Hop ID: A Virtual Coordinate-Based Routing for Sparse Mobile Ad Hoc Networks

Yao Zhao, Yan Chen, Bo Li, and Qian Zhang

Abstract—Routing in wireless communication systems such as ad hoc networks remains a challenging problem given the limited wireless bandwidth, users' mobility, and potentially large scale. Recently, a thrust of research has addressed these problems—the on-demand routing, geographical routing, and virtual coordinates. In this paper, we focus on geographical routing that has been shown to achieve good scalability without flooding; however, this usually requires the availability of location information and can suffer from poor routing performance and severe dead end problems, especially in sparse networks. Specifically, we propose a new Hop ID routing scheme, which is a virtual coordinate-based routing protocol and does not require any location information. This achieves excellent routing performance comparable with that obtained by the shortest path routing schemes. In addition, we design efficient algorithms for setting up the system and adapt to the node mobility quickly and can effectively route out of dead ends. Extensive analysis and simulation show that the Hop ID-based routing achieves efficient routing for mobile ad hoc networks with various density, irregular topologies, and obstacles.

Index Terms—Mobile ad hoc networks, routing protocols, virtual coordinate.

1 Introduction

Due to the limited spectrum, user's mobility, and power constraints, routing remains a challenge, particularly in wireless communication systems such as ad hoc networks and sensornets. Several other challenges complicate routing, including scalability, routing efficiency, adaptation to wireless networks of various densities, and distribution. For instance, scalability is difficult to achieve in the wireless environment because it lacks the inherent hierarchy in the address structure. That is, in an ad hoc network or sensornet, two neighboring nodes might have completely different addresses and/or identifiers.

Also, as most wireless devices use batteries to consume power, efficient routing paths are critical, especially for sensornet. Note that, although some sensornet applications use many-to-one or one-to-many routing, many applications still require point-to-point routing [24], [25], [26], [27], which is the focus of this research.

There are mainly two types of proposals especially designed for wireless routing to improve scalability: on-demand routing protocols [1], [2] and geographical routing schemes [6], [7], [8]. On-demand routing does not require any prior processing for route establishment; instead, it uses route request flooding to all nodes in the network in order to establish the route on-demand. This often relies on the computation of the shortest path between a source and a destination and tends to work well for small or moderately sized systems with relatively stable routes. However, such schemes do not scale well due to significant overheads in terms of both delay and flooding in large networks.

The basic idea in geographical routing to use a node's location as the address and forward packets based on a predefined routing metric, usually the geographic distance. The greedy nature comes from the fact that such algorithms usually forward packets only based on the decrease of this metric in each hop without taking into account more complete topological information. The geographical routing achieves excellent scalability in that each node only needs to be aware of the neighbors' location information and does not rely on flooding to exploit network topology. However, there is one serious limitation for geographical routing: the dead end problem, especially under low density environment or scenarios with obstacles or holes. The dead end problem is caused by the inherent greedy nature of the algorithm in that a packet may get stuck at a local optimal node that appears closer to the destination than any of its known neighbors under the predefined routing metric. Recently, virtual coordinates were proposed for geographic routing without location information [15], which, however, suffer the same dead end problem. The routing success rate of the greedy routing with such virtual coordinates drops quickly as the network density becomes sparser [15]. Thus, efficient geographic routing in sparse networks remains a challenging problem.

In this paper, we aim to design new routing protocols to solve the dead end problem without sacrificing routing efficiency, even for a sparse ad hoc network with various topologies and obstacles. For routing efficiency, we seek the shortest path route performance as that of the on-demand routing. To the best of our knowledge, we are among the *first* to achieve all these properties in one system with small overhead.

- *Y. Zhao and Y. Chen are with the Electrical Engineering and Computer Science Department, Northwestern University, Technical Institute, Room L359, 2145 Sheridan Road, Evanston, IL 60201. E-mail: {yzhao, ychen}@cs.northwestern.edu.*
- *B. Li and Q. Zhang are with the Computer Science Department, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, China. E-mail: {bli, qianzh}@cs.ust.hk.*

Manuscript received 14 Nov. 2005; revised 17 Apr. 2006; accepted 31 Oct. 2006; published online 7 Feb. 2007.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0335-1105.
Digital Object Identifier no. 10.1109/TMC.2007.1042.

We propose a novel routing algorithm utilizing a new virtual coordinate, called *Hop ID*. Each node maintains a Hop ID, which is multidimensional coordinates assigned based on its distance to some landmark nodes randomly selected in the network. With a predefined distance function, two nodes can calculate the "distance" between them. Based on this Hop ID metric, the routing algorithm performs greedy forwarding, similar to the geographic forwarding, i.e., a node forwards the packet to a neighbor that is "nearest" to the destination in the Hop ID space. But, in contrast to traditional geographical routing, such schemes effectively avoid the dead ends, even for a sparse network. The intrinsic reason is that the Hop ID coordinates are constructed based on the topology, and the virtual coordinate distance between any two nodes is very close to the shortest path length between them.

In addition, we design an efficient landmark selection algorithm. These landmarks are random nodes in the network. The number of landmarks remains constant even for a very large network. We further propose a novel landmark-guided detour scheme that can effectively route out of a small number of remaining dead ends.

The main contribution of this paper is a novel virtual coordinates system (Hop ID) and its greedy routing algorithm, which has the following key features:

1. no geographic location support is needed,
2. performs well in sparse ad hoc networks, and
3. scales well to large ad hoc networks.

The rest of the paper is organized as follows: In Section 2, we discuss related works. In Section 3, we present the design of Hop ID routing and evaluate its performance in Section 4. We conclude the paper and highlight several possible avenues for further study in Section 5.

2 Related Works

Before we present the Hop ID routing algorithm, we first describe the main motivations and put them in the proper context with the related works. Routing is a recursive procedure to forward packets increasingly "closer" to the destination. The most critical component in any routing algorithm is how to measure the "distance" between two nodes. This distance metric determines the route performance to a large degree, yet how to select this metric is nontrivial. Since packets are forwarded on a hop-by-hop basis, hop count or the shortest path distance is a natural candidate, but it requires significant overhead to find and maintain the shortest path. The on-demand routing algorithm [1], [2] and proactive routing protocols [3], [4], [5] are typical examples that use *hop distance*—the length in hops of the shortest path between a pair of nodes—as the routing metric.

Other metrics have been proposed to measure the "distance" between two nodes such as geometric distance, last encountered time [13], and ID space distance [14]. In geographic routing, which uses geometric distance as the distance metric, each node forwards a packet to a neighbor with a shorter distance to the destination. Geographic routing does not incur explicit route discovery using flooding; instead, it only requires obtaining the position of the destination and neighbors. Geographic routing consists of three parts: 1) greedy routing algorithm, 2) dead end resolution, and 3) location service. The existence of a dead end is a well-known problem for geographic routing, in which pure greedy algorithms hardly work in sparse networks or scenarios with obstacles or holes. Many protocols, such as GPSR/GFG [6], [7], used a face routing technique to overcome the dead end problem, but that usually requires a much longer routing path. GOAFR+ [8] attempted to enhance face-routing performance. In sparse networks, the fundamental problem in geographic routing is that geometric distance can hardly reflect the true hop distance between two nodes, often leading to the dead end problem. Face routing mitigates this problem at the cost of a longer routing path. In fact, the routing path can be several times longer than that of the shortest path length [8].

Another limitation of geographic routing is that it requires GPS or another location device to obtain precise location information. For geographic routing, an exact location might not be necessary and imprecise virtual coordinates accordant with the network topology may perform better than the real coordinates system. Under such motivation, Rao et al. recently made a fine attempt at geographic routing without location information [15]. They proposed a virtual coordinate construction algorithm which achieved comparable performance with the real geometric coordinates in dense networks. They also demonstrated in [15] that the virtual ordination has potential in the environment with obstacles or holes, as virtual coordinates can better reflect the connectivity than real coordinates. But, the approach in [15] performs badly in sparse networks because its greedy success rate drops quickly and the dead end problem reappears. We compare it with our Hop ID schemes in detail in Section 4.4. GEM [16] is another virtual coordinate-based routing for sensornet, which has no dead end problems. However, GEM still suffers from poor routing performance in sparse networks and can incur significant overhead under node and network dynamics.

3 Efficient Routing with Hop ID

To design a scalable and efficient routing scheme for mobile ad hoc networks, we observe that a well predefined distance metric in geographical routing, so that minimal to no flooding is necessary to explore the route discovery, is the key to obtain scalability. On the other hand, the accuracy of the predefined distance metric representing the hop distance determines the route performance. In other words, if the greedy metric can more accurately reflect the hop distance, the route performance will be closer to that of the shortest path routing. The existing geographical routing algorithms work poorly in sparse networks or scenarios with obstacles or holes. In these scenarios, the correlation between the geometric distance and hop distance is weak, which results in significantly more dead ends and unnecessarily longer route paths.

To address these problems, we present Hop-ID-based routing. We construct a multidimensional coordinates system, called a *Hop ID system*, and use corresponding distance function to calculate the Hop ID distance between a pair of nodes. A node's position, i.e., its Hop ID, is a vector in which each dimension is the hop distance from the node to a preselected landmark node. The Hop ID distance

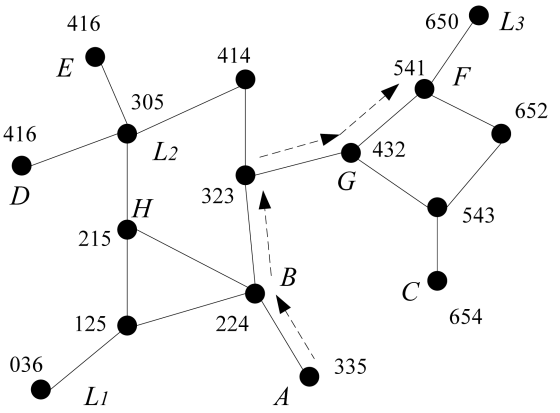

Fig. 1. Example of Hop ID. A node N's Hop ID xyz means N is x, y, z hops away from landmark L_1, L_2, and L_3, respectively.

