Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 1152–1163
computer communications
www.elsevier.com/locate/comcom

# A low overhead dynamic route repairing mechanism for mobile ad hoc networks

Chang Wu Yu [a], Tung-Kuang Wu [b,*], Rei Heng Cheng [c]

[a] Department of Computer Science and Information Engineering, Chung Hua University, Hsin-Chu, Taiwan, ROC
[b] Department of Information Management, National Changhua University of Education, No. 2, Shi-Da Road, Changhua, Taiwan, ROC
[c] Department of Computer Science, Hsuan Chuang University, Hsin-Chu, Taiwan, ROC

Received 11 June 2006; received in revised form 27 November 2006; accepted 4 December 2006
Available online 11 January 2007

**Abstract**

Ad hoc networks are wireless networks with no fixed infrastructure. Each mobile node in the network functions as a router that discovers and maintains routes for other nodes. These nodes may move arbitrarily, therefore network topology changes frequently and unpredictably. Many routing protocols have been designed for ad hoc networks. However, most of these kinds of protocols are not able to react fast enough to maintain routing. In this paper, we propose a new protocol that repairs the broken route by using information provided by nodes overhearing the main route communication. When links go down, our protocol intelligently replaces these failed links or nodes with backup ones that are adjacent to the main route. Theoretical analysis reveals that, in a given circumstance, our proposed protocol can find a backup route in more than 60% of time. Simulation results also demonstrate that our protocol achieves better (or as good) in terms of the packet delivery rate, control packet overhead and communication delay than the major ad hoc routing protocols under light and moderate traffic conditions.


*Keywords:* Dynamic route repairing; Ad hoc networks; Routing protocols; AODV; DSR; NS-2

## 1. Introduction

Ad hoc networks are wireless networks with no fixed infrastructure. Each mobile node in the network functions as a router that discovers and maintains routes for other nodes. These nodes may move arbitrarily, therefore network topology changes frequently and unpredictably. Other limitations of ad hoc networks include high power consumption, low bandwidth, and high error rates [1]. Applications of ad hoc networks are emergency search-and-rescue operations, meetings or conventions in which persons wish to quickly share information, data acquisition operations in inhospitable terrain, and automated battlefield [1].

For years numerous routing protocols have been developed for ad hoc networks including Destination-Sequenced Distance-Vector Routing protocol (DSDV) [2], Clusterhead Gateway Switch Routing protocol (CGSR) [3], Wireless Routing Protocol (WRP) [4], Ad Hoc On-Demand Distance Vector (AODV) [5,6], Dynamic Source Routing (DSR) [7,8], Temporally Ordered Routing Algorithm (TORA) [9,10], Associativity-Based Routing (ABR) [11], and Zone Routing Protocol (ZRP) [12].

In general, the existing routing protocols may be categorized as table-driven and source-initiated on-demand [1]. Table-driven protocols try to maintain routing information from each node to every other node in the network. As network topology changes, these protocols propagate updates throughout the network in order to maintain a consistent global network view. A large portion of network capacity is used to keep the routing information up-to-date, even though most of the information is never used. On the other

* Corresponding author. Tel.: +886 4 7232105x7615; fax: +886 4 7211162.
*E-mail addresses:* cwyu@chu.edu.tw (C.W. Yu), tkwu@mail.tkwu.net (T.K. Wu), rhc@hcu.edu.tw (R.H. Cheng).

0140-3664/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2006.12.008

hand, source-initiated on-demand protocol [1] creates routes only when desired. Once a route has been established, it is maintained by a route maintenance procedure until either link failures occur or the transmission job is complete. Because routing information may not be available when a route request is received, the latency to determine a route can be quite significant. When the rate of topological changes in the network is sufficiently high, most of the above protocols may not be able to react fast enough to maintain necessary routing.

In this paper, we propose a new routing algorithm, which overcomes the drawback associated with the conventional routing algorithms. The proposed new protocol dynamically repairs broken routes by using information provided by nodes overhearing the main route communication. When links go down, our protocol intelligently replaces these failed links or nodes with backup ones that are adjacent to the main route. Theoretical analysis reveals that, in a given circumstance, our proposed protocol can find a backup route in more than 60% of time. Simulations also show that the proposed algorithm achieves better in most aspects than most of the notable protocols. Note that an earlier concise version of this work has appeared in [13].

The rest of paper is organized as follows. In Section 2, we give a survey of how relevant routing protocols react to link failure. The main ideas of our protocol and brief comparison of ours and two other notable dynamic route repairing protocols are described in Section 3, with analysis of the proposed protocol and its simulation results presented in Sections 4 and 5. Section 6 concludes the paper.

## 2. Related work

Table-driven routing protocols require constant propagation of routing information, which incurs extra communication overhead and power consumption. As a result, these become the limiting factors of their applications in ad hoc network environment since both bandwidth and battery power is scare resource in mobile devices [1]. On the other hand, on-demand routing protocols establish a route only when a source requires to send messages to some destination without requiring periodic update of routing information. However, various on-demand routing protocols differ in how they handle route maintenance and how they react to link failure, which will be the focus of the following survey.

AODV [5,6] routing is an improvement to the table-driven and distance-vector-based DSDV algorithm. With DSDV routing [2], every mobile node maintains a routing table recording all the possible destinations and number of hops to each destination. In order to maintain routing table consistency, it requires nodes to periodically broadcast routing updates throughout the network. AODV minimizes the number of broadcast messages associated with DSDV by building routes on a demand basis. In case a broken link notification is received, source nodes in AODV would restart the route discovery process. But before sending the link failure notification to source, AODV allows the upstream node of the break to try to repair a recently used route by sending a Route Request (RREQ) message [6]. However, if the route repairing attempts were unsuccessful, more data packets would be lost.

Agarwal and Jain [14] proposed a modified AODV protocol in which a node records information of its two upstream nodes (instead of one with AODV) upon receiving a RREQ. When the same node receives Route Response (RREP) from the destination, it relays the RREP to the two nodes that it records earlier. After some evaluation, one of the upstream nodes will be included in the main route with the other being used for the backup route. Lee and Gerla [15] proposed a so called AODV-BR protocol in which a node (e.g., some node $B$ that is one hop away from the main route) overhears a RREP packet sent from node $Y$ to $X$ will mark $Y$ as its next hop. In case node $X$'s upstream node (say, node $A$) finds its link to node $X$ broken, it issues a FIND packet requesting a backup route. When node $B$ receives such FIND packet, it responds and forms a new backup route (route changes from original $A \rightarrow X \rightarrow Y$ to current $A \rightarrow B \rightarrow Y$). Chen and Lee [16] proposed a 2HBR protocol, which extends the idea of [15] further by including nodes that are two hops from the main route as the potential backup nodes. The Multiple Next Hops (MNH) routing protocol proposed by Jiang and Jan [17], applies the concepts of forward link and reverse link used in AODV. For each destination, each mobile node in MNH routing protocol maintains multiple next hops in its routing table. Hence the MNH may provide multiple routing paths for a source–destination pair. As link failure occurs, the upstream node will detect that and try to reconstruct a new route.

