Efficient Prediction-Based Location Updating and Destination Searching Mechanisms for Geographic Routing in Mobile Ad Hoc Networks

REI-HENG CHENG AND CHIMING HUANG
Department of Computer Science and Information Engineering
YuanPei University
Hsinchu, 300 Taiwan
"Department of Information Management
Hsuan Chuang University
Hsinchu, 300 Taiwan
E-mail: yalo.huang@gmail.com

A geographic routing protocol needs to know the locations of destination node before a source can communicate with the destination. However, it is difficult to provide the exact location information for location services in a mobile ad hoc network. Nevertheless, the location of nodes can be predicted based on the nodes’ mobility information. We propose prediction-based mechanisms for location updating and destination searching. Many simulations under different motion environments are given. The simulation results show that the proposed mechanisms can significantly reduce the control overhead of location updating and improve the successful rate of searching the destination nodes in the meantime.

Keywords: geographic routing, prediction, mobile ad hoc networks, searching, location update

1. INTRODUCTION

The geographic routing protocol is a kind of location-based routing protocols [1-6] for mobile ad hoc networks. A geographic routing protocol uses the location information of destination nodes to determine the next hop nodes for forwarding packets. Since each node makes the forwarding decision locally and there is no fixed next hop, the geographic routing protocols are pretty useful for a network whose topology varies fast, such as mobile ad hoc networks (MANETs) or Vehicular Ad-hoc Networks (VANETs).

However, a geographic routing protocol has to know the locations of destination nodes before it starts to send packets. A central challenge in geographic routing protocols is the design of scalable distributed location services that track mobile node locations and reply to location queries for them [7]. Though frequent updating assures that a source node can always find the correct location information of the destination node, it also causes heavy control overhead. Nevertheless, since every node has its own moving pattern, it is not easy to find out a common updating scheme which is efficient for every node.

It is not possible to avoid the time gap between the measurement of a location and the time when this location information is actually used for a routing decision. This is because of the latency involved in the delivery of location information and also because the time interval between location-updates is generally longer than the inter-packet arrival.
times [4]. So, some studies proposed refining mechanisms to modify the estimated locations of destination nodes [8, 9].

Some researches used the mobility information to select the best forwarders for packets [4, 10-12]. In this paper, we propose prediction-based mechanisms which use mobility information in both the location service and the destination searching. These mechanisms include a prediction-based location updating scheme, and a prediction-based destination searching scheme with refining and back-tracing mechanisms. For the location service, every node examines its current location and the mobility information it has sent before, and then decides whether it should send out location-update packets or not. Meanwhile, packets are forwarded to predicted locations instead of the last found locations of destination nodes. These mechanisms are proved to be efficient in our simulations.

The rest of the paper is organized as follows. Section 2 discusses some related works. Section 3 describes the details of the proposed prediction-based mechanisms for improving the efficiencies of geographic routing protocols. Section 4 shows the results and discussions of simulations. And our conclusions are included in section 5.

2. RELATED WORKS

In DREAM [13], nodes used flooding to inform all other nodes in the network about their location information. The nodes with higher moving speed flooded their information more frequently. In order to avoid flooding problems, LAR [2] used the location and speed information to form a so-called request zone. When a source node $S$ wanted to find a destination node $D$, node $S$ would compute the request zone based on its current location and the information, including location and speed, of node $D$ it had collected. Only the nodes within the request zone would help forward request packets, and thus reduced the overhead for searching destination nodes. However, the mobility information of each node could only be piggybacked in route reply packets; it would not be easy to collect fresh enough information of nodes in a mobile ad hoc network.

EASE [8] did not use any location updating mechanism, and thus saved the relevant control overhead. However, while searching a destination node, EASE had to expand the searching range around the source node to find new enough location information of the destination node. When the time threshold was set too small, the query packet was not easy to find the information which met the requirements, and caused the control overhead close to the ones of flooding. On the contrary, if the time threshold was set too large, the old information might mislead the query packet to a wrong region. In [14, 15], the RREQ packets were flooded to build the routes between source and destination nodes. However, in [14], the destination node would proactively send location-update packets to the source node for keeping the route valid during communication.