(vector) between two nodes is calculated from the relative hop distances to the set of landmarks. The results demonstrate that the Hop ID distance closely resembles the hop distance. In addition, Hop ID construction has no requirement on the density of the network, thus, the routing protocol works well under both high and low density environments. Compared to existing proposals, in particular, the virtual coordinates in [15], our proposed Hop ID system performs comparably well in dense environments and performs significantly better in sparse networks.

Hop ID routing is also one type of geographic routing, thus, it requires the careful design of the three parts identified in Section 2. First, Hop ID routing uses the greedy routing based on a prespecified distance metric; we construct multidimensional virtual coordinates—the Hop ID system—that rely on the elected landmark nodes to compute the Hop ID distance between a pair of nodes. Second, the dead end problem exists, but is considerably less severe than the current geographic routing scheme due to its insensitivity to the network density. We present effective techniques to solve this problem. Finally, a location service is needed for the source node to get the Hop ID of the destination. Helmy et al. [9], Camp et al. [10], Li et al. [11], and Bose et al. [12] extensively studied the location service, and many known techniques such as [9] and [10] can be used in the Hop ID system. We propose a simple location service as follows: Each node n randomly uses a predefined hash function to hash its IP address to one of the landmarks, which will serve as the location server for n. The routing to landmark nodes is always known to all nodes and each node updates its Hop ID with its location server when necessary. This will be further elaborated in Section 3.4.

3.1 Hop ID Description

We use the example in Fig. 1 to illustrate the basic ideas in the Hop ID system. We assume that some nodes have already been selected as landmark nodes by the landmark selection algorithm introduced in Section 3.3.1 and each node knows its hop distance to all the landmarks. In Fig. 1, L_1, L_2, and L_3 are three landmarks. Following a predefined order, the hop distance of a node to all the landmarks is combined into a vector, i.e., the node's Hop ID. For example, L_2's Hop ID is 305 in Fig. 1, representing that L_2 is three hops away from L_1, zero hop away from itself, and five hops away from L_3.

Intuitively, the Hop ID can reflect the proximity of the network to some extent. Taking two nodes, N_1 and N_2, for example, we define the hop distance between N_1 and N_2 as L_h. Assuming there are m landmark nodes, the Hop ID of N_1 is $H^{(1)}$ $(H_1^{(1)}, H_2^{(1)}, \cdots, H_m^{(1)})$, and the Hop ID of N_2 is $H^{(2)}$ $(H_1^{(2)}, H_2^{(2)}, \cdots, H_m^{(2)})$, the following triangulation inequality holds:

$$\underset{k}{Max}\left(|H_k^{(1)} - H_k^{(2)}|\right) \le L_h \le \underset{k}{Min}\left(H_k^{(1)} + H_k^{(2)}\right). \quad (1)$$

For each k from 1 to m, L_h is no more than the sum of $H_k^{(1)}$ and $H_k^{(2)}$ since there exists a path from N_1 to N_2 via landmark k and the hop count of this path is $H_k^{(1)} + H_k^{(2)}$. For the left part of the inequality, without losing generality, we assume that $H_k^{(1)}$ is no more than $H_k^{(2)}$, $H_k^{(2)}$ is no more than the sum of L_h and $H_k^{(1)}$ because there is a path from landmark k to N_2 via node N_1, and $H_k^{(2)}$ is the shortest hop distance from landmark k to N_2. These inequalities yield a lower bound L and an upper bound U of L_h. More landmark nodes can make the lower and upper bounds tighter, but as we can see from the discussion in Section 3.2, the number of landmarks actually needed is a constant determined by the precision requirement and not affected by the number of nodes in the network.

3.2 Distance Function

One of the key problems is what distance function is most efficient for greedy routing. We seek a distance metric calculated by Hop IDs, and such a distance is an accurate estimation of the hop distance.

Recently, Kleinberg et al. [22] studied this problem in a theoretical manner and introduced the following theorem:

Theorem 1. *In any s-doubling metric M, a constant number of randomly selected landmarks achieve an (ε, δ)-triangulation with probability $1 - \gamma$, where the constant depends on δ, ε, γ, and s.*

The (ε, δ)-triangulation means that, for all but an ε fraction of the pairs (u, v), we have $U/L < 1 + \delta$ for a metric holding a triangulation inequality. Like euclidean space, a ball is defined as all the nodes that are no farther than a radius away by a certain metric. A metric is an s-doubling metric if every ball can be covered by at most s balls of half the radius [22]. Obviously, the metric of the hop distance is an s-doubling metric, while s is related to the density of the network. Equation (1) shows that the hop distance satisfies the triangulation property. Based on this theorem, given a constant number of randomly selected landmarks, we can achieve $U/L < 1 + \delta$ for all but an ε fraction of node pairs with probability $1 - \gamma$. In other words, the lower bound and upper bound of hop distance can be quite precise if we have a sufficient number of landmarks. This motivates us to use U or L as the metric for greedy routing.

However, we found that U is not a good metric for greedy routing. For example, for two nearby nodes N_1 and N_2, U usually is not a good estimation of the hop distance between these two nodes since the closest landmark to

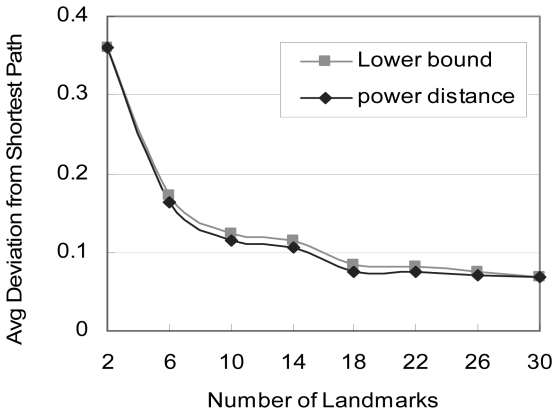

Fig. 2. Distance functions versus shortest path.

them may be even farther than the distance between N_1 and N_2. In fact, this is precisely the reason that there are an ε fraction of node pairs for which the precise estimation is not available as mentioned in the theorem. As a packet is routed closer and closer to the destination, it will suffer from the imprecise distance if U is chosen as the distance function. However, L does not have this problem. Is L then a good distance function?

Using the landmark selection algorithm described in Section 3.3.1, our simulation shows that the lower bound L is very close to the shortest path L_h. Fig. 2 illustrates the relationship between L and L_h as a function of number of landmark nodes. The network has 3,200 nodes distributed uniformly in a square and the density is 3π (see Section 4.1 for the detailed setup of the experiments). There are $N = 40{,}000$ paths and (u, v) are measured in each round of the simulation. The deviation of L away from L_h can be calculated as

$$E = \sqrt{\sum_{(u,v)} (1 - L/L_h)^2 / N}. \quad (2)$$

As shown in Fig. 2, with only a few landmarks, the deviation of L from L_h is very small. For example, when there are 18 landmarks in these 3,200 nodes, the average deviation is less than 0.1, where L_h is about 23 hops on average. Although L is a good distance metric that can fairly accurately estimate the shortest path, it is not a practical greedy metric for the following reason: L is discrete and it can easily cause a dead end if L itself is not exactly the same as the hop distance L_h. More specifically, for a destination d, a node may easily get a tie when comparing its L with L of its neighbors. For example, node B and H are of same distance to node D in Fig. 1. But node H seems to be closer to D because the Hop ID of H is no farther from D than B for each dimension of Hop ID. Thus, we add some modification to L to utilize more information from Hop ID and obtain a continuous distance value for distance estimation. Finally, we choose the power distance metric:

$$D_p = \sqrt[p]{\sum_{k=1}^{m} |H_k^{(1)} - H_k^{(2)}|^p}. \quad (3)$$

Specifically, when $p = \infty$, D_p is equivalent to L. When $p = 2$, D_p is the euclidean distance. Obviously, when p is reasonably large (e.g., 10), the value of D_p is mainly determined by L (see Fig. 2—the deviation of D_p is quite close to and even less than L). But, unlike the lower bound L, the power distance D_p has continuous values, which helps to break the ties and eliminate many dead ends. We had a sensitivity test of p over a range of 5, 10, 15, and 20. We found that the performance is good and similar when p is larger than or equal to 10. Thus, we choose p as 10 in all the simulation experiments in this paper.

3.3 Landmark Selection

Based on Theorem 1, it is straightforward to come up a simple landmark selection algorithm: random landmark selection. However, efficiently selecting a certain number of random landmarks when none of the nodes has a global view of the whole network is nontrivial. In this section, we first propose a distributed random landmark selection algorithm. Note that random landmarks may not be very efficient. Then, what nodes are better candidates for landmarks? We further investigate this problem and propose a peripheral landmark selection scheme.

3.3.1 Random Landmark Selection

Based on Theorem 1, we first select landmarks randomly. A simple way is to use some hash function to select landmarks randomly. For example, if we need m landmarks, we can simply generate m random IDs for landmark selection, called landmark IDs. Each node has its own unique ID that can be hashed from the IP address or any other unique number of a node. For each node, if its ID is the closest one to a landmark ID, it becomes a landmark.

This is much easier to accomplish if we can deploy an ad hoc network from scratch. However, we often have to set up the routing system with a deployed ad hoc network, such as in the battlefields. To this end, we design a more efficient algorithm to randomly select m landmarks for an existing ad hoc network based on the hashing idea. To prevent the overhead by nodes competing for landmarks, a coordinator node C is first selected to manage the landmark selection process. C can be any node that is relatively stable.