DSR [7,8] uses source routing, with each packet to be routed carrying in its header the complete, ordered list of nodes through which the packet have to go through. If a link fails, the upstream of this failed link sends a route error packet to the source node. When a route error is received, the hop in error is removed from this host's route cache, and all routes that contain this hop must be truncated at that point. A new route discovery process must be initiated by the source. A salvaging technique proposed by Maltz et al. [18] uses alternate route from caches when a data packet meets a failed link on its source route.

Chung et al. [19] proposed the Ad Hoc Backup Node Setup Routing Protocol (ABRP) that is similar to the DSR. ABRP saves backup route information in certain on-the-route node. When a link fails, data messages are sent back to a backup node. The backup node checks its backup route cache, and pick a path (if there exists one) to replace the current broken one. Similarly, ABRP does not update its backup route information to reflect the network topology change.

TORA [9,10], also a source-initiated routing algorithm, maintains multiple routes for any desired source–destination pair. The key idea of TORA is that control messages are restricted to a small set of nodes in case of topological

change. However, to achieve this, it needs to build a directed acyclic graph (DAG) rooted at the destination. As links fail, only local routes are re-established to a destination-oriented DAG within a finite time. For maintaining a list of a node's neighbors, each node periodically transmits a BEACON packet, which is answered by each node hearing it with a Hello packet. Furthermore, in order to maintain the order, TORA needs a global timer to record the time of link failure.

As we have seen in the above review, the two major on-demand protocols (AODV and DSR) do not respond quickly enough to link failure. And they usually suffer from the risk of flooding the whole network for new route discovery. Variations of the two protocols try to cope with these issues by storing extra backup routes for use upon link failure. However, the backup routes are usually created statically during initial routes construction stage, no effort is being done to modify these routes in order to reflect the changing network topology. Therefore, as the topology of network changes, there is little chance of using these backup paths. On the other hand, TORA replies on periodic HELLO messages to monitor the status of network topology and tries to fix broken routes that may not be used later on, which considerably increases its protocol overhead.

Contrary to the above mentioned static route repairing protocols, such as AODV-BR and 2HBR, Castañeda et al. [20] proposed two heuristics that utilize prior routing histories to localize the query floods to a limited region of the old routes and the dynamically collected information to repair the broken route. The first heuristic exploits route locality, which guarantees the new route (if there is one) would not be very different (at most $k$ nodes) from the most recently used one in case of route breaking. The second heuristic exploits node locality assuming that the destination node can be found within a small number of hops from some node on the most recently used route. In both of the two protocols, they need to include the last valid route within the query messages, thus increasing the size of messages. They have also evaluated the performance of the query localization techniques by integrating them to the DSR routing algorithm.

In [21], Hu et al. also proposed an efficient route update protocol (ERUP) to dynamically update the damaged route by using the information of the old route, but at the same time select a new route that may not be overlapping to the old one, which is essential for wireless sensor networks. Their process consists of two steps: (1) the nodes along the old route broadcast locally a route discovery region (RDR) packet to define the spreading area of route request (RRQ) packet; and then (2) a RRQ packet is released to discover new routes to the sink. Note that the early version of their idea can also be found in [22].

## 3. The proposed new routing protocol

In the following, we present a new fully distributed and on-demand based ad hoc routing protocol with nearly real time repairable route that handles the broken-link recovery in a more efficient way. Before moving on to the details, we first present the intuitive ideas behind our protocol.

The proposed routing protocol begins by finding a route from a source node $S$ to a destination node $D$, which we call the *main route*. All data packets are then sent along this main route to the destination. As the data packets proceed to move along the main route, nodes that are close enough to the path will overhear the messages. In other words, nodes that are able to overhear the messages should be close enough to the main route and are potentially good candidates for substituting the potential failed node. By piggybacking appropriate information within packets and applying proper procedures, our algorithm makes nodes that overhearing the packets the backup nodes for future route reconstruction in case of broken main route. Other than an additional field is added (for storing height value, as explained later) to a node's route table and header of message, there is no need for frequent message flooding and huge table maintenance.

The proposed algorithm, consists of route construction and route maintenance, is given as follows.

### 3.1. Route construction

In our proposed routing protocol, a routing path is constructed only when a node needs to communicate with another node. Assume that a source node $S$ needs to send a packet to some destination node $D$. If the destination node $D$ is a neighbor of source node $S$, the packet is sent directly to node $D$. Otherwise, the source node will first check if node $D$ is in its main route table (MRT). If it is, packets will then be sent directly to the next-hop node as specified by the corresponding entry. On the other hand, a path (the main route) from source node $S$ to destination node $D$ need to be constructed before source node $S$ can start the data transmission. The process of finding such a routing path is called the *main route construction*.

The main route construction process begins with source node $S$ sending a *main route request* (MREQ) to all its neighbors. Every host that receives the MREQ acts exactly the same as the source node does. MERQ is thus flooded over the network, and will eventually arrive at node $D$ if a routing path exists between them. When node $D$ receives a MERQ, it sends back *main route reply* (MRRP) and a value $H$, representing the hop number from this node ($D$) to the destination (which is zero in this case), to the host (say $P_1$) from which MREQ was received. Once node $P_1$ receives MREQ, it adds a main route entry for node $D$ to its MRT. It then propagates the MRRP, together with value $H+1$, to the host from which $P_1$ receives the MREQ. Every other host receiving MRRP behaves similarly as $P_1$ until node $S$ receives MRRP and updates its MRT accordingly. A routing path from node $S$ to $D$, referred to as the main route, is thus established. Host $S$ can now sends its packets destined for node $D$ through this path. Fig. 1 illustrates the process of main route construction described above.

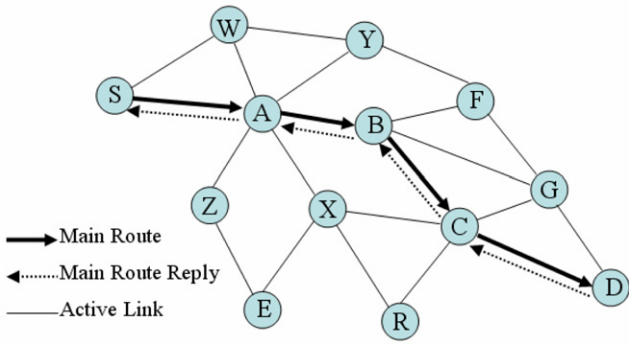



Fig. 1. Illustration of the main route construction process.

### 3.2. Route maintenance

The route maintenance process consists of two parts: (1) the main route messages sniffering and (2) the main route repairing. In main route messages sniffering stage, our concern is how to manage the messages going through the main route that are overheard by the neighboring nodes. With main route repairing, we need to take care of a broken main route using the messages collected in the main route messages sniffering stage and avoid the flooding of *repair query* (REPQ) packets as long as possible.

