Border Landmark Selection and Applications in Self-Configurable Wireless Networks*

Chong Wang and Hongyi Wu
The Center for Advanced Computer Studies
University of Louisiana at Lafayette
Lafayette, LA 70504
Email:{cxw0623,wu}@cacs.louisiana.edu.

Abstract

In this paper, we propose three algorithms for border landmark selection, namely the Convex Hull-Based (CHB) algorithm, the Center Node Elimination (CNE) algorithm, and the Hierarchy-Structured (HS) algorithm. CHB works perfectly in theory and provides a deep insight into the landmark selection problem. At the same time, it is noticed that CHB is centralized and sensitive to errors in distance estimation. The CNE algorithm is a distributed approach, devised to gradually exclude the nodes in the "center" of the network till the desired number of nodes left, which are employed as landmarks. While CNE works effectively in a small network, its high order computation complexity and communication overhead may eventually lead to scalability problem when it is applied in very large networks. To address this problem, we propose the HS algorithm for striking the balance between accuracy and complexity/overhead. In HS, we establish a hierarchical structure with multiple layers, and apply the CNE algorithm in an appropriate layer to identify an initial set of candidate nodes. The outcomes are then rectified through a recursive process, yielding the final landmarks. Three applications, including coordinates establishment, border detection, and landmark-based routing in general networks without location information, are introduced based on the selected landmarks. We carry out extensive simulations to compare the performance of our landmark selection algorithms and demonstrate their effectiveness in all of the applications.

*This work is supported in part by U.S. Department of Energy (DoE) under Award Number DE-FG02-04ER46136, by Board of Regents, State of Louisiana under Contract Number No. DOE/LEQSF(2004-07)-ULL, and by National Science Foundation CAREER Award under Award Number CNS-0347686.

1. Introduction

The nodes at the border of the self-configurable wireless network are commonly employed as landmarks for many applications, including infrastructureless localization, border detection, and routing. However, how to identify the best set of nodes as such landmarks is still an open problem. To this end, we develop three algorithms for landmark selection, namely the Convex Hull-Based (CHB) algorithm, the Center Node Elimination (CNE) algorithm, and the Hierarchy-Structured (HS) algorithm. The basic idea of the first approach is to identify the convex hull of the network, and choose a subset of the convex hull vertices as the landmarks. CHB works perfectly in theory and provides a deep insight into the landmark selection problem. At the same time, it is noticed that CHB is centralized and sensitive to errors in distance estimation. The CNE algorithm is a distributed approach, devised to gradually exclude the nodes in the "center" of the network till the desired number of nodes left, which are employed as landmarks. While CNE works effectively in a small network, its high order computation complexity and communication overhead may eventually lead to scalability problem when it is applied in very large networks. To address this problem, we propose the HS algorithm for striking the balance between accuracy and complexity/overhead. In HS, we establish a hierarchical structure with multiple layers, and apply the CNE algorithm in an appropriate layer to identify an initial set of candidate nodes. The outcomes are then rectified through a recursive process, yielding the final landmarks. Extensive simulations are carried out to demonstrate the effectiveness of the proposed landmark selection algorithms. Furthermore, we discuss the applications of the landmarks in establishing GPS-free coordinates systems, detecting border nodes in general networks without location information, and efficient routing based on the selected landmarks.

The rest of this paper is organized as follows. Sec. 2

978-1-4244-2575-4/08/$20.00 ©2008 IEEE

introduces our proposed landmark selection algorithms. Sec. 3.1 discusses their applications in coordinates calculation, border detection, and landmark-based routing. Finally, Sec. 4 concludes the paper.

2. Landmark Selection Algorithms

In this section, we discuss the proposed landmark selection algorithms. We first introduce the assumptions and problem formulation. Then the three proposed algorithms are presented in sequence.