1. Build the shortest path tree. Node C generates m random landmark IDs and then floods to the network a CENTER packet including these m IDs. Every node adds its upstream node ID when it rebroadcasts the CENTER packet and, thus, the upstream node knows its downstream children. Thus, through the flooding, we can build a shortest path tree with a root as C. Note that we may not get an absolute shortest path tree because of the lossy wireless channel, but this will not introduce problems as we choose landmarks randomly.
2. Aggregate landmark candidates. This process starts from the leaves of the shortest path tree. It is simple for a node N to determine whether or not it is a leaf node: If no other node claims N as its upstream node, N is a leaf node. With the assumption that there is not much data transmission before the routing is set up,

selecting a reasonable timeout for node N to believe all its neighbors have rebroadcast the CENTER packet is easy. The leaf node N will send a CANDIDATE packet to its upstream nodes, in which it selects itself as the landmarks for each landmark ID. The upstream node will collect all the CANDIDATE packets from its children and find the best candidate (the closest one) for each landmark ID. Iteratively, the upstream node reports to its upstream and, at last, the coordinator C will get the best candidates of the whole network. Again, to avoid endless waiting for a report from its children, a nonleaf node can set an expiration time or even actively query all children. This bottom up report scheme for landmark selection is very efficient. Ideally (without packet loss), each node only needs to send one packet containing m landmark candidates.

3. Inform landmarks. At the end of Step 2, the coordinator C finds the m best landmark candidates. Now, it needs to inform them. Here, node C only needs to send m packets instead of a packet flooding. Since each nonleaf node N aggregates the candidate recommendation from its subtree and selects the best candidates, node N knows the child nodes from which each of node N's candidates are recommended. Using only $O(m)$ memory for each of the nonleaf nodes, the algorithm actually builds m inverse paths from C to the landmarks and, thus, eliminates the use of flooding.
4. Build Hop ID. After receiving the notification from the coordinator C, each landmark node floods a LANDMARK packet to the network with its landmark ID. On receiving LANDMARK packets, each node records its hop distance to the corresponding landmark to compose its Hop ID. After the LANDMARK flooding is complete, every node sets up its Hop ID.

Optimization for Step 2. The purpose of random landmark selection is to have landmark nodes distributed more uniformly in the network. When m is not very large (usually $m \leq 30$ with the current scale of ad hoc networks), randomly selected landmarks may not distribute uniformly in the network. Next, we apply some optimization to improve the random landmark selection algorithm described in Step 2.

We collect one more metric—subtree size in the shortest path tree. For a nonleaf node N, its subtree size is the number of nodes that have N on their shortest path to the coordinator. When the landmark candidates are reported from the leaf nodes in a bottom up manner, the subtree size can be obtained in a similarly recursive manner. Thus, when the coordinator node C selects landmarks from all the candidates provided by C's neighbors, C can take the subtree size into account and select landmarks from each subtree proportionally to the size of the subtree. For example, even if a small subtree has a relatively large number of landmark candidates whose IDs are very close to the landmark IDs, the number of real landmarks chosen from this subtree is still proportional to its size, i.e., relatively small.

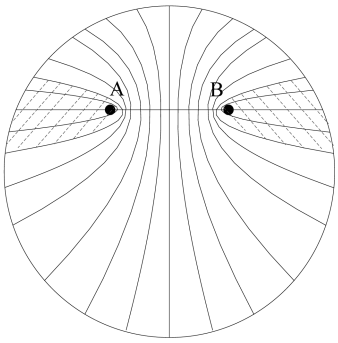

Fig. 3. The hyperbola with focus A and B.

3.3.2 Peripheral Landmark Selection

Although selecting landmarks randomly is simplest, usually random selection is unlikely to be the best option. In this section, we explore other possible landmark selection schemes. To simplify the problem, we adopt an *ideal dense model* to illustrate our intuition. In this simple model, we assume the geographic distance between two nodes equals the hop distance (or within a constant factor). This assumption is reasonable when the network is very dense and each node has a similar communication range.

Assuming D_p is close to L, as shown in Section 3.2, we use L as the basis for analysis instead of D_p due to its complicated nature. Fig. 3 shows a hyperbola graph and nodes A and B are two foci. So, for each node N on a line, the value of $(|NA| - |NB|)/|AB|$ is a constant. Given the threshold of estimation accuracy, finding out the region where a landmark should be (e.g., the shaded area for 95 percent accuracy) is easy. As node A or B gets closer and closer to the perimeter of the circle, i.e., the boundary of the network, the shaded area is more and more confined to the border area. Thus, landmarks on the perimeter have a high probability to be the best ones.

In a realistic random network, the contour line of the graph is distorted, but we believe that the perimeter nodes remain more advantageous. Again, simulation results confirm our analysis for dense and sparse networks.

By simple modification of the random landmark selection algorithm, we propose the following peripheral landmark selection approach:

1. Detect the perimeter of the network. In reality, we need not acquire the precise perimeter of the network. Borrowed from [15], initially, a coordinator node C floods a CENTER packet to the entire network. Hence, every node finds its distance to C and all its neighbors' hop distance to C. If a node finds that its hop distance is no smaller than all its neighbors' hop distances to the voluntary node C, this node becomes a perimeter node. Note that we build the shortest path tree rooted at node C at the same time.
2. Similar steps to random landmark selection. Only the perimeter nodes will probably be landmarks. The remaining steps are similar to the random landmark selection, except that only perimeter nodes are considered for landmarks.

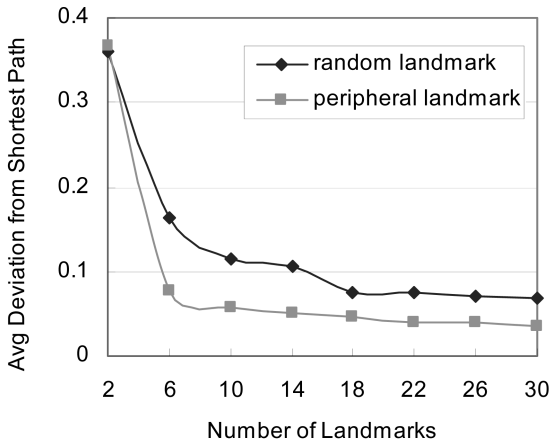

Fig. 4. Random landmark versus peripheral landmark.

By simple modifications, we can randomly select landmarks from the perimeter of the networks. Fig. 4 shows the comparison on hop distance estimation between the two landmark selection algorithms, random landmark selection and peripheral landmark selection. Using the same setup as Fig. 2, we can clearly see that a peripheral landmark provides a more accurate estimate. This simulation result confirms this conjecture that peripheral landmark selection is more advanced, as we described in Section 3.3.2.

3.3.3 The Choice of Landmark Selection Algorithm

Peripheral landmark selection proves to be more appealing, as it requires fewer landmarks than the random landmark selection algorithm. However, it has its own limitations. In a wireless system with high mobility, all the nodes including landmarks can move rapidly and continuously change their location. So, the peripheral landmark selection algorithm will degrade to the random landmark selection algorithm if the mobility of nodes is random (e.g., the random way point model). It is possible to keep landmarks on the border by frequent reselections. However, reselection will introduce overhead and make the algorithm more complicated.

On the other hand, the random landmark selection algorithm seems to be more robust to mobility. Thus, the users can select a suitable landmark selection algorithm based on their applications. If the network is known to be a sensornet, which usually has slow or even no mobility, we use the peripheral landmark selection algorithm. Otherwise, random landmark selection is used by default.

3.4 Hop ID Adjustment

Once the landmark selection procedure is completed, each node obtains a Hop ID. In the mobile environment, nodes can move, and the Hop ID has to be periodically updated to reflect the topology changes. One straightforward way is to let each landmark node flood periodically, and then every other node gets its updated Hop ID. Apparently, it is not scalable due to the significant flooding cost.

Here, we propose a Hop ID adjustment algorithm, which applies the distance vector routing principle. The algorithm only utilizes the periodic HELLO messages, which are originally used for collecting and maintaining the Hop IDs of each node's neighbors.

Assume that, at T_0 time, node N broadcasts a HELLO message. Node N first calculates its new Hop ID and then broadcasts the new Hop ID in the HELLO message. Assume that N has n neighbors, $N_1, N_2 \ldots N_n$, and neighbor N_i's Hop ID is $(H_1^{(i)}, H_2^{(i)}, \cdots, H_m^{(i)})$. For N's new Hop ID $(H_1, H_2, \cdots, H_m)$, we have

$$H_i = \begin{cases} 0 & \text{if } N \text{ is the } i\text{th landmark} \\ \underset{1 \le k \le m}{Min}\,(H_i^{(k)}) + 1 & \text{otherwise.} \end{cases} \quad (4)$$

In fact, it is a variant of distance vector routing, calculating the hop distance of all the nodes to the landmark nodes. Standard distance vector routing has two problems: slow convergence and "counting to infinity." As RIP [20] says, we use "split horizon with poisoned reverse" to make the convergence process fast. In the transition stage, the Hop ID of a node may not be very precise, but the distance function can tolerate such errors in the Hop IDs as the greedy algorithm does not rely on the absolute value of distance (see Section 3.6). In some cases, the inaccurate Hop IDs do cause some failure of the greedy routing algorithm. The dead end remedy algorithms introduced in Section 3.7 can finally solve the problem. If the unreachable landmarks can be identified and replaced quickly, the "counting to infinity" problem disappears. In Section 3.5, we will describe our algorithms to manage the landmarks.

Each node N stores the latest Hop ID sent to the location server. After adjustment, if the Hop ID distance between its new Hop ID and latest reported Hop ID is larger than a threshold t, N needs to send an update to its associated location server (in our simple scheme, the corresponding landmark). Here, we select t as 2 for a good balance between the stability and routing adaptation of the system.

The Hop ID adjustment algorithms are quite efficient and cost only the periodical exchange of HELLO messages with neighbors. Part of our future work will include studying the more dynamic scenarios where nodes join or leave the network, but we expect more similarity between these more dynamic scenarios and mobile-only scenarios. The problem may occur when a landmark node leaves without any notification, which is studied in the next section.