#### 3.2.1. The main route messages sniffering

The main route messages sniffering stage begins after packets start delivering through the main route. However, we need to modify the packets delivery phase slightly to accomplish this task. First of all, we need to insert an $H$ field into the header of data packet and require the nodes to maintain an extra height table for storing the $H$ value (as described in Section 3.1). Second, with the broadcast nature of wireless communication, a node promiscuously "overhears" packets transmitted by their neighboring nodes that are within the radio range. In case a node, which is not part of the main route, overhears a data packet transmitted by a neighbor on the main route, it records the $H$ value within the packets header into its height table. If more than one such packet is received, the average of the received $H$ values is computed and then recorded in its height table. The recorded $H$ value can later be used to assist the repairing of the route and to restrain the flooding of control packets.

To further illustrate the idea, we give an example as follows. Suppose that we have just constructed a main route from node $S$ to node $D$ (as illustrated in Fig. 2(a), together with $H$ value of each node). Packet delivering is then begun, which also triggers the main route messages sniffering process. In case node $A$ is within the communication range of node $S$, it should overhear the packets that are sent by source node $S$ (destined for node $W$). Since the packets from node $A$ have tagged 5 in their $H$ field, node $A$ would store 5 ($H = 5$) to its height table. Furthermore, as in this case, node $A$ also overhears data packets (with $H = 4$) from node $W$ to $X$, a new value equal to the mean (4.5) of the two $H$ values it receives (5 and 4 received from

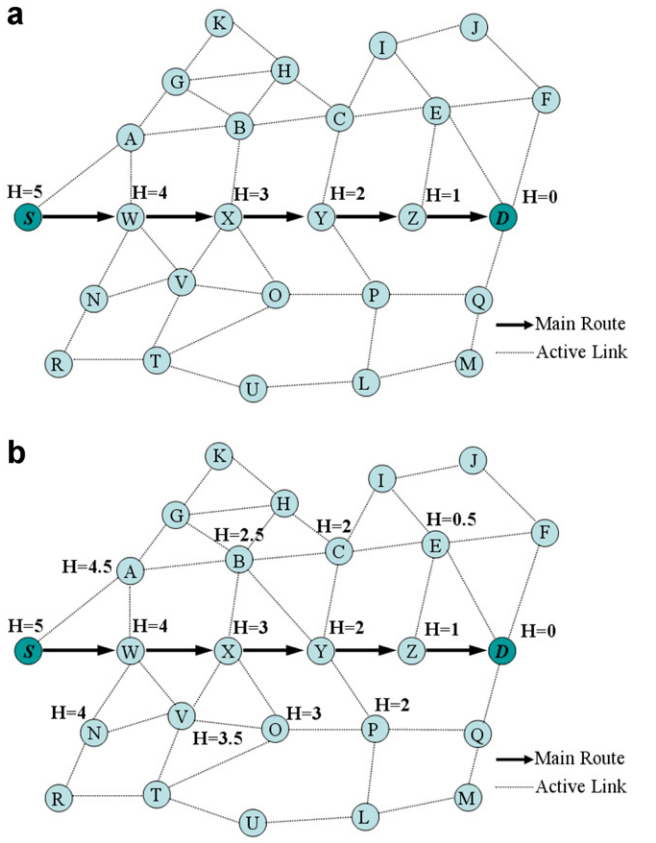



Fig. 2. Main route message sniffering and $H$ value construction. (a) Scenario showing $H$ values associated with nodes along the main route. (b) Scenario showing $H$ values, after message sniffering, associated with nodes neighboring to the main route.

node $S$ and $W$, respectively) will then replace the original value (5) in the height table. More similar examples of height value computation are shown in Fig. 2(b).

#### 3.2.2. The main route repairing

When a link on the main route is broken, the upstream node of this link will find out in a period of time and then initiate the main route repairing process by broadcasting a *repair query* (REPQ) packet. The REPQ packet contains the height value of the initiating node. Each node that receives the REPQ packet would first check if it has a route to the destination. If it does, a *repair reply* (REPR) packet is sent back to the node from which the *repair query* (REPQ) packet was received and the route repairing process is done. Otherwise, it compares the height value (denotes $H_{req}$) contained in the REPQ packet to its own height value (denoted by $H_t$) in the height table to determine what the next step will be. In case $H_{req}$ is greater than (or equal to) $H_t$, which means very likely the route repairing request was sent from a node that is closer (or as close) to the source node, the receiving node would then rebroadcast the REPQ and hope the REPQ packet would propagate to the nodes that are closer to the destination. However, if $H_{req}$ is smaller than $H_t$ or the node has no $H_t$ in its height table, the REPQ packet will be dropped.

For example, as shown in Fig. 3(a), there is a route from node $S$ to $D$, subsequently going through nodes $W$, $X$, $Y$, and $Z$. As node $Y$ moving out of the radio range of its upstream node $X$, node $X$ will realize after a while that the link is broken since it does not receive any more reply from node $Y$. It would then send a *repair query* (REPQ) packet to initiate the route repairing process. In that case, node $W$, $B$, $V$, and $O$ receive the REPQ packet (Fig. 3(b)). Assume that none of the nodes has a route in its main route table that destined for node $D$. According to the protocol we described earlier, node $W$ and $V$ would drop the REPQ packet since they have larger height values than that contained in the REPQ packet. On the other hand, node $B$ and $O$ will rebroadcast the REPQ packet. Let just focus on node $B$; the rebroadcast REPQ packet (by $B$) will be received by node $A$, $G$, $H$, and $C$. As in previous case, node $A$ drops the REPQ packet, and so is node $G$ and $H$ since it has no height value in their height table. Node $C$ is the only one that would rebroadcast the REPQ packet, and again received by node $E$, $H$, $I$, and $Y$. The same rule applies to node $E$, $H$, and $I$, and the process goes on. However, node $Y$ reacts differently by sending back REPR packet to node $C$ since it has a route to the destination node $D$ in its MRT. The REPR packet is then forwarded, with the reverse order that REPQ packet was received, to node $B$, and eventually arrives at node $S$. As shown in Fig. 3(c), a repaired main route ($S \rightarrow W \rightarrow X \rightarrow B \rightarrow C \rightarrow Y \rightarrow Z \rightarrow D$) is now operational.