2.1. Problem Formulation

Without loss of generality, we consider a network with N nodes, denoted by Ψ. The node ID's are continuously indexed from 1 to N. Assume a trivial routing protocol (e.g., flooding) is available, based on which each node can find the shortest path to other nodes in the network. Note that such trivial routing protocol is employed during network initialization only, for the selection of landmarks. Thereafter, more efficient routing protocol (such as landmark-based routing to be discussed in Sec. 3.3) can be employed.

Let S_{ij} denote the estimated distance between Node i and Node j. Since the two nodes are usually multiple hops away from each other, S_{ij} is an unknown parameter. However, several methods are available to estimate it:

- **Shortest path distance:** S_{ij} can be calculated as the sum of the link distance of all links along the shortest path between i and j, where the link distance may be estimated via Received Signal Strength (RSS) or Time-of-Arrival (ToA) [1], with an error of $\delta\%$.
- **Statistic model-based distance:** This is a more accurate approach, where S_{ij} is obtained by a Euclidean distance estimation model [2], which can map the shortest path between two nodes into their estimated Euclidean distance.
- **Hop count distance:** S_{ij} is simply the number of hops along the shortest path between Nodes i and j.
- **Ideal distance:** For the purposes of illustrating our algorithms and deriving the performance bounds, we also consider the case where S_{ij} is the perfect Euclidean distance between Node i and Node j.

We assume each node, e.g., Node i, knows the set of nodes in the network, i.e., Ψ, the total number of nodes, N, and the set of distances $\{S_{ij}|1 \leq j \neq i \leq N\}$. This assumption is reasonable because such information can be readily obtained from any trivial routing protocol. The goal is to find K border landmarks, forming a polygon that contains the maximum area. Formally, we have the following definition.

Definition 1 *The landmarks are a set of K nodes in the network, which form a polygon with the maximum area.*

2.2. Convex Hull-Based (CHB) Algorithm

The basic idea of the first approach is to identify the convex hull of the network, and choose a subset the convex hull vertices as the landmarks. In the following discussions, we first introduce the basic theories based on a given graph, and then discuss how to apply the proposed approach in real networks.

2.2.1 Basics and Theories

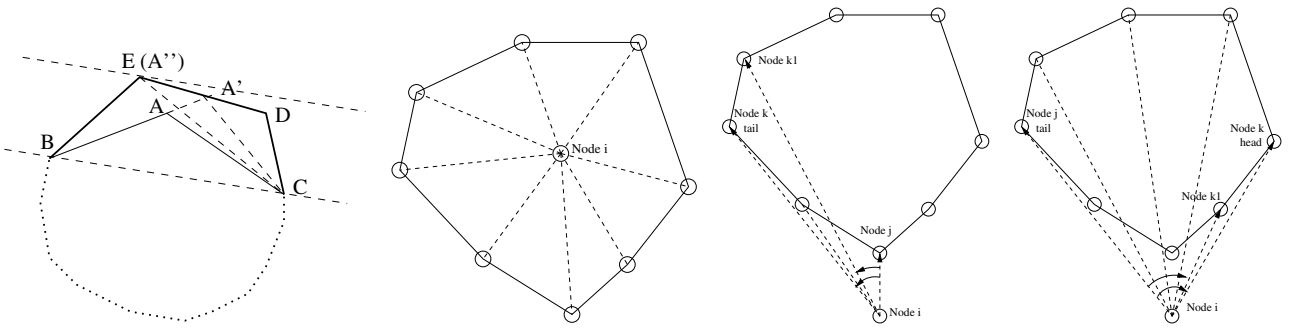
The convex hull of a set of nodes in the plane is defined to be the smallest convex polygon containing all of the nodes. First of all, we'd like to show that the landmarks must overlap with the convex hull. Formally, we establish the following Lemma 1.