3.5 Landmark Management

Landmarks are relatively important nodes in the network, as they maintain the Hop ID coordinates. It is a classic problem in distributed systems to elect and maintain some coordinating nodes. We use the following algorithm to maintain a certain number of active landmarks in the network:

- *Dealing with landmark failure or leaving*. A coordinator among the landmarks is elected to be responsible for managing the landmarks. The coordinator will use PING/PONG messages to query the existence of all other landmarks with certain frequency. When the coordinate finds that a landmark leaves the network, the coordinator uses the initial landmark selection algorithm to select a new landmark. When the

coordinator itself fails, the landmarks will reelect the coordinator through the classic ring election or bully algorithms [21], which are proven to be robust.

- *Dealing with network partition*. The HELLO message of a landmark includes a timestamp, indicating the new update from the landmark. When nodes receive or send HELLO messages, they record or include the latest timestamp of the landmark nodes. When a node finds out that it has not received a new update from the landmarks for a long time (e.g., three times the average interval between two updates of a landmark receive before), the node realizes that there is a network partition and maybe no landmark exists in its partition. New landmarks can be elected as the network is initialized.

3.6 Hop ID Greedy Routing Algorithm

The greedy routing algorithm is similar to that of geographic routing and the main difference is in the choice of distance function. We make several assumptions. First, we assume that the source knows the destination node's Hop ID and the Hop ID is included in the packet header of each data packet. Thus, we need a Hop ID lookup service, which was discussed in the beginning of Section 3. Second, each node knows the Hop ID of its neighbors or, even more aggressively, its 2-hop neighbors. This can be achieved by periodically broadcasting the HELLO packets.

For simplicity, we describe our routing algorithm with only 1-hop neighbors. Using the distance function D_p, the source node or a relay node S calculates its distance to the destination, which is designated as D_{sd}. Then, node S calculates each neighbor's "distance" to the destination by using D_p. Assume that $Min(D_{nd})$ is the minimal distance and the right neighbor is $\overline{N}$. If $Min(D_{nd})$ is less than D_{sd}, the sender will forward the data packet to the neighbor $\overline{N}$. Otherwise, the node S is a dead end, and the greedy routing will stop there. We next present a novel landmark-guided routing to address the dead end problem.

3.7 The Dead End Problem

In geographic routing, *voids* cause dead ends. Essentially, voids make the physical distance fail to reflect the hop distance. Similarly, the Hop ID distance metric also deviates from the hop distance to some extent, so dead ends still exist. The number and selection of landmarks determines the Hop ID coordinates, which greatly affect the possibility of dead ends. For example, in Fig. 1 for destination node C, node A is a dead end. There is one kind of dead end where a relay node has the same Hop ID as that of the destination, or some nodes have the same Hop ID. We call this kind of dead end a *SHID dead end*. Unlike geographic routing, the dead end problem is significantly alleviated because the distance metric is closer to the hop distance; thus, it better resembles the topology of the network. The simulation results in Section 4 demonstrate this. Still, a small number of dead ends do exist and the problem needs to be addressed. However, the face routing technique cannot be applied because our Hop ID coordinates have much higher dimensions.

We next present a novel landmark-guided routing to solve this problem. The key observation is that the landmark nodes themselves are good guides for routing around dead ends. When the destination and a landmark are close to each other, moving a packet toward the landmark will probably make the packet closer to the destination. Thus, when a dead end is encountered, the landmark-guided algorithm attempts to send the packet toward the closest landmark to the destination. The details of the algorithm are described as follows.

When a node E finds it is a common dead end (not a SHID dead end), it records its distance to the destination (denoted as D_e) in the data packet. Then, the node finds the nearest landmark node to the destination, which will become the *guide*. The packet will be forwarded to the guide hop by hop. The routing then enters a *detour* mode from the original *greedy* mode. For example, node A is a dead end for destination node C in Fig. 1. Then, A enters the detour mode and sends the packet to L_3, which is the nearest landmark to C. When the packet reaches node G, G will notice that it is closer than node A to C, and node G can leave the detour mode and switch back to greedy mode again. This detour process continues until one of the following conditions is satisfied:

1. The current node is closer to the destination than the dead end node E. Thus, the routing returns to the greedy mode from the detour mode.
2. The packet in the detour mode has been forwarded more than t hops or it reaches the landmark. In this case, the landmark-guided algorithm fails and we use expanding ring flood, introduced later, to guarantee routing success.

We use this detour algorithm only for routing out of a dead end, and then we can resume the greedy procedure if it succeeds. We use a parameter t to control the depth of the detour algorithm. Intuitively, t should be a small number because sending the packet to a landmark does not help when this landmark is not in the same direction as the destination. Large t-values cannot help much and introduce additional overhead. The simulation shows that when a t-value of 5 is chosen, we can route out most dead ends without relying on the flooding for a variety of network sizes (e.g., from 800 to 51,200 nodes).

This detour algorithm cannot completely resolve all dead end problems, but it effectively mitigates the dead end problems without making the routing paths much longer, as shown in the simulation results of Section 4. Note that this will not cause landmark nodes to become bottlenecks, because data will only be passed to a landmark when both the dead end and destination are very close to the same landmark. The simulation further validates this observation.

Expanding ring flooding is a very simple but costly algorithm in routing search. It floods to search some node and increase the flooding range (e.g., by increasing the TTL) until the destination is reached. Our routing algorithm uses this algorithm to solve the remaining dead ends, including the SHID dead ends. Except for SHID dead ends, we only use this algorithm to find a closer node and, thus, the greedy algorithm can move on. As for a SHID dead end, the real destination is usually not very far from this dead end,

so no large range flooding of the expanding ring is needed. As a result, the overhead is low.

Notably, the overall routing algorithm is loop-free. Once the greedy routing algorithm fails at a dead end, the distance from the dead end to the destination is stored in the packet header. The landmark-guided algorithm and expanding ring flooding switch back to the greedy algorithm only if the current node is closer to the destination than the stuck dead end. Overall, the routing is a greedy process and, thus, no loop exists in the routing.

3.8 Summary and Analysis

Our routing algorithm relies on the construction and maintenance of the Hop ID system. It has the following three key steps:

1. A voluntary node floods to the entire network and builds a shortest path tree rooted at this node.
2. Landmark nodes are selected randomly using the landmark selection algorithm. After this procedure, each node can obtain its Hop ID.
3. Each node adjusts its Hop ID periodically and broadcasts its new Hop ID by a HELLO message.

The routing algorithm is a common greedy procedure, which is similar to geographic forwarding. To deal with the dead end problems, we design a landmark-guided detour algorithm and apply it with the expanding ring algorithm to route out of dead ends.

Now, we analyze the overhead in construction and maintenance for the Hop ID system. Assume we have totally N nodes in the network and m landmarks. To construct the Hop ID system, there is $O(m)$ flooding to the entire network, i.e., $O(m \cdot N)$ control packets. As shown in Section 4, m is usually a small number (less than 40), even for a reasonably sparse and large ad hoc network of 3,200 nodes. Furthermore, it will change little as N increases.

To maintain the Hop ID system, each node broadcasts the HELLO message periodically, so the overhead is $O(N)$ control packets in a period, but the overhead of bandwidth consumption (in bytes/second) is $O(m \cdot N)$ as each HELLO packet contains a node Hop ID, an m-dimensional vector. Thus compared with the geographic routing, we consume more bandwidth for sending larger messages. Another overhead is the packet header. Every data packet must include the Hop ID of the destination, which costs $O(m)$ (usually m bytes of Hop ID) bytes per packet. As for Hop ID lookup overhead, it is $O(\sqrt{N} \cdot N)$, since each node will send a packet to the location server and the average hop distance between a node and a landmark is $O(\sqrt{N})$. Another overhead is the flooding overhead of the expanding ring when the landmark-guided routing fails. It is hard to give a theoretical bound for this overhead, while our simulation shows that the overhead is very low in practice (see Section 4.3.2). Table 1 shows the comparison of overhead between our Hop ID system and GWL [15].

TABLE 1
Overhead of Hop ID versus GWL

	Hop ID	GWL
Initialization (packets)	O(m·N)	$O(\sqrt{N} \cdot N)$
HELLO Packets	O(N)	O(N)
Packet Header (bytes/packet)	O(m)	O(1)
Location Server	$O(\sqrt{N} \cdot N)$	$O(\sqrt{N} \cdot N)$

4 Performance Evaluation

In this section, we evaluate the Hop ID system through simulations. The routing algorithm has three stages: pure greedy routing, the detour algorithm for most dead ends, and expanding ring algorithm to guarantee the routing success. For convenience, we call the pure greedy routing algorithm HIR-G and call the HIR extended with detour algorithm HIR-D. The HIR-D with the expanding ring algorithm is called HIR-E. Using expanding ring, HIR-E can always guarantee routing success if the source and destination are connected. That is the reason why most graphs omit it. For comparison, we also implemented the geographic routing without location information [15], which is called GWL. In addition, we implemented the pure greedy geographic forwarding (GFR) and GOAFR+ [8] protocol with the real geographic coordinates.

4.1 Evaluation Methodology

4.1.1 Experiment Design

Unless otherwise indicated, most scenarios include N (N varies from 200 to 51,200) nodes distributed randomly in a 2D $C \times C$ square area. The communication range of each node is 1. We use λ to denote the density of the network, where $\lambda = \pi \times N/(C \times C)$. The definition of the density reflects how many nodes there are per unit disk in the 2D space, i.e., the average number neighbors of a node. In the 3D space, the density is similarly defined as $\lambda = \pi \times N/C^3$. For each scenario, 200 random nodes are chosen as both source and destination nodes. Thus, 38,900 routing paths populate one scenario. We repeated this process 20 times to obtain an average.