To reduce the number of location-update packet and the required transmission distance, SLURP [16] divided a network into regions and assigned each node a home region. When the node moved from one region to another, it would send a location-update packet to the nodes in its home region (by broadcasting in its home region). Region size was an important parameter. A larger region might need fewer location-update packets, but would cause more overhead while broadcasting in the home regions to find destination nodes. On the contrary, a smaller region would result in frequent location-update, and smaller
successful rate of finding usable information in a home region. Meanwhile, when the nodes moved back and forth on the borders of regions, it would surely cause unnecessary location-updates. GLS [3] also divided a network into sub-regions, but each node had its hierarchical home regions (4 minimal sub-regions formed a higher order sub-region, and 4 higher order sub-regions formed a more advanced sub-region). When a node moved in the minimal sub-region, only the smallest sub-region’s location server had to update the related information. Whenever a node moved out of a smallest sub-region, the higher class of location server would need to be informed, therefore the cost of maintaining location information in servers was reduced. In VPDS [17], each node had a VHR (Virtual Home Region). Once a node left from previous reporting location more than a certain distance, it had to report a new location-update packet to its VHR. CALM [18] divided a circular environment into sectors and cylinders, and set the nearest node to the central point in each area as a local server. The authors discussed how the location-update and query mechanisms worked among these local servers.

Quorum [9] updated location information when a pre-specified number of links incident on a node had been established or broken since the last update. Seeker [19] determined the location updating frequencies of the boundary nodes based on the average speed of nodes in the specific areas.

In [4], the authors discussed the link broken problem caused by the mobility of nodes. And in [10, 11], a node selected a better forwarder based on the mobility information of its neighboring nodes, so that the successful rate of geographic routings could be improved. Also, a method was developed in [12] to trace nodes’ trajectory in order to predict nodes’ moving patterns. Basing on the prediction results, a node determined a better forwarder to deliver packets for itself, and thus a better packet delivery rate could be achieved. LPBR [20] used flooding to establish the paths between source and destination nodes, and collected nodes’ location and speed information in the meantime. As a transmission path was broken due to nodes’ mobility, the destination node calculated possible paths based on the collected information and sent the new path data back to the source node to reconstruct the transmission path without flooding again.

While searching a destination node, once the source node got the location information about the destination node, EASE [8] and Quorum [9] would forward a query packet to the location where the destination node had been before. As the query packet traveled towards its destination, it had to successively refine the target location for the destination node might have moved to a new location.

The literatures mentioned above might use the nodes mobility information to prevent links broken problems among neighboring nodes. However, there are no further discussions on how to apply mobility information for enhancing the location updating and destination searching performance in mobile ad hoc networks.

3. THE PREDICTION-BASED MECHANISMS

Basing on mobility (including location and speed) information of nodes, we propose two prediction-based mechanisms to improve the performance of geographic routing protocols. These mechanisms include a dynamic location updating mechanism which uses mobility information to determine the appropriate location updating time, and a location
prediction searching mechanism which bases on nodes’ mobility information to predict the destination nodes’ current locations, and forwards query packets to the predicted locations of destination nodes instead of the historical locations of destination nodes. A shortcut first refining and a back-tracing approaches are included in the searching mechanism to help reduce the length of searching paths and further improve the successful rate of searching destination nodes. The following will address these four mechanisms to detail. To describe the proposed mechanisms more precisely, some important notations are defined and listed in Table 1.