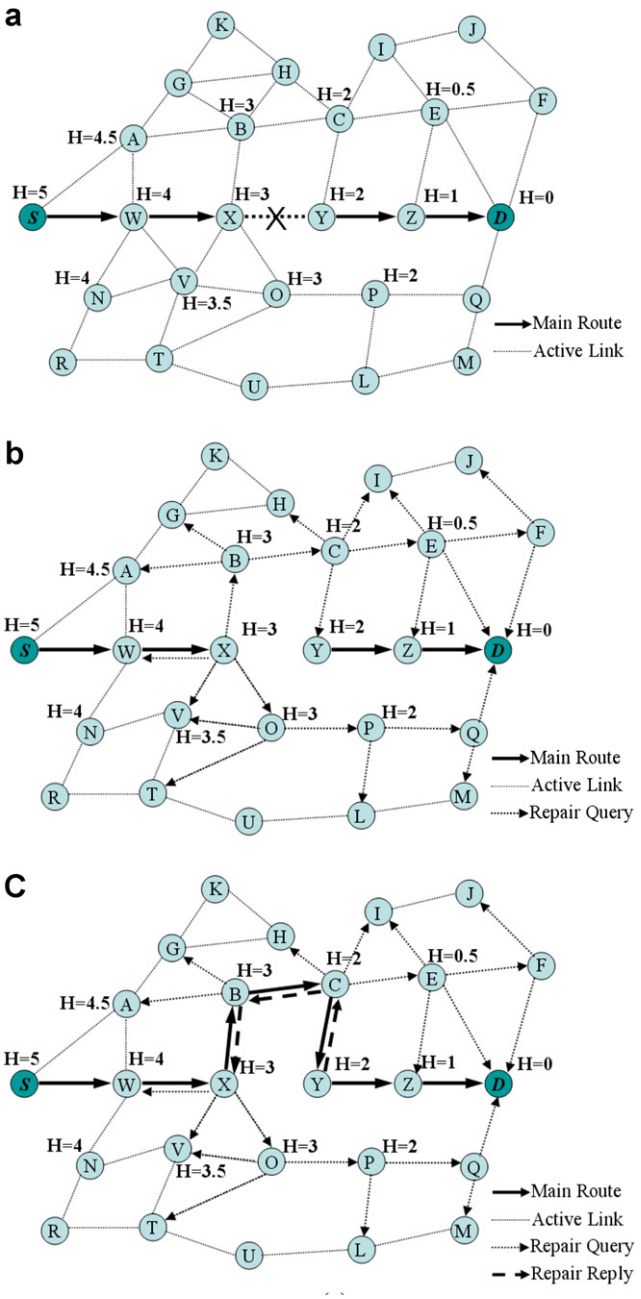



Fig. 3. Illustration of the main route repairing process. (a) Scenario showing the main route is broken on edge $X$–$Y$. (b) Scenario showing the flowing of REPQ messages after route breaking. (c) Scenario showing a repaired main route has been constructed and operational.

Our proposed routing protocol described above is simple, and should be viable to be included in any ad hoc network without incurring much overhead. The next two sections will demonstrate that our proposed protocol also performs better than the existing routing protocols. The formal description of our protocol is described as follows.

*3.2.2.1. Route construction.* Source node $S$ floods a MREQ for constructing a main route.

When destination node $D$ receives a MERQ, it sends back MRRP with a value $H$.

When some node $P_1$ receives MREQ, it adds a main route entry for node $D$ to its MRT. It then propagates the MRRP, together with value $H + 1$, to the host from which $P_1$ receives the MREQ.

*3.2.2.2. The main route messages sniffering.* When a node $A$ overhears a data packet transmitted by a neighbor on the main route, it records the $H$ value within the packets header into its height table. If more than one such packet is received, the average of the received $H$ values is computed and then recorded in its height table.

*3.2.2.3. The main route repairing.* When a node finds out that a link on the main route is broken, the node initiates the main route repairing process by broadcasting an REPQ packet.

When receiving the REPQ packet, each node checks if it has a route to the destination. If yes, an REPR packet is sent back to the node from which the REPQ packet was received and the route repairing process is done. Otherwise, it compares the height value (denoted by $H_{req}$) contained in the REPQ packet to its own height value (denoted by $H_t$) in the height table to determine what the next step will be. If $H_{req} \geqslant H_t$, the receiving node rebroadcasts the REPQ. On the other hand, if $H_{req} < H_t$ or the node has no $H_t$ in its height table, the REPQ packet will be dropped.

### 3.3. Comparison of ours and other major route repairing protocols

After presenting our protocol, we are now able to make a brief comparison of the three dynamic route repairing

Table 1
Comparison of the characteristics of some route repairing protocols

| Characteristics | QL | ERUP | Ours |
|---|---|---|---|
| Method used to exploit locality for route repairing | Caching prior routing histories | Confining the repairing route within the route discovery region | Restricting the search area with the altitude information |
| The size of route discovery region | Dynamic | Dynamic | One-hop neighbors |
| Capability to find a new route that is mostly disjoined with the old one | N/A | Good | N/A |
| Additional field needed | "last valid route" field in the query message | "TTL" field in route discovery region packet and "mark bit" in every node | "height value" field in route table and header of messages |
| Considering balance in power consumption | No | Yes | No |
| Target networks | Ad hoc networks | Wireless sensor network | Ad hoc networks |

protocols: query localization (QL) [20], efficient route update protocol (ERUP) [21,22], and ours in terms of their design philosophies in exploiting route or node locality.

The QL protocol caches prior routing histories to estimate a small region in the network with high probability of finding the destination node. The ERUP broadcasts locally a control packet to define the spreading area; then another packet is used to discover new routes to the sink in the area. Our protocol, on the other hand, restricts the search boundary for repairing nodes by monitoring the altitude information maintained by nodes on the main route through packet sniffering. The details are organized and presented in Table 1.

In Section 5, we will compare the performance of ours and QL protocol through simulations. ERUP, on the other hand, is excluded for further comparison as it was developed specifically for wireless sensor networks, which has quite different design considerations.

## 4. Effectiveness of the proposed protocol in repairing broken route

In this section, we will show the effectiveness of our proposed protocol by first giving a theoretical analysis derived from random geometric graphs, and followed by presenting the simulation results.

Given a *geometric graph* $G = (V, r)$, which consists of nodes placed in 2-dimension space $R^2$ and edge set $E = \{(i,j) | d(i,j) \leqslant r$, where $i, j \in V$ and $d(i,j)$ denotes the Euclidian distance between node $i$ and node $j\}$. Let $X_n = \{x_1, x_2, \ldots, x_n\}$ be a set of independently and uniformly distributed random points. We use $\Psi(X_n, r, A)$ to denote the *random geometric graph* (RGG) [23] of $n$ nodes on $X_n$ with radius $r$ and placed in a rectangle area $A$. RGGs consider geometric graphs on random point configurations, which can be used to model an ad hoc network $N = (n, r, A)$ consisting of $n$ mobile devices with transmission radius $r$ unit length. When each vertex in $\Psi(X_n, r, A)$ represents a mobile device, each edge connecting two vertices represents a possible communication link as they are within the transmission range of each other. A random geometric graph and its representing network are shown in Fig. 4.

One device in Fig. 4 is deployed nearby the boundary of rectangle $A$ so that its radio communication range (often modeled by a circle) is not properly contained in $A$. This is due to *border effects*, which complicate quantitative analyses of ad hoc networks; therefore, previous discussions usually circumvent the border effects by using *torus convention*. Torus convention models the network topology in a way that nodes nearby the border are considered as being close to nodes at the opposite border and they are allowed to establish links. In the paper, we adopt torus convention to avoid border effect problem.