Lemma 1 *Given a set of N points in a planar space, the polygon formed by K out of these N points must overlay with the convex hull of the point set, if the polygon has the maximum area among all polygons formed by any K out of N points and the maximum is unique.*

Proof: we prove it by contradiction. Let's denote $S_{B\cdots CDE}$ the area of the convex hull, as shown by the bold solid and dotted lines in Fig. 1(a). Assume we have found the maximum K-polygon with the area of $S_{AB\cdots C}$, but at least one of the vertices of the maximum K-polygon, e.g., point A, doesn't reside on the convex hull. B and C are two neighboring vertices of A on the polygon. According to the definition, A must be inside the convex hull, because the convex hull encloses all points. Obviously we can extend BA (or CA) so that it intersects with an edge of the convex hull at point A'. Let's denote D and E the two vertices that form this edge. Since A is a point within the convex hull, the area of the new polygon denoted by $S_{A'B\cdots C}$ must be greater than $S_{AB\cdots C}$.

Since we have assumed the maximum is unique, segment DE is not parallel to BC. Now, we can push point A' that is on the edge of the convex hull to the vertex A'', which is either D or E, so that $S_{A''B\cdots C} > S_{A'B\cdots C}$. Clearly, if A is on the maximal polygon, it must overlap with A''. Therefore, Lemma 1 is proven. ■

Lemma 1 shows that the landmarks are a subset of the convex hull vertices. This motivates us to find the convex hull in order to identify the optimal landmark nodes. Several algorithms have been proposed in the literature for finding the convex hull of a graph [3, 4]. However, all of them require the coordinates of the vertices of the graph, which are unknown in our problem. Therefore, they cannot be applied here. To this end, we employ an algorithm (i.e., Algorithm 1), which can find the convex hull of the graph based



(a) Illustration of Lemma 1. (b) Note i inside current polygon. (c) Tail extremal node. (d) Head extremal node.

Figure 1. Illustration of Algorithm 1.

on the distances between the vertices only. To facilitate our discussion, let's assume that the exact distance between any pair of nodes is known for now. We will discuss how to apply this algorithm in the real network later in this section.

As shown in Algorithm 1, we initialize $List$ by any three nodes in the network, e.g., Nodes 1, 2 and 3. Then, $List$ will be updated through $N-3$ iterations. In each iteration, a new node may be added into and the existing nodes may be removed from $List$, in order to form a larger polygon. By the end of the iterations, $List$ contains a set of nodes that constitute the convex hull.

More specifically, in an iteration, we pick a Node i to check if it is inside the polygon formed by the nodes in $List$, by using the function $IsInPolygon$. Node i forms an angle with any two consecutive nodes in $List$. If Node i is inside the polygon, all such angles sum up to 2π, as shown in Fig. 1(b). If so, Node i should be discarded, because it is definitely not on the convex hull. Otherwise, we update $List$ to include Node i and possibly remove some nodes from the current $List$. To do so, we identify two extremal nodes on the current polygon (i.e., in $List$), namely tail extremal node and head extremal node, by using the function $FindExtrema$. Figs. 1(c) and 1(d) illustrate how to find the extremal nodes. First, we identify Node j, which is the node in $List$ closest to Node i. Then, we consider $List$ as a cyclic list, and check each node sequentially (i.e., go through every node in the current polygon in clockwise order) until the extremal nodes are found. The extremal nodes form two largest angles with Nodes i and j, on the right side and left side, respectively, as shown in Figs. 1(c) and 1(d). The details of calculating the angles are given in Algorithm 1. Once the two extremal nodes are found, all nodes (if any) between them are removed from $List$, while Node i is inserted into $List$ between the two extremal nodes. Clearly, this insertion guarantees that the nodes in $List$ are maintained in a clockwise order according to their positions on the convex hull. Note that this is done without knowing the coordinates.