<table>
<thead>
<tr>
<th>Notations</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>Source node.</td>
</tr>
<tr>
<td>D</td>
<td>Destination node.</td>
</tr>
<tr>
<td>Q</td>
<td>Intermediate node.</td>
</tr>
<tr>
<td>LS</td>
<td>Location server.</td>
</tr>
<tr>
<td>L(x, t)</td>
<td>Real location of node x at time t.</td>
</tr>
<tr>
<td>V(x, t)</td>
<td>Speed of node x at time t.</td>
</tr>
<tr>
<td>P(x, t, y, s)</td>
<td>Predicted location of node x at time t based on the mobile information obtained from node y, which was sent by node x at time s.</td>
</tr>
</tbody>
</table>

3.1 Prediction-Based Location Updating Mechanism

In mobile ad hoc networks, each node has its own moving pattern. Current researches usually used fixed time/distance thresholds to determine whether a node had to send the location-update packet. However, a slowly or constantly moving node needs not to update its location information frequently for its location is predictable. That is, a node will not send out location-update packets if it is aware of that its current location can be predicted by the latest mobility information it has sent. By using one-hop broadcasting, a node can communicate with the neighboring nodes within one-hop distance easily, thus in our implementation, a node sends a location-update packet whenever it finds that the difference between its current location and the predicted location according to the latest mobility information it has sent is not smaller than one hop.

Take Fig. 1 for example, at time $t_0$, node A moves along the black line and sends a location-update packet at location $L(A, t_0)$. When node A arrives at location $L(A, t_1)$, it finds that the location predicted according to the latest mobility information it has sent out is $P(A, t_1, A, t_0)$, and the distance between $P(A, t_1, A, t_0)$ and $L(A, t_1)$ is greater than one hop, thus it sends out a new location-update packet piggybacked with current mobility information. Later, while node A moves from location $L(A, t_1)$ through $L(A, t_2)$ to $L(A, t_3)$, it knows the predicted locations (such as $P(A, t_2, A, t_1)$ and $P(A, t_3, A, t_1)$) are within one hop away from real locations (such as $L(A, t_2)$ and $L(A, t_3)$), so it just keeps quiet and sends out no location-update packet.

3.2 Prediction-Based Searching Mechanism

In geographic routing, a source node must know the location of the destination node
to send request packet and establish the connection between them. However, for the moving characteristic of nodes, a source node is hard to know the correct location of the destination node. However, each node can infer its speed according to the changes in location. As a source node gets the speed information of the destination node, it can predict the destination node’s current location, thus the inaccuracy problem caused by nodes mobility is eased.

When a source node $S$ sends out a location query for a destination node $D$, and this query packet arrives at the location server at time $t'$, the location server can predict the current location of destination node $D$ by using Eq. (1) according to the mobile information of node $D$ at time $t$ stored in location server.

$$P(D, t', LS, t) = L(D, t) + V(D, t) \times (t' - t)$$

Finally, the location server forwards the query packet to the predict location of destination node $D$.

Whenever a node finds that the predicted location (according to the latest mobile information it has sent) is not within one hop away from its current location, it will send a new location-update packet to the location server. So, in ideal case, according to the mobility information stored in the location server, the query packet will meet the destination node at the predicted location.

### 3.2.1 Shortcut first refining approach

Every time a node receives a location-update packet, it records the piggybacked information in its local memory. As a node receives a query packet, it searches the destination node of the query packet in its local memory. If newer valid information is found, instead of forwarding the query packet to the location server, the node directly forwards the query packet to the predicted location of the destination node. During the process of forwarding, if any newer information of the destination node is found, the target location of the query packet will be updated in order to hit the exact location of the destination node.

For example, in Fig. 2, the node $D$ at location $L(D, t_1)$ sends a location-update packet to the location server at time $t_1$. While the location-update packet is forwarding to the location server $LS$, a node $Q_1$ overhears the packet and keeps the piggybacked information. Later, at time $t_2$, the node $D$ moves to $L(D, t_2)$, finds the predicted location $P(D, t_2; Q_1, t_1)$ is not within one hop distance, and then the node $D$ sends a new location-update
packet to the \( LS \). When the node \( S \) wants to search for the node \( D \), it will send a query packet to the \( LS \). As an intermediate node \( Q_1 \) receives the query packet at time \( t_3 \), it finds itself owning the valid information of node \( D \), that is, \( L(D, t_1) \) and \( V(D, t_1) \). Assume that the predicted location of node \( D \) at time \( t_3 \) is \( P(D, t_3, Q_1, t_1) \), the node \( Q_1 \) will forward the query packet to location \( P(D, t_3, Q_1, t_1) \) instead of the \( LS \). Later in another intermediate node \( Q_2 \) on the way to the location \( P(D, t_3, Q_1, t_1) \), newer information of node \( D \) at time \( t_2 \) is found. The query packet is redirected to the new predicted location \( P(D, t_3, Q_2, t_2) \). By using this shortcut first refining approach, the query packet can be sent to the right location as soon as possible, and the possible traffic jam around the location server can be also avoided.

