICOIN-15 Best Paper Award. His current research include parallel/distributed processing, network security, wireless communication/mobile computing, and ad-hoc QoS routing protocols. Dr. Chen is a member of the IEEE Computer Society and Phi Tau Phi Society.

Kuan-Chang Lai received the B.S. and M.S. degrees in Computer Science and Information Engineering from Feng-Chia and Chung-Hua University, Taiwan, R.O.C., in June 1998 and June 2000, respectively. His current research interests include parallel and distributed processing and collective communication.