4.1.2 Evaluation Metrics

In our evaluation, we consider the following metrics:

- **Routing success rate**: the fraction of packets that can be successfully delivered to the reachable destination. This metric is trivial for HIR-E since it can always get a 100 percent success rate by flooding. However, the metric of HIR-D tells us how well the greedy (with detour extension) algorithm works and, thus, how often the Hop ID routing has to resort to expanding ring flooding.
- **Flooding range**: the number of hops HIR-E uses in expanding ring flooding to resume the greedy routing. Other than the overhead of flooding in packets, this metric provides a clear picture on the size of the flooding range.
- **(Shortest) path stretch**: the ratio of the real routing path length to the shortest path length between the source and destination. Routing path length may not be the best metric to depict the data transfer overhead, and other metrics are proposed, such as ETX [18]. But, as in other geographic routing

protocol papers, we only use path length, and plan in future work to explore other metrics as the greedy metric.

4.1.3 Simulation Model

First, we implement our algorithm in ns2 [23]. Unfortunately, ns2 itself is not scalable to a large system, e.g., a system with 3,200 nodes. To evaluate the performance of large networks and compare it with the previous work [15], we also implemented a packet level simulator that can scale to tens of thousands of nodes. In this simulator, radios have a precise (circular) radio range 1, and nodes can send packets only to nodes within this range. There is no bandwidth limit in the simulator. We do not simulate collisions in the simulator, though packets may be dropped with certain probability based on the configuration. This simple model enables us to abstract the impact of message loss and signal attenuation on routing performance and allows us focus on how well the routing algorithm performs. We compared the results of our ns2 simulator and scalable but simple simulator using small networks (800 nodes) and found that they did not have a significant difference. Thus, we mainly present the simulation results from the scalable simulator.

In addition, to evaluate how our algorithm works in a more realistic environment, we also added simulations that include:

1. Density—Network density varies from π to 4π, which covers the worst critical densities for geographic routing.
2. Scalability—We simulate large ad hoc networks with sizes of up to 51,200 nodes.
3. Mobility—For mobility, we use the modified random way point model [1] suggested in [17].
4. Losses—Nodes drop incoming packets with a given probability. Since we do not model a specific MAC layer, radio technology, or data-traffic pattern, we resort to a uniform loss model. While this may not be a realistic loss model, it does provide some insight into the robustness of the algorithm in the presence of loss.
5. Obstacles—We model obstacles by using straight walls that are parallel to the x or y-axis. Nodes cannot communicate with each other if the line connecting them intersects with a wall.
6. 3D space—Hop ID is a multidimensional virtual coordinate with no assumptions on the dimension of the network. As for geographic routing, more work need to be done for face routing to be applied in a 3D space.
7. Irregular shapes and voids—We create networks with voids inside the network that do not contain any nodes. We further simulate networks of various shapes, including concave shapes.

4.2 Ns2 versus Simple Scalable Simulation Tool

Ns2 [23] is a widely used simulation tool in wireless network research. It can simulate most details of the IEEE 802.11 MAC layer [19] and some simple but typical physical models. Thus, the simulation result based on ns2 is much more realistic than that of packet level simulators and, hence, is more widely accepted. However, ns2 is not scalable and runs very slow if the network is large (e.g., 3,200 nodes). Thus, we resort to our own simple but scalable simulation tool, which ignores the details of the MAC layer and physical layer. Here, we want to explore the gap between these two simulation tools and show that it is viable to adopt the simple simulation tool.

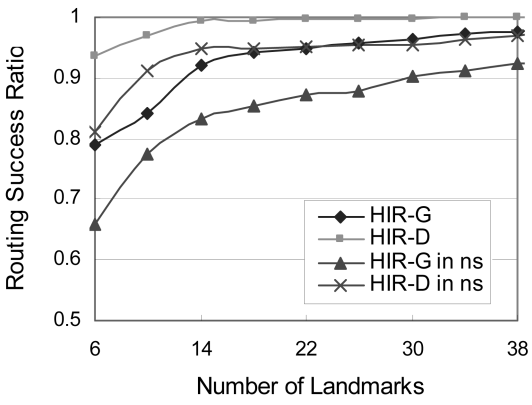

Fig. 5. Ns2 versus scalable simulator on success rate.

To compare the two kinds of simulation tools, we select some networks of 800 nodes. The density is 3π. Fig. 5 shows the routing success rate of both HIR-G and HIR-D obtained by the two simulation tools. HIR-G and HIR-D in ns2 achieves a little bit lower success rate than those in the simple simulation tool. Actually, the simple simulation tool gives the perfect running environment for the routing protocols since there is no loss and delay of packet transmission. In ns2, packets may get collision and, thus, link loss is common. More specifically, some control messages are lost in the building process of the Hop ID system and the power of the Hop ID system is affected. For example, if a landmark floods a LANDMARK packet to announce its landmark position and, unfortunately, this packet gets collided, the flooding fails and no other nodes receive the LANDMARK packet. Thus, this landmark does not in fact help routing. IEEE 802.11 MAC does not provide reliable broadcast and the back-off algorithm does not adjust the contention window when broadcast packets are collided [19]. One trick we use in our implementation is to set network adapters in a promiscuous node and change broadcast to unicast, i.e., a broadcast message is modified to send to a particular neighbor, while all the neighbor nodes can sniff and take the packet as a broadcast.

Fig. 6 shows the length of routing paths in terms of span of the shortest path. Again, the results from the ns2 simulator are a little bit worse. When there are enough redundant landmarks, these two simulators have the same performance. Fig. 7 shows the flooding range of HIR-E. Since the expanding ring algorithm is to find the next hop to resume the greedy routing, in Fig. 7, we can tell that a small range of flooding is usually enough.

4.3 Landmark Selection

4.3.1 Landmark Selection Preference

As we discussed in Section 3.3.2, a peripheral landmark is potentially more efficient than random landmark selection. In this experiment, we select 3,200-node scenarios and the density of networks is 3π. It is clear that peripheral

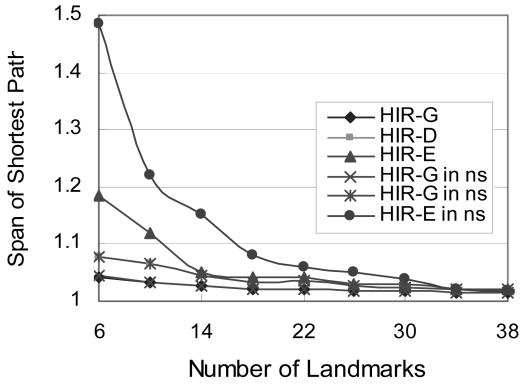

Fig. 6. ns2 versus scalable simulator on span of path.

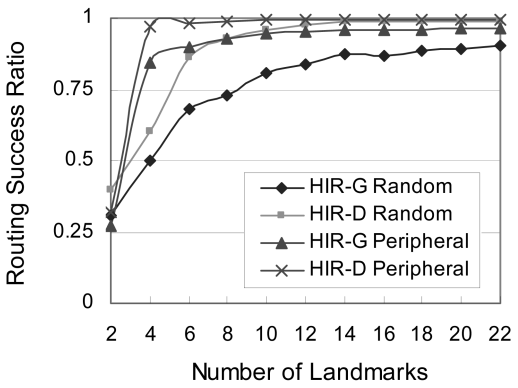

Fig. 8. Random landmark versus peripheral landmark.

landmark selection significantly outperforms random landmark selection (see Fig. 8). With only about 10 landmarks, HIR-D achieves a nearly 100 percent routing success rate if a peripheral landmark selection algorithm is adopted. However, in mobile scenarios, nodes are moving dynamically, making the maintenance of landmarks on the perimeter of the network more difficult. Furthermore, robustness may also be affected since nodes on the perimeter of the network tend to get disconnected. Therefore, we foresee that the peripheral landmarks selection algorithm will be used in less dynamic networks as sensor networks. In the following simulations, we only use the random landmark selection algorithm.

4.3.2 Landmark Sensitivity

The number of landmark nodes is a very important parameter for the Hop ID system. The theorem described in Section 3.2 shows that, with a constant number of landmarks, we can obtain a precise hop distance measurement regardless of the network size, but this constant is related to network density. Thus, the sparser the network, the more landmarks needed. In the following simulation, we find that the landmark number is actually not very sensitive to the density and the size of networks, i.e., after the number of landmarks exceed certain value (like 30), the increase of the number of landmarks yields little improvement for routing performance.

Fig. 9 shows how the landmark number affects the routing success rate. We use a 3,200-node network and the density varies from 2π to 3π. In a moderate dense network ($\lambda = 3\pi$), HIR-D has higher than 98 percent routing success rate with only about 20 landmarks. While in a quite sparse network ($\lambda = 2\pi$), HIR-D requires 30 or more landmarks to ensure 95 percent routing success rate. In the following sections, we fix the number of landmarks as 30 in all the simulations. We find that such a small number of landmarks are robust and sufficient for a large range of network settings.

Fig. 10 shows the overhead of flooding in networks of different densities. Even in the sparse network, the flooding range is very small (less than seven hops if landmark number is 30), compared with the network diameter above 60 hops. Actually, we only use the expanding ring algorithm to find one closer hop, which can usually be satisfied in a small region, so the expanding ring algorithm needs little overhead to find the next greedy hop.

4.4 Density

In this section, we study the performance of our algorithm under various network densities. As shown in [8], the critical density for routing is around 4.5 nodes per unit disk ($\lambda = 4.5$). In our simulation, the density of the network varies from π to 4π. This density range does not cover the extremely dense network ($\lambda = 5\pi$), as is used in [15], simply because all the simulated protocols route with an almost 100 percent route success rate and path stretch of nearly 1.0. For the partial-connected network, we only take the largest connected subnetwork into account and omit those scattered nodes. We choose a 20×20 square

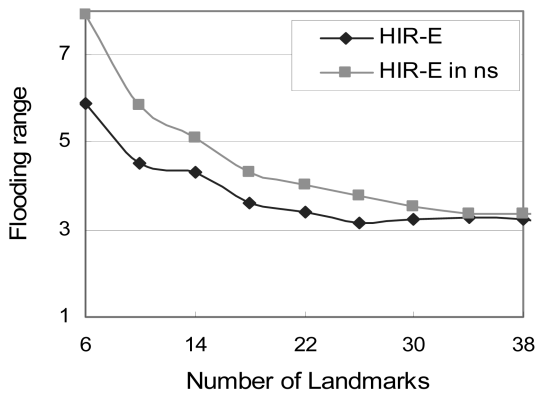

Fig. 7. ns2 versus scalable simulator on flooding range.