However, it is not easy to justify the effectiveness of the information stored in nodes. Since each node knows the locations where it had sent the location-update packets, it can calculate the distances between the successive locations (these distances are referred as updating distances hereafter). The effective distance \( y \) is got by calculating the first \( x \) percentile value of the updating distances. The information contained in a location-update packet includes (1) current time, (2) location, (3) speed, and (4) effective distance. To verify the effectiveness of information, a node checks if the distance between the predicted location and the location at which the information was sent is smaller than the effective distance. The positive answer means that the information should still be effective. In the simulations shown in section 4, we set \( x \) to 10.

### 3.2.2 Back-tracing approach

For the mobility of nodes, the location server is difficult to ensure the true location of nodes according to the information stored in it. Meanwhile, it seems to be unavoidable that the predicted location does not match up with the actual position sometimes. So we propose a back-tracing approach to increase the successful rate of finding the destination node. That is, if a query packet has arrived at the predicted location and cannot find the destination node, the query packet will be redirected to the location where the location-update packet was sent.
Take Fig. 3 for example, the query packet initiated by the source node $S$ at time $t_3$ is redirected to the predicted location $P(D, t_3, Q_1, t_1)$ around node $Q_1$ according to the information stored in the intermediate node $Q_1$. Before the query packet is sent out by $Q_1$, the location $L(D, t_1)$ where the information was sent is recorded in the query packet. Since the query packet cannot find the destination node $D$ around the predicted location $P(D, t_3, Q_1, t_1)$, it will be redirected to the location $L(D, t_1)$. To avoid missing the destination node, every node on the way to the location $L(D, t_1)$, before forwarding the query packet, will send out an one-hop searching packet to check if any neighboring node has newer information of node $D$. If any newer information is replied, the query packet will be redirected to the new predicted location. In Fig. 3, when the query packet is received by node $Q_3$, it finds newer information of node $D$ and redirects the query packet to the new predicted location of node $D$, that is, the location $L(D, t_3) = P(D, t_3, Q_3, t_2)$.

3.2.3 The overall algorithm of the searching mechanism

Each node $X$ which receives a query or one-hop searching packet will do following steps. Assume that the destination node of the query/one-hop searching packet is node $D$. The distributed algorithm for each node can be summarized as below.

**Step 1:** If a node $X$ receives a one-hop searching packet, it searches local memory for the information of node $D$. As newer information of node $D$ is found, node $X$ replies the found information to the sender of the one-hop searching packet. Go to step 11.

**Step 2:** If node $X$ is the location server, go to step 6.

**Step 3:** If node $X$ is node $D$, go to END.

**Step 4:** If the back-tracing bit in query packet is off, go to step 6.

**Step 5:** Broadcasting a one-hop searching packet to node $X$’s one-hop neighbors. Waiting one second for the possible replies from neighbors, and updating node $X$’s local memory according to the replied information.

**Step 6:** Searching local memory for the information of node $D$. If newer information is found, updating the predicted destination location of node $D$ in the query packet.
Step 7: Selecting the next hop node $Y$ based on the predicted destination location of node $D$.

Step 8: If node $Y$ is nearer the destination location than node $X$ or the distance between node $X$ and the target location is greater than the power range, the query packet is forwarded to the next hop $Y$. Go to step 11.

Step 9: If the back-tracing bit in query packet is on, the searching process is failed, go to END.

Step 10: Set the back-tracing bit in query packet to on, go to step 4.

Step 11: END.