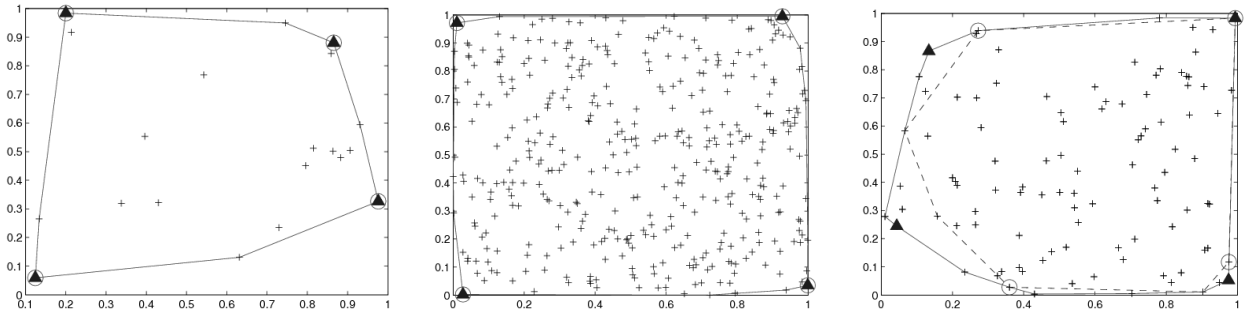
The above procedure repeats until every node in the network has been examined once. Finally, $List$ contains the set of vertices of the convex hull. Since convex hull consists of at least three vertices, we can obtain minimum three landmarks. At the same time, we observe that the number of vertices on the convex hull can be greater than K. Here we adopt a simple approach for the selection of K nodes from $List$. More specifically, we calculate the area of the triangle formed by each vertex and its two neighboring vertices on the convex hull, and sort these triangles in an ascending order according to their areas. The node with the smallest triangle is removed. Then, the areas corresponding to its two neighboring nodes on the convex hull are updated (while other triangular areas remain the same). Repeat the above procedure until only K nodes are left.

Clearly, the total computational complexity of the CHB algorithm is dominated by Algorithm 1, which is $O(N^2)$.

2.2.2 Practice and Discussions

The above convex hull-based algorithm is centralized and based on the known distance between every pair of nodes in a graph. To apply it in the network, one node needs to be the server to collect the distance information and run the algorithm. Some simulation results are shown in Fig. 2, where each "+" represents a node in the network, the red circles represents the landmarks obtained by using our CHB algorithm, and the blue triangles represent the landmarks obtained by brutal search.

Based on the simulation results, we have two observations on CHB. First, the performance of the algorithm (especially Algorithm 1 to find the convex hull) largely depends on the accuracy of the distance estimation. Without distance errors, CHB always results in perfect solution, which is the same as the outcome of brutal search, as shown in Figs. 2(a) and 2(b). However, when we use the shortest path to estimate the Euclidean distance, the results of Algorithm 1 may be substantially degraded, as shown in



(a) No distance estimation errors. $N = 20$. (b) No distance estimation errors. $N = 400$. (c) With estimation errors. $N = 100$.

Figure 2. Results of the convex hull-based algorithm. Each "+" represents a node in the network; the red circles stand for the landmarks obtained by using our CHB algorithm; and the blue triangles are the landmarks obtained by brutal search.

Fig. 2(c). Second, the centralized server may become performance bottleneck and a single point of failure, because it needs to collect the distance information from all other nodes in the network, leading to a worst case communication overhead of $O(N^3)$.

As a result, although the CHB algorithm works perfectly in theory and provides insight into the landmark selection problem, it is not a practical solution.

2.3. Center Node Elimination (CNE) Algorithm

The basic idea of the second approach, dubbed Center Node Elimination (CNE), is to gradually exclude the nodes in the "center" of the network, until there are only K nodes around the "corners" of the network left, which are employed as the landmarks.

The key problem involved in this approach is to identify the nodes in the center of the network, based on the available distance information. To this end, we introduce the nodal *candidacy degree*. For example, the candidacy degree of Node i is

$$C_i = \sum_{j \in \Psi, j \neq i} S_{ij}^2. \tag{1}$$

We argue that the node with smallest candidacy degree is located in the center of the network. Formally, we establish the following Lemma 2.