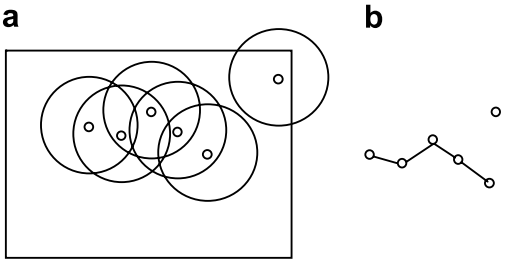



Fig. 4. (a) An ad hoc network $N = (6, r, A)$, where $A$ is a rectangle. (b) Its associated random geometric graph $\Psi(X_6, r, A)$.

If one of two equal-sized circles in the place contains the center of the other, we call them two *properly intersecting circles*. The expected overlapped area of two properly intersecting circles has been computed in the next theorem.

**Theorem 1.** [24]: *The expected overlapped area of two properly intersecting circles with radius r is* $\left(\pi - \frac{3\sqrt{3}}{4}\right) r^2$.

The next theorem estimates the expected number of common neighboring nodes of each communication link in a given ad hoc network.

**Theorem 2.** *For each communication link in a* $\Psi(X_n, r, A)$, *the expected number of common neighboring nodes* $N_{expected}$ *of a communication link is* $N = n \times \left(\pi - \frac{3\sqrt{3}}{4}\right) r^2 / |A|$, *where n is the number of randomly deployed nodes and A is the deployed area.*

**Proof.** Given two circles with the same radius $r$ in a $\Psi(X_n, r, A)$, the expected overlapped area of these circles whose centers are within the other is $\left(\pi - \frac{3\sqrt{3}}{4}\right) r^2$ (by Theorem 1). Then the probability that a node is located in the

overlapped area of these two properly intersecting circles equals $\Pr(e_i e_j | e_k) = \left(\pi - \frac{3\sqrt{3}}{4}\right) r^2/|A|$, for three arbitrary distinct edges $e_i = (u, v)$, $e_j = (u, w)$, and $e_k = (v, w)$ in a $\Psi(X_n, r, A)$ where $u \neq v \neq w$. Finally, we have the expected number of common neighboring nodes $N_{\text{expected}}$ of a communication link is $N = (n) \times \left(\pi - \frac{3\sqrt{3}}{4}\right) r^2/|A|$, where $n$ is the number of randomly deployed nodes and $A$ is the deployed area. □

Suppose that each communication link has the same probability of link failure, denoted by $P_F$. Also, let $P_R$ denote the probability of that a link successfully recover from its link failure in our protocol. The following theorem tries to derive a bound of the probability of successful link recovery in our routing protocol.

**Theorem 3.** *We have* $P_R \geqslant 1 - (P_F \times (2 - P_F))^N$ *in a* $\Psi(X_n, r, A)$, *where* $N = (n) \times \left(\pi - \frac{3\sqrt{3}}{4}\right) r^2/|A|$ *and n is the number of randomly deployed nodes and A is the deployed area.*

**Proof.** If a communication link $(a, b)$ on the main route fails in a $\Psi(X_n, r, A)$, there is high probability that the common neighboring nodes $N_{ab}$ of both $a$ and $b$ can be used to repair such failed link in our protocol. Suppose that $U$ are nonempty subsets of $V$. The subgraph of $G = (V, E)$ whose vertex set is $U$ and whose edge set contains those edges of $G$ that have both ends in $U$ is called the subgraph of $G$ induced by $U$. In other words, the induced subgraph of each of these nodes together with nodes $a$ and $b$ form a cycle of length 3; these nodes form $N$ disjoint paths with length 2 (Fig. 5) as backup routes in our protocol. By Theorem 2, we have $N = (n) \times \left(\pi - \frac{3\sqrt{3}}{4}\right) r^2/|A|$. The probability of failure recovery of each path is $P_F + (1 - P_F) \times P_F = P_F \times (2 - P_F)$ because the breakage of any link of the path results its failure recovery. Since there are at least $N$ disjoint paths, the failure of link recovery $(a, b)$ occurs when all $N$ disjoint paths fail. Therefore, we have the desired result $P_R \geqslant 1 - (P_F \times (2 - P_F))^N$. □

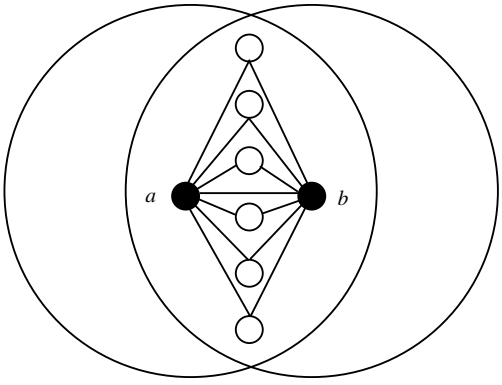



Fig. 5. A link $(a, b)$ with some common neighboring nodes.

Table 2
Percentage of successful route repair

| Pause time (in seconds) | Percentage of successful repair |
|---|---|
| 10 | 45.95% |
| 20 | 47.26% |
| 30 | 50.32% |
| 40 | 51.60% |
| 50 | 55.56% |

For example, given $P_F = 0.5$, $n = 100$, $r = 250$ m, and $A = 1500 \times 1500$ m, we have $N = (n) \times \left(\pi - \frac{3\sqrt{3}}{4}\right) r^2/|A| = 100 \times \left(\pi - \frac{3\sqrt{3}}{4}\right) 250^2/(1500 \times 1500) \cong 5.118$. That indicates each link may have approximately five backup routes. Consequently, we have $P_R \geqslant 1 - (P_F \times (2 - P_F))^N = 1 - (0.5 \times (2 - 0.5))^{5.118} \cong 0.637$. This means that around 60% of cases in the given circumstance, a broken route can be recovered with our proposed algorithm without flooding the network with new route discovery requests. Although the recovery rate may vary with different variable settings, however, this is still quite an encouraging result.

To validate the theoretical analysis, we simulate scenario as depicted in the above calculation using NS-2. The results, as shown in Table 2, indicate that the broken routes can be repaired in around 50% of cases of the simulations. As can be expected, the successful route repairing rate increases in accordance with the pause time since longer pause time implies more stable nodes.

## 5. Performance of the proposed protocol as compared to major protocols

Three on-demand protocols including AODV, DSR, and TORA are used as the basis of comparison to our protocol. These three protocols are all supported by NS-2. Four aspects in evaluating how these algorithms perform are simulated and given in the following four subsections, include (1) data delivery rate, (2) routing overhead, (3) communication latency, and (4) average number of hops a message needs to traverse to reach its destination.

In addition to the three major routing protocols, we will also indirectly compare our protocol with QL protocol that implemented as a simple extension of DSR [20]. The comparison is made possible through the use of common simulation settings and the fact that DSR is appeared in both simulations.

We use NS-2 [25], which was adopted in numerous researches to evaluate the performance of existing ad hoc routing protocols [26,27], as the simulation tool. The link layer of our simulator is IEEE 802.11 Distributed Coordination Function. Physical and data link layer models are devised and described in [26].