4. PERFORMANCE EVALUATIONS AND DISCUSSIONS

The simulation program is written with Borland c++ 6.0 and run on a Pentium 4 CPU 3.0 GHz machine with 1G RAM. There are 500 nodes uniformly randomly deployed in a 1,000 meters by 1,000 meters rectangle region. The radio power range is set to 100 meters. In each simulation, 500 query packets are initiated in 200 seconds. The source and the destination nodes of these query packets are randomly selected from the 500 deployed nodes. Each result is the average value of 10 independent runs. The motion patterns of nodes are discussed in section 4.1. Three performance metrics are defined as follows.

1. Successful rate: The rate of the query packets which are successfully delivered to the destination nodes.
2. Query length: The average hop count of the paths through which query packets are successfully delivered from source nodes to destination nodes.
3. Control overhead: The total number of control packets. The control packets include location-update packets, and the one-hop searching packets used in the back-tracing approach. Each hop the control packet forwarded in the network is counted as one control overhead.

There are many researches discussed how to implement the location service. To simplify the comparison, we use a dedicated location server in our simulation. That is, every location-update packet is sent to a single location server.

It is not easy to compare with the methods without location updating mechanism. Intuitively, the approaches with location updating mechanism pay the maintenance costs for the locations of nodes, so the destination can be quickly found with less time and control overhead. On the contrary, the approaches without location updating mechanism save the maintenance costs, and take larger costs for searching the destinations; the cost may even be close to the cost of flooding. As these two kinds of approaches suit for different application scenarios, thus in our following simulations, we will not take the approaches without location updating mechanism into consideration. That is, we assume that there are two basic parts in a geographic routing protocol: location updating and nodes searching.

In some researches, a node will send its latest location to the location server whenever the node moves one-hop away, and this kind of mechanism is referred as one-hop location updating mechanism (referred as OHLU hereafter) in section 4.2. As the node
searching is concerned, whether to use the prediction approach is our major comparison factor. As we mentioned before, as the best of our knowledge, there are no approaches using similar prediction mechanisms as we do. Therefore, in section 4.2, we will discuss the effect of these individual mechanisms.

4.1 The Mobility Models

Two different mobility models are used in our simulations. The brief introductions of these two mobility models are given below.

4.1.1 Random way point

Each node \( X \) randomly picks a destination \( D \) and a speed \( S \) (between 1 and 20 m/sec), and then moves to \( D \) at \( S \) velocity. After reaching \( D \), node \( X \) re-selects a new random destination and speed, and then moves again.

4.1.2 Moving at random direction and speed

Each node starts to move at randomly selected direction and speed, and then it changes its moving direction and speed every 2 seconds. The changing quantities of direction and speed are normal distributed random numbers with zero mean. By varying the standard deviation of the normal distributed function, we get different moving patterns. A larger standard deviation value represents a more irregular moving pattern. The changing quantities for moving direction and speed are determined by the random numbers multiply 0.1 radius and 0.05 m/sec respectively. This kind of moving pattern is denoted as \( RM(x, y) \), where \( x \) and \( y \) are the standard deviation values for direction and speed changing quantities respectively. In the following simulations, we vary the values of \( x \) and \( y \) from 0.1, 0.5, 1, 2, to 4 respectively to evaluate the proposed mechanisms’ performance under different environments. Though there are totally 25 different moving patterns, however, the simulation results show that we can use same values for \( x \) and \( y \), that is (0.1, 0.1), (0.5, 0.5), (1, 1), and etc., to demonstrate the influence of moving patterns on performance metrics without loss of generality.

4.2 Performance Evaluation

To serve as a comparison baseline, we design a simple searching mechanism (SSM). In the SSM, a source node sends a query packet to the location server, and the location server forwards the query packet to the destination node according to the location information stored in the location server. For convenience of explanation, some abbreviations are defined first and listed in Table 2.