Lemma 2 *Given a set of nodes, the node which is closest to the center of the network has the smallest candidacy degree.*

Proof: let (x_i, y_i) be the coordinates of Node i in the network.[1] Without loss of generality, we assume $x_i \geq 0$ and $y_i \geq 0, \forall\, i \in \Psi$.

[1]Note that, the coordinates are used in this proof only. Our proposed algorithms do not rely on any coordinates to identify the landmark nodes.

Given the coordinates, C_i can be calculated as below:

$$C_i = \sum_{j \in \Psi, j \neq i} [(x_i - x_j)^2 + (y_i - y_j)^2]. \tag{2}$$

Differentiate C_i and let it equal to 0, we arrive at

$$\begin{cases} x_i = \frac{\sum_{j=1, j\neq i}^{N} x_j}{N-1}, \\ y_i = \frac{\sum_{j=1, j\neq i}^{N} y_j}{N-1}. \end{cases} \tag{3}$$

It is easy to show that the second order derivative of C_i equals to $2(N-1)$, which is positive.

Therefore, we conclude that the node at the center of the network must have the minimum candidacy degree (as indicated in Eq. (3)). ■

Based on Lemma 2, we propose the CNE algorithm, which is summarized in Algorithm 2. First of all, each node acquires the up-to-date distance information, i.e., the distances from itself to all other nodes in the network, through the trivial routing protocol. Then it calculates its candidacy degree according to Eq. (1), and advertises it to other nodes in the network. Although the advertisement is essentially a flooding process, improvement can be made to reduce the flooding overhead. For example, a node immediately drops all other advertisements if it receives an advertisement with smaller candidacy degree. Therefore, only one advertisement with the smallest candidacy degree will be eventually propagated to all nodes in the network. Consequently, this node will be removed from Ψ. Note that this is done by every individual node in a distributed way. The above procedure repeats $N - K$ time, until only K nodes are left.

Lines 5, 8, and 9 of Algorithm 2 result in a computational complexity of $O(N)$. Since each node repeats the iteration $N-K$ time, the overall complexity is $O(N^2)$. On the other hand, the communication overhead is mainly introduced by

Algorithm 1: Convex Hull Selection

$List = [1, 2, 3]$;
for $i = 4 : N$ **do**
  if `IsInPolygon` *(i, List)* **then**
    do nothing;
  else
    $(head, tail) \leftarrow$ `FindExtrema`$(i, List)$;
    $List \leftarrow List - \{head..tail\}$;
    $List \leftarrow List + i$;
  end
end

`IsInPolygon` (i, List)
foreach $j \in List$ **do**
  $\alpha_j \leftarrow \arccos \frac{d_{ij}^2 + d_{ij+1}^2 - d_{jj+1}^2}{2 * d_{ij} * d_{ij+1}}$;
end
if $\sum_{j=1}^{n} \alpha_j = 2\pi$ **then**
  return true;
else
  return false;
end

`FindExtrema` (i, List)
$tail = head = 0$;
$j \leftarrow$ `FindClosest`$(i, List)$;
$k \leftarrow$ `NextItem`$(j, List)$;
$\alpha_0 \leftarrow \arccos \frac{d_{ij}^2 + d_{ik}^2 - d_{jk}^2}{2 d_{ij} d_{ik}}$;
$k1 \leftarrow$ `NextItem`$(k, List)$;
while *True* **do**
  $\alpha \leftarrow \arccos \frac{d_{ij}^2 + d_{ik1}^2 - d_{jk1}^2}{2 d_{ij} d_{ik1}}$;
  if $\alpha < \alpha_0$ **then**
    if $tail = 0$ **then**
      $tail \leftarrow k$; $j \leftarrow k$;
      $k \leftarrow$ `NextItem`$(j, List)$;
      $\alpha_0 \leftarrow \arccos \frac{d_{ij}^2 + d_{ik}^2 - d_{jk}^2}{2 d_{ij} d_{ik}}$;
      $k1 \leftarrow$ `NextItem`$(k, List)$;
    else
      $head \leftarrow k$; break;
    end
  else
    $\alpha_0 \leftarrow \alpha$; $k \leftarrow k1$;
    $k1 \leftarrow$ `NextItem`$(k1, List)$;
  end
end