The radio coverage region of each mobile node is assumed to be a circular area of diameter 250 m. The transmission time for a hop takes 0.002 s, and the beacon period is 1 s. We assume the source node sends a 64-byte data

packet every 0.33 s (with constant bit rate), and use UDP as the transport protocol. Each node has a queue, provided by network interface, for packets awaiting transmission that holds up to 50 packets and managed in a drop-tail fashion.

The simulation environment is specified by a $2200 \times 600$ m rectangular region with 100 mobile nodes moving around. Each of the 100 nodes is placed randomly inside the region. Once the simulation begins, each node moves toward a randomly selected direction with a random speed ranges from 0 to 20 m per second. Upon reaching some randomly determined location, the node pauses for a fixed time, *pause time*, and proceeds again in a similar manner.

Transmissions between pairs of source and destination nodes, called *conversions*, are selected arbitrarily from 100 nodes according to the simulation setup. In our simulations, we first limit the number of conversation to a single pair and vary the pause time. We then vary the number of conversations while set the pause time to zero. The simulation time limit is set to 500 s with each simulation scenario representing an average of 30 test sample runs for the single conversation simulations and 5 test sample runs for the multiple conversations cases. Note that TORA is not included in the multiple conversations simulation due to its much worse performance compared to the other protocols.

### 5.1. Data delivery rate

Fig. 6 shows the simulation results in data delivery rate. According to Fig. 6(a), our protocol performs slightly better than (or as good as) AODV in term of packet delivery rate in the single conversation scenario, and both of our protocol and AODV perform much better than DSR and TORA. However, as we see in Fig. 6(b), when the numbers of conversation increase, the performance of DSR also catches up. The lead of our protocol remains in this aspect when the conversation pairs are kept within 30. But beyond that point, performance of our protocol seems to drop somewhat rapidly than the other two protocols. The reason for such dramatic turnover may be due to the fact that path repairing in the case of busy traffic results in a much busier MAC operations on nodes in the neighborhood of broken links, which induces more potential packet loss and reduces the delivery rate. On the other hand, AODV always tries to find an alternative path that usually comprises of nodes that are less busy and thus respond more promptly to route query request. As a result, the newly established routes usually contain less traffic and this may explain why AODV keeps performing well until the number of conversations exceeds 40.

When comparing with results presented in [20], we see that DSR performs better than QL protocol in case the number of conversation is less than 15, and falls slightly behind (around 3%) between 20 and 30, with the gap being gradually widen beyond that (See Fig. 9(b) in [20]). On the other hand, our protocol has about 3–4% lead over DSR in case the conversation pairs are under 30. Accordingly, based upon the observation, we may draw a conclusion that our protocol outperforms QL protocol when the traffic condition is lighter (number of conversation pairs $\leqslant 15$) and the situation turns around in the case of heavy traffic (number of conversation pairs $\geqslant 40$). While in between the two extreme conditions ($15 <$ number of conversation pairs $< 40$), the two protocols are roughly comparable.

### 5.2. Routing overhead

Routing overhead, in term of number of control packets sent, is presented in Fig. 7. We can see that TORA has a much higher control overhead than the other three protocols, and in most of cases (except when number of conversation pairs exceeds 30) our proposed protocol sends less control packets than that of AODV. This is not surprising since AODV, instead of repairing the partially broken main route, always reconstructs a new main route from scratch. As a result, every time a main route is broken, the main route request (MREQ) broadcast packets are flooded to the network. On the other hand, our protocol would try to repair the main route by taking advantage of the neighboring nodes that are close to the main route.

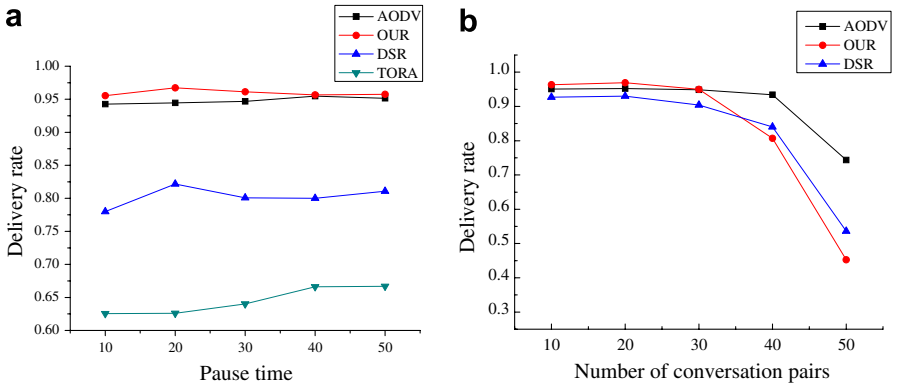



Fig. 6. Comparison in data delivery rate. (a) Single conversation with varying pause time. (b) Multiple conversations with zero pause time.

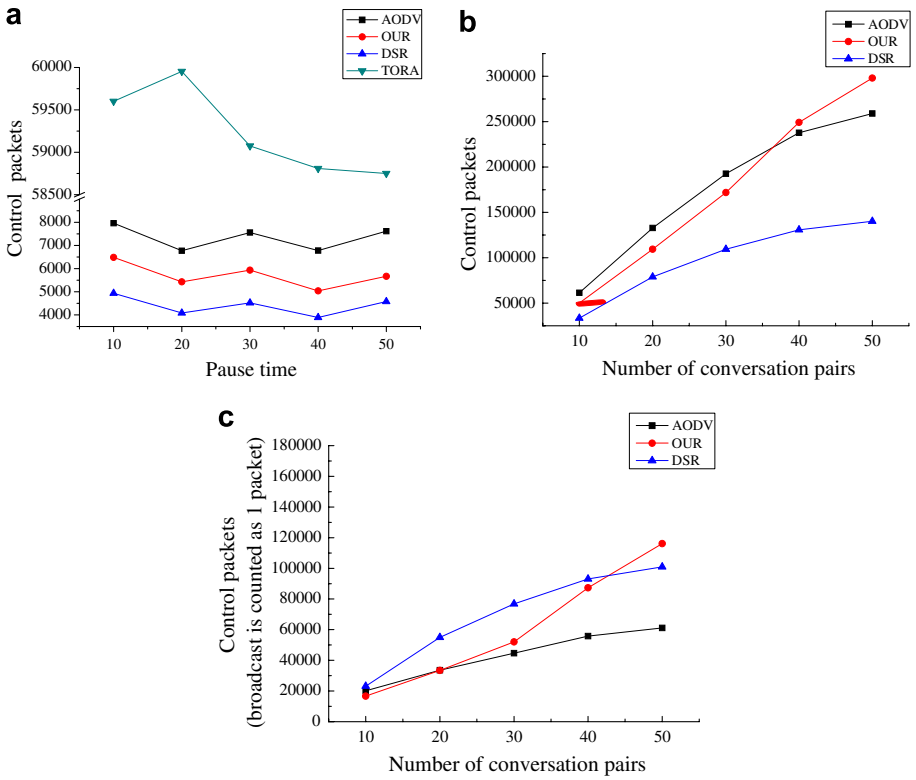



Fig. 7. Comparison in routing overhead. (a) Single conversation with varying pause time. (b) Multiple conversations with zero pause time. (c) Multiple conversations with zero pause time (broadcast is counted as one packet).