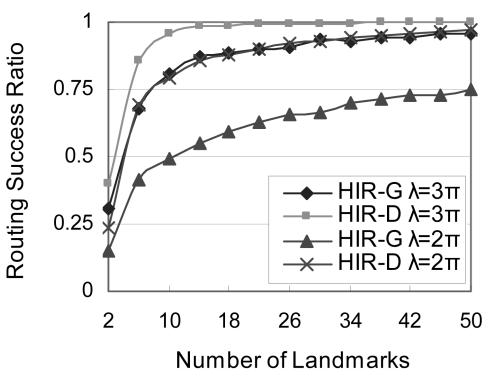

Fig. 9. Landmark number versus routing success rate.

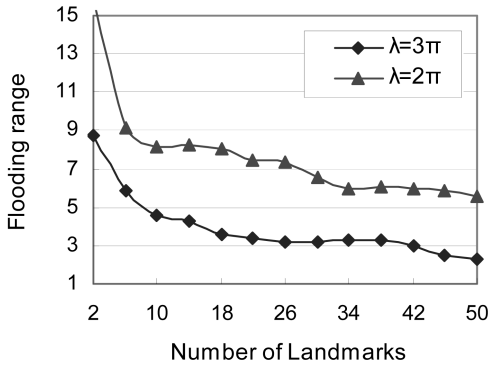

Fig. 10. Landmark number versus flooding range.

area and nodes uniformly distributed in the area. The number of nodes N is determined by the network density, e.g., 800 nodes when λ is 2π.

Fig. 11 shows the success rate of GFR, GWL, HIR-G, and HIR-D as a function of network density. GOAFR+ is not included in this figure because, by using the face routing algorithm, GOAFR+ can provide guaranteed routing. The same goes for the HIR-E, as the expanding ring algorithm can always find the next hop or the destination. It seems to be strange that the success rates of all algorithms decrease as network density increases when the network density is smaller than some critical value (about 5.0). The reason is that the network is split into many small disconnected subnetworks, but we only take into account the node pairs in the same connected subnetwork. When the network is extremely sparse and each subnetwork contains only tens of nodes, routing is much easier. Fig. 11 shows that GFR performs very poorly in some critical sparse networks because the geographic distance severely deviates from the hop distance and the GFR routing encounters a large number of dead ends. GWL outperforms GFR when the density is very low, which shows that the virtual coordinates can better capture the topology. HIR-D performs the best, which is more than 97 percent in the most critical network density. This shows that landmark nodes are a good guide for dead ends and the detour algorithm can effectively help to resolve the dead ends. Note that, the higher the route success rate of HIR-D reaches, the less flooding overhead is introduced by HIR-E.

Fig. 12 shows the path stretch of GFR, GWL, GOAFR+, HIR-G, and HIR-D with different network densities. The

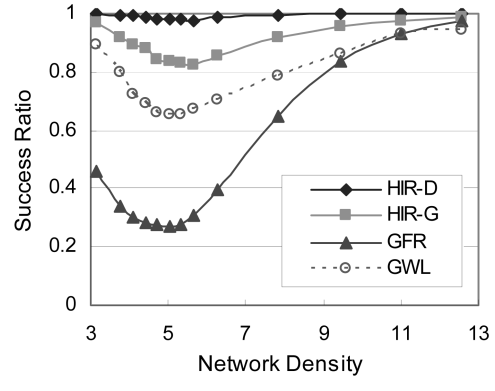

Fig. 11. Routing success rate as a function of network density.

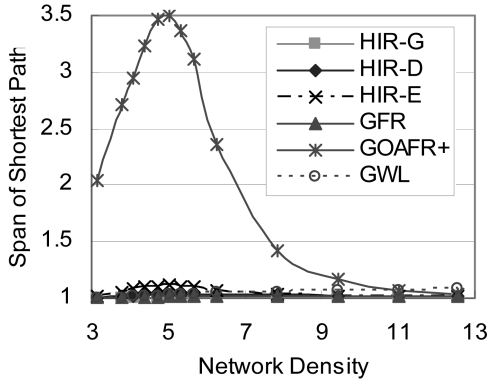

Fig. 12. Path stretch as a function of network density.

path stretch of GFR, GWL, HIR-G, and HIR-D are always very close to 1.0 (the four curves overlap in Fig. 12), irrespective of how sparse the network is. GOAFR+ performs the worst; the path stretch is as high as about 3.5 for the critical density. HIR-E is not included because HIR-E has nearly the same path stretch as HIR-D. Since the routing efficiency of all the simulated protocols except GOAFP+ are always close to shortest path, we omit the graphs on routing path stretch in the following tests.

Fig. 12 also shows that the path length of GOAFR+ is much higher in very sparse networks. Because geographic forwarding has a low success rate, GOAFR+ mostly has to resort to face routing. As a result, the routing path stretch is even more than three in the worst case. It is worth mentioning that the performance of GFR is much better than that in [8] because we use 2-hop neighbor information when executing the greedy algorithm.

4.5 Scalability

In this section, we investigate the scalability of our routing algorithm.

As discussed in Section 3.2, Theorem 1 shows that the number of landmarks does not increase as much as the number of nodes increases because it goes asymptotically to a constant in the square-shaped networks. We next use simulation to verify this observation. In simulations, we adopt a moderate density 2D network, where $\lambda = 3\pi$. The network size varies from 200 to 51,200 nodes and the number of landmarks increases from 5 to 50.

In Fig. 13, we use a 3D graph to show the relation between routing success rate, number of nodes, and number of landmark nodes. First, the performance of both HIR-G and HIR-D degrades as the network size increases. Intuitively, this is because the average routing length increases with the growth of the network size. The probability of encountering nodes with imprecise distance to the destination increases, which usually causes dead ends. Furthermore, the local information becomes less accurate for the greedy algorithm. Thus, with the help of landmarks, HIR-D can resolve more dead ends and performs better than HIR-G. For example, when there are only 30 landmark nodes in a 51,200-node network, the routing success rate is still about 97 percent. Second, Fig. 13 shows that only a small number of landmarks are sufficient for a large ad hoc network. The surface of HIR-D in Fig. 13 is quite flat when there are 20 or more landmark nodes. This shows that HIR-D is not sensitive to the number of

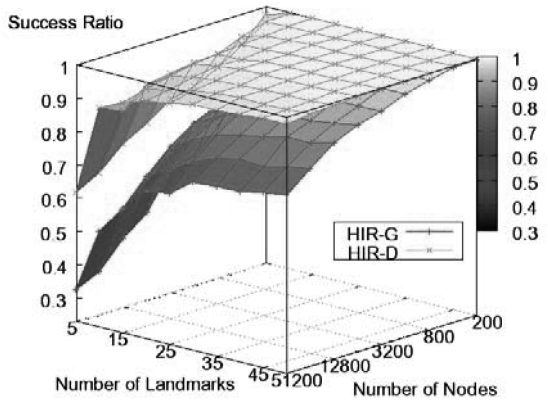

Fig. 13. Routing success rate as a function of number of nodes and landmarks.

landmark nodes. By simply choosing a proper landmark number (such as 30), HIR-D can adapt to various scenarios. Fig. 14 shows the efficiency of the routing paths of HIR-G and HIR-D. Clearly, given a high enough number of landmark nodes (e.g., 30), the routing paths chosen by HIR-G and HIR-D are close to optimal as the shortest path, even in networks that are quite large.

Fig. 15 compares the success rate of GFL, GWL, HIR-G, and HIR-D with two kinds of network densities, respectively, as a function of network size. GFL and GWL also have similar characteristics, i.e., they perform worse as the network size increases. Clearly, HIR-D outperforms other schemes; GFR performs the worst and degrades rapidly as the network size increases.

4.6 Mobility

In this section, we model mobility by using the modified random way point model [17]. Each node picks a destination at random within the square grid and moves toward the destination with a speed uniformly distributed in the range [0.004, 0.076]. The average speed is 0.04. If one unit in our simulation is equivalent to 250 m (the typical 802.11 communication range in simulations [6]), the average speed is equivalent to 10 m/s. When a node searches for its destination, the node remains stationary for a time interval

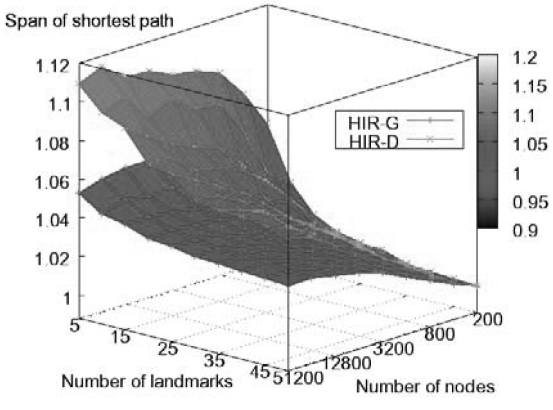

Fig. 14. Span of shortest path as a function of number of nodes and landmarks.

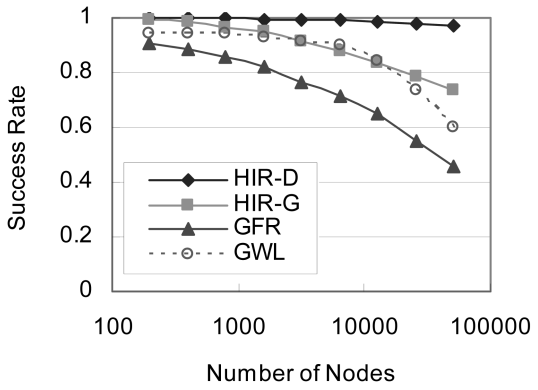

Fig. 15. Routing success rate as a function of network size.

called *pause time*. After staying for the *pause time*, the node selects another destination, and the process repeats.