This paper proposes four mechanisms which include PLU, PRS, SFS and BTS. However, some previous researches also proposed similar mechanisms like PRS and SFS. For example, the EASE [8] used PRS and partial SFS as its searching mechanism, Quorum [9] and VPDS [17] used OHLU-like location updating mechanism, and Seeker [19] used an advanced ELU based on average speed of neighboring nodes. Some researches [16-18] focused on building and maintaining the hierarchical structure of location servers,
Table 2. List of abbreviations used in section 4.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLU</td>
<td>Prediction-based Location Updating mechanism.</td>
</tr>
<tr>
<td>OHLU</td>
<td>One-Hop Location Updating mechanism.</td>
</tr>
<tr>
<td>ELU-x</td>
<td>Periodical Location Updating mechanism; every nodes send location-update packets to the location server every $x$ seconds.</td>
</tr>
<tr>
<td>PRS</td>
<td>Prediction-based Searching mechanism.</td>
</tr>
<tr>
<td>SFS</td>
<td>Shortcut First refining Searching approach.</td>
</tr>
<tr>
<td>BTS</td>
<td>Back-Tracing Searching approach.</td>
</tr>
<tr>
<td>SSM</td>
<td>A Simple Searching Mechanism.</td>
</tr>
</tbody>
</table>

and used only SSM as the destination searching mechanism. Because of focusing on different targets, we cannot directly compare the performance of these strategies proposed in previous studies. Therefore, the following simulations just focus on revealing the effects of the proposed mechanisms. Basing on the results, we may correctly use these mechanisms with some other mechanisms proposed in previous studies in the future research.

To evaluate the performance of the proposed prediction-based location updating mechanism (PLU), we run SSM with PLU, OHLU, ELU-20 and ELU-5 respectively. However, while cooperating with PLU, SSM forwards query packets to the predicted location, instead of the historical location, of destination nodes. While using random way point mobility model, Fig. 4 (a) shows that the PLU produces 65% and 52% less control overhead than the OHLU and the ELU-20 do respectively. In PLU, a node sends the location-update packets only when it perceives that the other nodes may not predict its location correctly. Thus, though nodes send fewer location-update packets, there is always correct information to predict the location of each node in the location server. We can also see the dynamic characteristic of PLU in Fig. 4 (d). The simulation results show that the PLU can vary nodes’ location updating frequency according to the irregularity of nodes’ moving patterns. Although PLU sends less location-update packets than ELU-20 and OHLU do, Fig. 4 (e) depicts that the SSM uses PLU can still achieve the best successful rate.

The PRS proposed in this paper sends query packets to the predicted locations of the destination nodes, instead of the locations where the destination nodes were. Fig. 5 reveals that PRS can improve the successful rate of SSM with ELU-20 about 21% to 38% without increasing significant control overhead. However, higher successful rate also means that many far destination nodes can be found, and thus increase the average query length. Nevertheless, Fig. 5 (f) shows that the using of PRS can even decrease the query length, which means the PRS can send query packets to the destination node through shorter paths.

Finally, we combine the two prediction-based mechanisms, PRS and PLU, with SSM to see how the overall performance is improved. When nodes’ motion patterns tend to be regular, the proposed mechanisms can even improve the successful rate to 21% higher, and reduce 52% control overhead in the meantime. However, if nodes change their moving direction and speed frequently, the prediction-based mechanisms may need more control overhead to correct the prediction error and keep the high successful rate of searching destination nodes.
4.3 Results Analysis and Discussion

In this section, we will analyze how the mobility of a node may affect the correctness of the location prediction mechanism. To simplify the analysis, let’s consider the
case that a node moves at a constant speed but changes the moving direction randomly. That is, at any time, the node can change its moving direction to any angle. Thus, the probability of a node moving toward to any direction at any single time is all the same. In such a network, we have the Theorem 1 shown below for a simpler case.