The scope of repair query (REPQ) packet broadcast is thus effectively restricted within two-hops from the original main route.

According to the results shown in Figs. 6 and 7, under light or moderate traffic condition, our protocol not only performs better than AODV in term of packet delivery rate, but also achieves that with lesser cost (approximately 20–25% less in number of control packets with single conversation case and 10–20% saving with conversations fewer than 30). With reason similar to that presented in Section 5.1, the repaired routes tend to be vulnerable under heavy traffic condition, which may result in repeated repairing operations and thus more additional control packets. On the other hand, DSR performs consistently better than both of ours and AODV protocols in this aspect, which may be attributed to the facts that (1) DSR establishes multiple paths from a given source to its destination, which are then cached in nodes, at route construction stage. Accordingly, nodes with DSR protocol tend to have a better chance in responding to route requests without further re-propagating these messages, and (2) DSR also adopts some local repairing technique (non-propagating route request).

For comparison to QL protocol, we need to refer to Fig. 7(c), where a broadcast packet is counted as only one control packet, which was the criterion taken in [20]. With careful evaluation, we see that our protocol uses less control packets than QL protocol when there are less or equal to 30 conversations and roughly the same in between 30 and 40 conversations (See Fig. 9(a) in [20]).

### 5.3. Communication delay

The communication latency, measured by the average data packet delivery delay per route in our simulation, is presented in Fig. 8. It is expected that our protocol would perform better than AODV and TORA in this aspect. The reason for such speculation is based on the fact that instead of starting from scratch every time a route is broken, our protocol would try to repair the broken main route with some substitute route around the main link. As a result, our protocol can recover from link failure more quickly in a lot of cases (as shown in Section 4), which results in the low communication latency. As we can see, the results, at least under light and moderate traffic conditions (number of conversation pairs $<30$), correspond exactly to our expectation. On the other hand, it seems that DSR protocol performs somewhat better than ours in most cases. However, the fact that the communication latency is calculated with data packets that are successfully received by the destination nodes only and DSR has a lower delivery rate make the difference not so significant.

In comparison to QL protocol, our protocol performs about the same in case the number of conversations is fewer than 20, but has longer delay than QL protocol beyond 20. The conclusion is drawn according to the fact that QL pro-

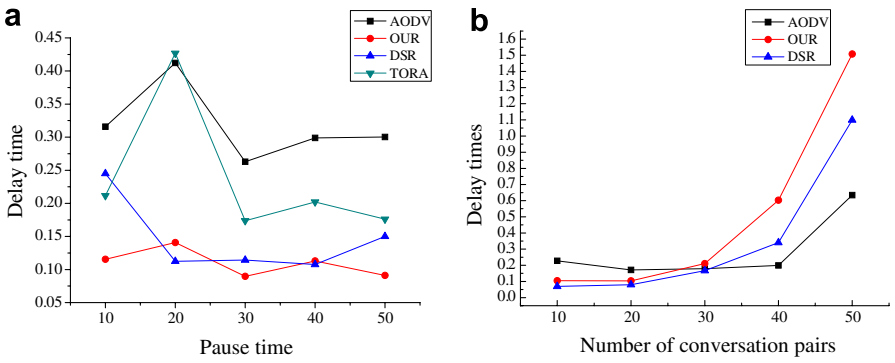



Fig. 8. Comparison in communication delay (in seconds). (a) Single conversation with varying pause time. (b) Multiple conversations with zero pause time.

tocol has slightly longer delay than DSR with conversations less than 20 and outperforms DSR since that point (See Fig. 9(c) in [20]).

### 5.4. Average number of hops

Fig. 9 shows the average number of hops that packets need to go through in order to reach the destination. The result indicates that our protocol in average requires up to 1 hop more than AODV and DSR take. The reason is also resulted from our protocol's attempt to repair the main route. A successful repairing action implies that packets now travel via an alternative route that is not as direct or straight as the original one, which contributes to the increase of the average hop count that a route traverses. On the other hand, both the AODV and DSR protocols always try to find a new main route, which tends to be a shorter one. In the case of TORA, if the upstream node of a broken link does not find any downstream node, it would reverse the direction of its incoming route and starts from there to find a new route to the destination, which may establish a potentially longer link as in our case.

Note that with number of conversations exceed 30, route repairing tends to be unsuccessful and packets that need to go through longer path often fail to reach their destination. In other words, packets that are successfully delivered to the destinations are probably ones that have fewer hop count between their source and destination nodes, which explains why the curves of our protocol and DSR drop more sharply than AODV.

## 6. Conclusion and future work

In this paper, we present a new on-demand routing protocol that is able to quickly repair a link or node failure with less communication overhead. Compared with the other notable on-demand routing protocols, our protocol has a higher successful data delivery rate than AODV and DSR, and saves approximately 10–25% in the number of control packets compared to AODV. In term of communication delay, our protocol also performs very well compared to the two major routing protocols, AODV and DSR. In addition, through indirect comparison, we also found that our protocol perform better than the other notable local repairing protocol, QL, in terms of data delivery rate and control overhead, under light and moderate traffic conditions. In other words, there may be conditions or threshold that performing a new route discovery would

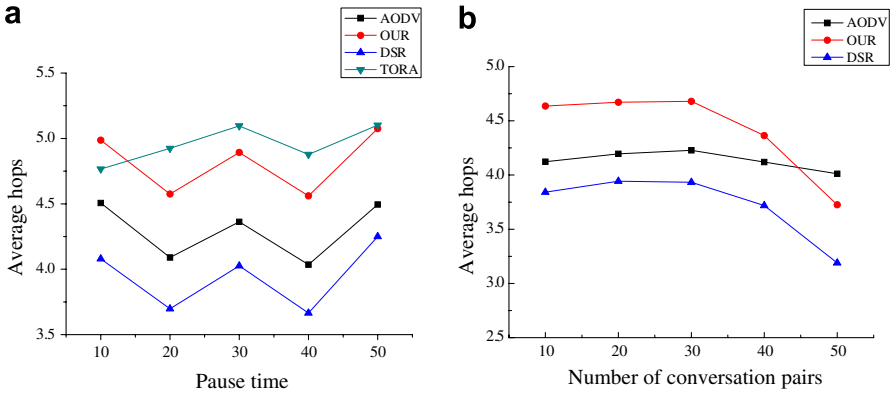



Fig. 9. Comparison in average number of hops of a route. (a) Single conversation with varying pause time. (b) Multiple conversations with zero pause time.

be more economic than repairing the existing main route. As a result, it would be desirable to modify our protocol so that it is capable of determining dynamically when to (or not to) initiate repairing mechanism. In addition, it would also be interesting to explore ways in evaluating among various potential alternate routes so that a more robust or reliable backup route can be chosen under different network conditions or environments. As a result, in the future, we will be working on increasing the data delivery rate and improving other performance metrics through a route maintenance mechanism that can dynamically determine when to initiate the route repairing process and how to select a more reliable alternative route. To be more specific, the alternative route construction process could be initiated at any time, not just when a route is broken. The dynamically constructed alternative routes information can be passed to the upstream nodes, which then determine by themselves when to direct their packets to the ''optimal'' alternative route. Finally, we will also be exploring the possibilities in applying similar idea to the sensor network environment.