In the mobile scenario, the Hop ID of a node may not be accurate and, thus, degrades the performance of the HIR algorithm. The Hop ID adjustment algorithm adjusts the Hop ID of each node locally and eventually adjusts the Hop ID of all nodes globally. The HELLO packet interval determines the adjustment frequency and, in our setting, is 1 second on average, which can detect the local topology change in time. The landmark coordinator sends out one PING message to each landmark and waits for the PONG messages. Once the coordinator or a landmark is found to be out of reach for 5 seconds, the corresponding coordinator or landmark reelection will be launched. There are 3,200 nodes in the square and the network density is 3π or 5π. Fig. 16 shows that even high mobility is not harmful to our Hop ID system. The imprecise Hop ID system works quite well as the success rate of HIR-D is above 92 percent in the worst case. As pause time increases, the mobility of the network becomes lower and lower and, as a result, HIR-D obtains close to a 100 percent success rate.

Now, we examine the control overhead introduced by Hop ID algorithms. In the mobile scenario, nodes may disconnect from or reconnect to the network, which requires the management of landmark nodes and even causes the reelection of landmark nodes; Hop ID coordinates need to be maintained by HELLO messages adapting to the change of the topology and more expanding ring flooding may be conducted because of the imprecise Hop ID coordinates. Table 2 lists the overhead of the

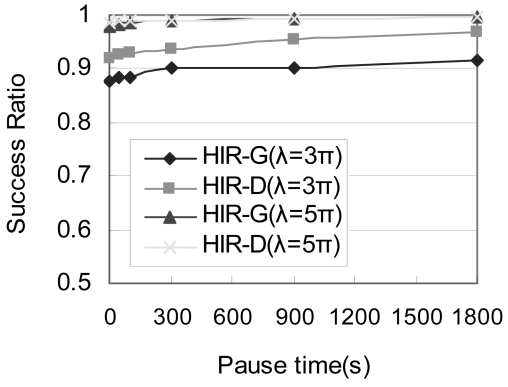

Fig. 16. Routing success rate as a function of node mobility.

TABLE 2
Overhead of Hop ID

Pause (second)	HELLO message (pkt/second/node)	Landmark related message (pkt/second/node)	Expanding ring flooding (pkt/route)
0	0.999	0.65	1.38
100	1.001	0.59	1.22
900	0.999	0.31	0.08

Hop ID algorithm in the mobile scenario with a density of 3π. For HELLO messages, the overhead is determined by the parameter of the hello interval, explained in the paragraph above. Landmark-related messages include the CENTER message, CANDIDATE message, LANDMARK message, and PING/PONG message. In the scenarios with highest mobility (i.e., pause time of zero second), on average, there is one reelection of landmarks every 80 seconds because some moving landmarks may be disconnected from the network and, then, dynamic landmark selection is executed. All the landmark management actions introduce 0.65 packet/s on average for each node. The landmark coordinator is the hotspot, which sends about 30 packets/s to query the states of other landmarks and to reelect new landmarks. The expanding ring overhead of HIR-E depends on the number of route requests during the simulation, so we use the number of packets in expanding ring flooding divided by the number of route requests to reflect the overhead of HIR-E. In the worst simulated case, there are 1.38 control packets per route introduced by the expanding ring. It is worth mentioning that, since the packet level simulator does not consider packet collision and, hence, introduces no loss, the overhead listed in Table 2 actually underestimates the overhead to some extent. In general, loss of packets usually triggers some extra retransmission overhead, which may increase the overhead with a certain factor. However, an extremely large loss rate may challenge the robustness of the protocols and it would be one our future works to study this problem.

Compared to flooding-based routing protocols [1], [2], we believe that the control overhead of Hop ID is very low and will not impact the data delivery. Supposing that there are m (new or resent because of mobility) routing requests every second, Hop ID introduces $1.65N + 1.38\ m(N = 3{,}200)$ control messages in the case of pause time 0. For AODV [2], there will be $(1 + m)N$ control messages if the average interval of HELLO messages in AODV is also 1 second. Flooding-based routing is not scalable as m may be very large when the network is very large. When m is larger than 0.65, the Hop ID's overhead is smaller than that of AODV. On the other hand, Hop ID uses 65 percent more overhead than AODV in the worst case $(m = 0)$.

4.7 Simulations of Other Practical Issues

4.7.1 Loss and Collisions

In this section, we study the robustness of our algorithm in the presence of losses. We model losses by randomly dropping control packets with a probability p. To factor out the routing failures due to data packet losses, we do not drop any data packets. While this is arguably not a very realistic loss model, it allows us to study the robustness of our algorithm with incomplete information. We simulated a dense network with λ be 5π and network size as 3,200 nodes.

Fig. 17 shows the success rate of greedy routing when the loss rate p increases from zero to 30 percent. As expected, the success rate drops as the loss rate increases. However, this drop is not severe. For a 30 percent loss rate, the average success rate of greedy routing is still greater than 98 percent. The success rate is greater than the probability of hop-by-hop packet delivery because we ignore losses on the data path. These results suggest that our algorithm is robust in the presence of packet losses. Intuitively, this is because, even with imprecise Hop ID measurement, the greedy algorithm can still work well. In contrast, with the same setup, the performance in [15] is much worse.

4.7.2 Obstacles

In this section, we examine the routing performance in the presence of obstacles. We model the obstacles as walls with lengths of up to 6.25 units. For comparison, the radio range of a node is assumed to be 1 unit and a node only knows its two-hop neighborhoods. Thus, for large obstacles, it is not always possible for nodes to bypass using only the greedy routing. However, it is interesting to observe that the Hop ID distance is mainly determined by the topology of the network and obstacles in network will not directly affect the algorithm's performance. In fact, with the presence of more obstacles, more links in the original scenarios without obstacles are broken and the network becomes sparser. In other words, it reduces the average neighbor number and thus brings a minimum impact to the performance of our algorithm.

Fig. 18 plots the success rate for our greedy routing in a 3,200-node network with 20 obstacles of different obstacle lengths. The network density is 5π—very dense—and we wish to eliminate the effect of density. As expected, the success rate decreases as their length increases, but the performance of HIR-D is still very good. For example, when the obstacles have length 6.25, the success rate of HIR-D is

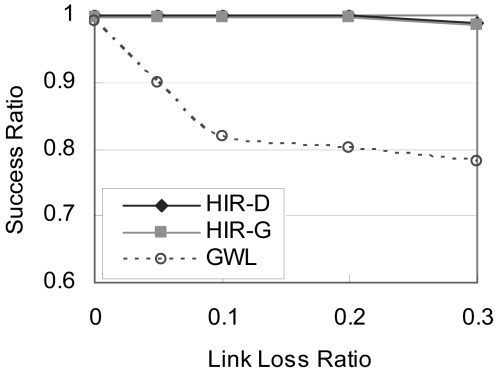

Fig. 17. Success rate as a function of loss rate of each link.

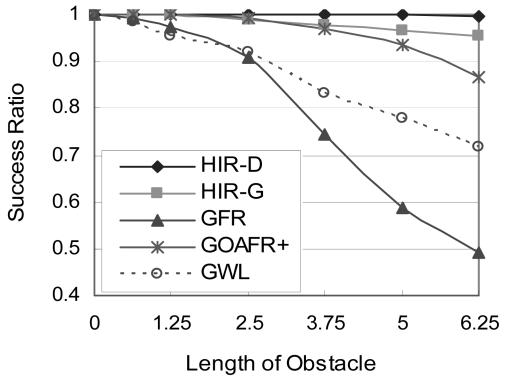

Fig. 18. Success rate as a function of obstacle length.

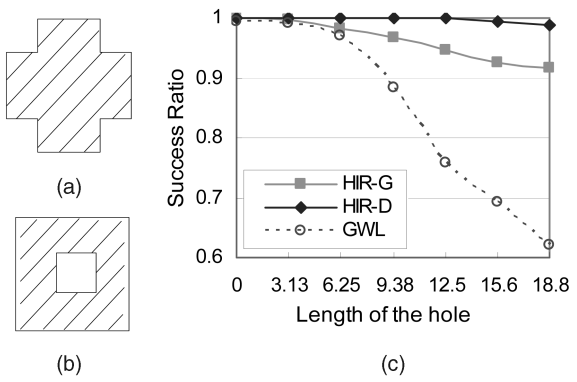

Fig. 19. Irregular shapes and success rate for shape (b) as a function of hole length.

still over 98 percent. On the other hand, geographic routing with real coordinates performs badly, which drops below 50 percent in critical scenarios. For GOAFR+, scenarios with obstacles make it hard for GOAFR+ to calculate the planar subgraph. Thus, the success rate of GOAFR+ drops from 1.0 to about 0.86 when the obstacles are as long as 6.25. As for GWL, it shows that the virtual coordinates are also severely affected by obstacles. The performance drops more than 20 percent when the length of obstacles is longer than 5.

4.7.3 Irregular Shapes

In this section, we explore networks where the nodes are distributed in areas of irregular shapes. Fig. 19 presents two kinds of irregular 2D shapes. The irregular shape in Fig. 19a has a concave perimeter and the shape in Fig. 19b has a large hole in the center of the square. In the simulation, 3,200 nodes are distributed in the shadow area of a 25×25 square grid, and the void space varies.

For the shape in Fig. 19a, the routing protocols exhibit similar performances and we omit the results. Fig. 19c shows the success rate of our algorithm for the irregular shape in Fig. 19b. The size of the hole in the center of the square varies from 0 to 18.75, i.e., from 0 to 9/16 in terms of the proportion of the area.