**Theorem 1**  Assume that the original speed and location of a node \( A \) at time \( t \) are \( v(t) \) and \( p(t) \) respectively, and node \( A \) uniform-randomly selects an angle \( d \), where \( d \in [-\pi, \pi] \), as its new moving direction. If the moving speed of node \( A \) remains unchanged, then after node \( A \) moves \( h \) hops away, the probability of that node \( A \) can still be found based on \( v(t) \) and \( p(t) \) is denoted as \( \text{prob}(h) \) and can be calculated as shown below.

\[
\text{prob}(h) = \frac{\theta_h}{2\pi}, \text{ where } \theta_h = 2\times \cos^{-1}\left(\frac{h - \frac{1}{2}}{\sqrt{h^2 - h + 1}}\right)
\]  

(2)

**Proof:** Let’s assume that the moving direction \( v(t) \) of node \( A \) is shown as the half line \( OS \) in Fig. 6. Since the new moving direction of the node is uniform-randomly selected from \(-\pi \) to \( \pi \), the possible location where the node \( A \) may appear after \( h \) hops can be denoted as a circle (the black bold circle shown in Fig. 6). Once the distance between the predicted location of a node and the real ones is not greater than one hop (the red bold curve shown in Fig. 6), the node can be found with the predicted location and a simple one hop message exchanging. Thus the \( \text{prob}(h) \) can be denoted as follows.

\[
\text{prob}(h) = \frac{\theta_h}{2\pi}
\]  

(3)

In Fig. 6, we get the following equation based on trigonometry.

\[
\frac{(h - 1) \times r}{\sin(60^\circ - (\theta_h / 2))} = \frac{\sin(\theta_h / 2)}{\sin(\theta_h / 2)}
\]  

(4)

\[\Rightarrow \cos(\theta_h / 2) = \left(\frac{h - \frac{1}{2}}{\sqrt{h^2 - h + 1}}\right) \Rightarrow \theta_h = 2\times \cos^{-1}\left(\frac{h - \frac{1}{2}}{\sqrt{h^2 - h + 1}}\right)\]

When a node moves randomly as mentioned in Theorem 1, we find that the probability of a node can be correctly predicted after it moves one hop away is only 1/3. And the probability drops to 1/6 while a node moves 2 hops away. To keep the prediction mechanism working, a node may need to send location-update packets frequently. However, the motion pattern assumed in Theorem 1 is not fully random. Therefore, we may conclude that the location prediction model is not suited for a network in which all nodes move irregularly. In Fig. 4 (d), we observe that the control overhead of PLU getting higher as the deviation value getting larger. This phenomenon matches the conclusion derived from Theorem 1. However, how to determine the right time to use the location prediction mechanism is included in our future research.
Fig. 6. The possible location of a node which uniform randomly selects a direction and moves $h$ hops away from the original location $O$ (shown as the black bold circle). If the node appears on the red bold curve, then the node can be found based on the mobile information of the node at location $O$.

5. CONCLUSION

In this paper, we propose the prediction-based location updating mechanism and the prediction-based searching mechanism to improve the performance of geographic routing protocols. The proposed protocol includes a prediction-based location updating mechanism which can reduce significant control overhead without losing searching successful rate. On the other hand, the prediction-based searching mechanism has shown its ability to improve the searching successful rate and query length in our simulation results.

In the future, we will try to implement the proposed mechanisms with different location service approaches, such as home region. And we will also try to develop a more sophisticated motion prediction model which can further improve performance of the proposed mechanisms.

REFERENCES


20. N. Meghanathan, “A location prediction-based reactive routing protocol to minimize...
the number of route discoveries and hop count per path in mobile ad hoc networks,”

**Rei-Heng Cheng (鄭瑞恒)** received the M.S. and Ph.D. degrees in Computer Science and Information Engineering from National Chiao Tung University, Taiwan, in 1989 and 1995 respectively. In 2011, he joined the faculty of Yuanpei University, Hsinchu, Taiwan. He is currently a Professor at the Department of Computer Science and Information Engineering. His research interests include wireless and sensor networks, image processing, and character recognition.

**Chiming Huang (黃志明)** received the M.S. degree in Management Science and Ph.D. degree in Computer Science and Information Engineering from National Chiao Tung University, Taiwan, in 1988 and 1993, respectively. Since 2007, he has been on the faculty of Hsuan Chuang University, Hsinchu, Taiwan. He is currently an Associate Professor at the Department of Information Management. His research interests include wireless and sensor networks, artificial intelligence, algorithm, and distributed computing.