## References


[1] E.M. Royer, C.-K. Toh, A review of current routing protocols for ad hoc mobile wireless networks, IEEE Personal Communication (1999) 46–55.

[2] C.E. Perkins, P. Bhagwat, Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers, in: ACM SIGCOMM, 1994, pp. 234–244.

[3] C.C. Chiang, Routing in clustered multi-hop, mobile wireless networks with fading channel, in: Proceedings of IEEE SICON, 1997, pp. 197–211.

[4] S. Murthy, J.J. Garcia-Luna-Aceves, A routing protocol for packet radio networks, in: ACM MOBICOM, 1995, pp. 86–94.

[5] C.E. Perkins, Ad hoc on demand distance vector (AODV) routing, in: IETF Internet-Draft, draft-ietf-manet-aodv-00.txt, November 1997.

[6] C.E. Perkins, E.M. Royer, Ad-hoc on-demand distance vector routing, in: Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications, 1999, pp. 99–100.

[7] J. Broch, D.B. Johnson, D. Malt, The dynamic source routing protocol for mobile ad hoc networks, in: IETF Internet Draft, draft-ietf-manet-dsr-00.txt, March 1998.

[8] D.B. Johnson, D.A. Maltz, Dynamic Source Routing in Ad-Hoc Wireless Networks, in: T. Imielinske, H. Korth (Eds.), Mobile Computing, Kluwer, Dordrecht, 1996, pp. 153–181.

[9] M.S. Corson, V.D. Park, Temporally ordered routing algorithm (TORA) Version 1: functional specification, Internet-Draft, draft-ietf-manet-tora-spec-00.txt, November 1997.

[10] V.D. Park, M.S. Corson, A highly adaptive distributed routing protocol for mobile ad hoc networks, in: Proceedings of INFOCOM, 1997, pp. 1405–1413.

[11] C.-K. Toh, Associativity based routing for ad-hoc mobile networks, Wireless Personal Communication 4 (2) (1997) 103–139.

[12] Z.J. Haas, The zone routing protocol (ZRP) for ad-hoc networks, IETF Internet Draft, draft-zone-routing-protocol-00.txt, November 1997.

[13] Chang Wu Yu, Tung-Kuang Wu, Rei Heng Cheng, Po Tsang Chen, A low overhead ad hoc routing protocol with route recovery, in: Proceedings of the International Conference on Mobile Ad-hoc and Sensor Networks (MSN), 2005, pp. 666–675.

[14] Ajay Agarwal, Bijendra Jain, QoS-based on-demand segmented backup routing in mobile ad hoc networks, Proceedings of the Twelfth IEEE International Conference on Networks, vol. 1, 2004, pp. 331–335.

[15] S.-J. Lee, M. Gerla, AODV-BR: backup routing in ad hoc networks, IEEE Wireless Communications and Networking Conference (WCNC) 3 (2000) 23–28.

[16] Hsing-Lung Chen, Chein-Hsin Lee, Two hops backup routing protocol in mobile ad hoc networks, in: Proceedings of Eleventh International Conference on the Parallel and Distributed Systems, vol. 2, 2005, pp. 600–604.

[17] M.H. Jiang, R.H. Jan, An efficient multiple paths routing protocol for ad-hoc networks, in: Proceedings of Fifteenth IEEE International Conference on Information Networking, 2001, pp. 544–549.

[18] D. Maltz, J. Broch, J. Jetcheva, D. Johnson, The effects of on-demand behavior in routing protocols for multi-hop wireless ad hoc networks, IEEE Journal on Selected Areas in Communication 17 (8) (1999) 1439–1453.

[19] C.M. Chung, Y.H. Wang, C.C. Chuang, Ad hoc on-demand backup node setup routing protocol, Proceedings of Fifteenth IEEE International Conference on Information Networking, 2001, pp. 933–937.

[20] R. Castañeda, S.R. Das, M.K. Marina, Query localization techniques for on-demand routing protocols in ad hoc networks, Wireless Networks 8 (2002) 137–151.

[21] Xuhui Hu, Yong Liu, Myung J. Lee, Tarek N. Saadawi, Route update and repair in wireless sensor networks, IEEE Consumer Communications and Networking Conference (2004) 82–87.

[22] Xuhui Hu, Yong Liu, Myung J. Lee, Tarek N. Saadawi, Efficient route update protocol for wireless sensor network, IEEE Military Communications Conference 1 (2003) 549–554.

[23] M.D. Penrose, Random Geometric Graphs, Oxford University Press, Oxford, 2003.

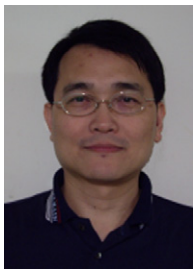
[24] C. W. Yu, L.-H. Yen, Yang-Min Cheng, Computing subgraph probability of random geometric graphs with applications in wireless ad hoc networks, Tech. Rep., CHU-CSIE-TR-2004-005, Chung Hua University, ROC.

[25] Kevin Fall, Kannan Varadhan, Ns notes and documentation, The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, November 1999. Available from: <http://www-mash.cs.berkeley.edu/ns/>.

[26] J. Broch, D.A. Malt, D.B. Johnson, Y.C. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, in: Proceedings of MOBICOM, 1998, pp. 85–97.

[27] S.R. Das, C.E. Perkins, E.M. Royer, Performance comparison of two on-demand routing protocols for ad hoc networks, in: Proceedings of INFOCOM, 2000, pp. 3–12.


**Chang Wu Yu** received the BS degree from Soochow University in 1985, MS degree from National Tsing Hua University in 1989, and PhD degree from National Taiwan University in 1993, Taiwan, all in computer sciences. From 1995 to 1998, he was an Associate Professor at the Department of Information Management, Ming Hsin Institute of Technology. In 1999, he joined the Department of Computer Science & Information Engineering, Chung Hua University. His current research interests include graph algorithms, wireless networks, and distributed computing.



**Tung-Kuang Wu** received his PhD degree in computer engineering from the Department of Computer Science & Engineering at Pennsylvania State University in 1995. He is currently an associate professor in the Department of Information Management at National Changhua University of Education, Changhua, Taiwan. His current research interests include wireless communication, special education technologies, and e-Learning.

**Rei Heng Cheng** received the MS and PhD degrees in Computer Science and Information Engineering from National Chiao Tung University, Taiwan, R.O.C. in 1989 and 1995 respectively. Since 1999, he has been on the faculty of Hsuan Chuang University, HsinChu, Taiwan. He is currently an Associate Professor at the Department of Computer Science. His research interests include wireless and sensor networks, image processing, and character recognition.