The results mirror those presented in Section 4.4. The irregular shape does not bring any essential change to the Hop ID system and, thus, has little effect. On the contrary, geographic routing using either real coordinates or virtual coordinates is significantly affected by the holes.

4.7.4 Three-Dimensional Space

So far, we have assumed that nodes lie in a 2D space. 3D space may be a more realistic scenario in many practical systems in that 2D space can be viewed as a special case. For example, buildings or mountains are typical 3D scenarios. In this section, we simulate a 3D network in a $10 \times 10 \times 10$ cube with a random number of nodes (determined by the density).

Geographic forwarding can directly be applied to a 3D space network or with minimum modification, as the greedy routing algorithm is not affected by the dimension. However, the face routing scheme used in most geographic routing algorithms such as GPSR [6] and GOAFR+ [8] does not work in a 3D space because the planar graph is the basic requirement. There has been no work on the geographic routing in 3D space. As for virtual coordinates, both our Hop ID system and GWL [15] make no assumption on the dimensions of the system, thus, they will not be affected by the change of dimensions. The results in Fig. 20 confirm that our algorithm works well in high-dimensional space. The key factor that affects the performance of our algorithm is the density of the network rather than the number of dimensions.

5 Conclusion and Future Work

In this paper, we have designed efficient routing schemes for mobile ad hoc networks of various densities, topologies, and obstacles. We propose a new virtual distance metric, called Hop ID distance, and design efficient algorithms for setting up the system, adapting to the node mobility quickly, and effectively routing out of dead ends. Extensive simulations show that the Hop ID scheme achieves excellent scalability and performance and can work in both sparse and dense networks; in particular, it is insensitive to obstacles and voids and can thus potentially be used in a wide variety of ad hoc environments.

Several issues need further investigation: 1) We have not explicitly taken into account the node addition and deletion. 2) Also worth examining is the impact from loss rate or delay when choosing the next hops for greedy routing.

Acknowledgments

The work conducted by Yao Zhao and Yan Chen was in part supported by a Motorola grant and a US Department of

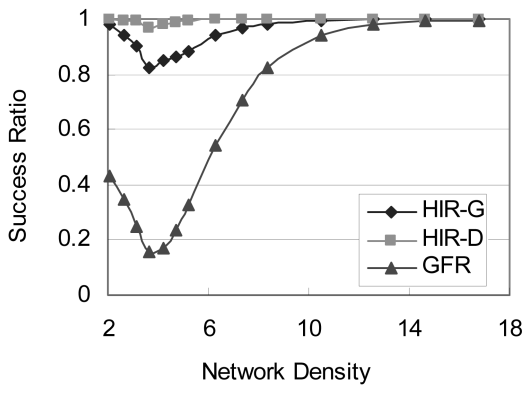

Fig. 20. Success rate as a density of density in 3D space.

Energy CAREER Award DE-FG02-05ER25692//A001. The work conducted by Bo Li was support in part by grants from RGC under contracts HKUST 6104/04E, HKUST6165/05E, and HKUST6164/06E and by grants from the National Science Foundation of China under contracts 60429202 and 60573115. The work conducted by Qian Zhang was in part supported by grant HKUST DAG05/06.EG05 and the National Basic Research Program of China (973 Program) under Grant No. 2006CB303000.

REFERENCES

[1] D.B. Johnson, D.A. Maltz, and J. Broch, "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks," *Ad Hoc Networking,* chapter 5, pp. 139-172, Addison-Wesley, 2001.

[2] C.E. Perkins and E.M. Royer, "Ad Hoc On-Demand Distance Vector Routing," *Proc. Second IEEE Workshop Mobile Computing Systems and Applications,* 1999.

[3] G. Pei, M. Gerla, and T. Chen, "Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks," *Proc. IEEE Int'l Conf. Comm. (ICC '00),* 2000.

[4] T. Clausen and P. Jacquet, *Optimized Link State Routing Protocol (OLSR),* IETF RFC 3026, Oct. 2003.

[5] X. Hong, K. Xu, and M. Gerla, "Scalable Routing Protocols for Mobile Ad Hoc Networks," *IEEE Network Magazine,* pp. 11-21, July-Aug. 2002.

[6] B. Karp and H. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proc. MobiCom,* Aug. 2000.

[7] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with Guaranteed Delivery in Ad-Hoc Wireless Networks," *ACM Wireless Networks,* Nov. 2001.

[8] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric Ad-Hoc Routing: Of Theory and Practice," *Principles of Distibuted Computing,* 2003.

[9] A. Helmy, S. Garg, P. Pamu, and N. Nahata, "Contact Based Architecture for Resource Discovery (CARD) in Large Scale MANets," *Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS '03),* Apr. 2003.

[10] T. Camp, J. Boleng, and L. Wilcox, "Location Information Services in Mobile Ad Hoc Networks," *Proc. IEEE Int'l Conf. Comm. (ICC '02),* pp. 3318-3324, 2002.

[11] J. Li, J. Jannotti, D. De Couto, D.R. Karger, and R. Morris, "A Scalable Location Service for Geographic Ad Hoc Routing," *Proc. MobiCom,* 2000.

[12] P. Bose et al., "A Survey on Position-Based Routing in Mobile Ad-Hoc Networks," *IEEE Network,* vol. 15, no. 6, 2001.

[13] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, "Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages," *Proc. MobiHoc,* June 2003.

[14] Y.C. Hu, H. Pucha, and S.M. Das, "Exploiting the Synergy between Peer-to-Peer and Mobile Ad Hoc Networks," *Proc. Ninth Workshop Hot Topics in Operating Systems (HotOS-IX),* May 2003.

[15] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic Routing without Location Information," *Proc. MobiCom,* 2003.

[16] J. Newsome and D. Song, "GEM: Graph Embedding for Routing and Data-Centric Storage in Sensor Networks without Geographic Information," *Proc. First ACM Conf. Embedded Networked Sensor Systems (SenSys '03),* pp. 76-88, 2003.

[17] J. Yoon, M. Liu, and B. Noble, "Random Waypoint Considered Harmful," *Proc. INFOCOM,* 2003.

[18] D. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A High-Throughput Path Metric for Multi-Hop Wireless Routing," *Proc. MobiCom,* 2003.

[19] *International Standard ISO/IEC 8802-11; ANSI/IEEE Std 802.11,* IEEE, 1999.

[20] J. Linn, *Generic Security Service Application Program Interface,* IETF RFC 1508, Sept. 1993.

[21] A.S. Tanenbaum and M. van Steen, *Distributed Systems: Principles and Paradigms.* Prentice-Hall, 2001.

[22] J. Kleinberg, A. Slivkins, and T. Wexler, "Triangulation and Embedding Using Small Sets of Beacons," *Proc. 45th Ann. IEEE Symp. Foundations of Computer Science (FOCS '04),* 2004.

[23] *NS2 Network Simulator,* http://www.isi.edu/nsnam/ns/, 2007.

[24] D. Ganesan, D. Estrin, and J. Heidemann, "DIMENSIONS: Why Do We Need a New Data Handling Architecture for Sensor Networks?" *Proc. ACM Workshop Hot Topics in Networks,* 2002.

[25] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin, "Data-Centric Storage in Sensornets," *ACM SIGCOMM Computer Comm. Rev.,* vol. 33, no. 1, pp. 137-142, 2003.

[26] X. Li, Y. Kim, R. Govindan, and W. Hong, "Multi-Dimensional Range Queries in Sensor Networks," *Proc. First Int'l Conf. Embedded Networked Sensor Systems,* 2003.

[27] D. Ganesan et al., "An Evaluation of Multi-Resolution Storage for Sensor Networks," *Proc. First Int'l Conf. Embedded Networked Sensor Systems,* 2003.

Yao Zhao is a PhD candidate in electrical engineering and computer science at Northwestern University. He received the BS and MS degrees in computer science at Tsinghua University in 2001 and 2004. He worked for the Oracle R&D Department in Beijing for half a year before his PhD study. His research interests include network measurement, monitoring and security, and wireless ad hoc and sensor networks.

Yan Chen is an assistant professor in the Department of Electrical Engineering and Computer Science at Northwestern University, Evanston, Illinois. He received the PhD degree in computer science at the University of California at Berkeley in 2003. His research interests include network measurement, monitoring, and security for both wired and wireless networks. He won the DOE Early CAREER award in 2005 and the Microsoft Trustworthy Computing Awards in 2004 and 2005.

Bo Li received the BEng (summa cum laude) and MEng degrees in computer science from Tsinghua University, Beijing, in 1987 and 1989, respectively, and the PhD degree in electrical and computer engineering from the University of Massachusetts at Amherst in 1993. Since 1996, he has been with the Department of Computer Science, Hong Kong University of Science and Technology. His current research interests are in adaptive video multicast, peer-to-peer streaming, resource management in mobile wireless systems, across layer design in multihop wireless networks, and content distribution and replication. He has published 90 journal papers and holds several patents in above areas. He received the Outstanding Young Investigator Award from the National Natural Science Foundation of China (NSFC) in 2004. He is currently a distinguished lecturer in the IEEE Communications Society.

Qian Zhang received the BS, MS, and PhD degrees from Wuhan University, China, in 1994, 1996, and 1999, respectively, all in computer science. Dr. Zhang joined the Hong Kong University of Science and Technology in September 2005 as an associate professor. Before that, she was at Microsoft Research, Asia, Beijing, China, since July 1999, where she was the research manager of the Wireless and Networking Group. Dr. Zhang has published about 150 refereed papers in international leading journals and key conferences in the areas of wireless/Internet multimedia networking, wireless communications and networking, and overlay networking. She is the inventor of about 30 pending patents. Her current research interests are in the areas of wireless communications, IP networking, multimedia, P2P overlay, and wireless security. Dr. Zhang has received the TR 100 (MIT Technology Review) world's top young innovator award. She also received the Best Asia Pacific (AP) Young Researcher Award elected by the IEEE Communication Society in 2004.