`FindClosest` (i, List)
Return j, with $d_{ij} \leq d_{ik}, \forall k \in List$;

`NextItem` (i, List)
Return the next (cyclically) entry in $List$ after the one that contains Node i;

Algorithm 2: Center Node Elimination (CNE) Algorithm

Input: Ψ, N, K;
Output: Φ;
1 $i \leftarrow$ Node ID;
2 Read in $\{S_{ij} | 1 \leq j \neq i \leq N\}$;
3 **for** $m = 1 : N - K$ **do**
4 **if** $i \in \Psi$ **then**
5 $C_i = \sum_{j \in \Psi, j \neq i} S_{ij}^2$;
6 Advertise C_i;
7 Collect $\{C_j | j \in \Psi\}$;
8 Identify Node c, with $C_c \leq C_j, \forall\, j \in \Psi$;
9 $\Psi = \Psi - c$;
10 **else**
11 GoTo Line 15;
12 **end**
13 **end**
14 Advertise i as a landmark node;
15 Collect landmark advertisements to create Φ.

the advertisement of C_i (as shown in Lines 6 and 7 of Algorithm 2). In each iteration, the worst case overhead is $O(N^2)$. Thus the total overhead of the algorithm is $O(N^3)$.

To validate the effectiveness of CNE, we have implemented and tested it in various networks, where the distance (S_{ij}) is estimated by the shortest path method discussed in Sec. 2.1. A few representative results are shown in Fig. 3. As can be seen, the proposed algorithm can always identify the landmarks around the corners of the network. Compared to the CHB algorithm, CNE is very robust to the possible errors in distance estimation. More discussions and performance comparison of the algorithm will be given in Sec. 2.5.

2.4. Hierarchy-Structured (HS) Algorithm

While CNE works effectively in a small network, its high order computation complexity and communication overhead may eventually lead to scalability problem when it is applied in very large networks. To address this problem, we propose the HS algorithm for striking the balance between accuracy and complexity/overhead. In HS, we establish a hierarchical structure with multiple layers, and apply the CNE algorithm in an appropriate layer to identify an initial set of candidate nodes. The outcomes are then rectified through a recursive process, yielding the final landmarks.

Algorithm 3: Hierarchy-Structured (HS) Algorithm

Input: N, K, M, $\{\Psi_j | 0 \leq j \leq R_{max} = \lfloor \log_M N \rfloor\}$;
Output: Φ_0;
1 $i \leftarrow$ Node ID;
2 Read in $\{S_{ij} | 1 \leq j \neq i \leq N\}$;
3 $R_i = \max\{R_i | i \mod M^{R_i} = 0\}$;
4 $r = \min\{r | \sum_{j=r}^{R_{max}} |\Psi_j| \geq \beta\}$;
5 $\Psi = \sum_{j=r}^{R_{max}} \Psi_j$;
6 **if** $i \in \Psi$ **then**
7 | Run Algorithm 2;
8 **end**
9 Collect landmark advertisements to create Φ_r;
10 **if** $i \notin \Phi_r$ **then**
11 | **if** $i \in \Psi_{r-1}$ **then**
12 | | Identify the nearest landmark node $k \in \Phi_r$;
13 | | $C_i = \sum_{j \in \Phi_r, j \neq k} S_{ij}^2$;
14 | | Register (i,C_i) to Node k;
15 | **end**
16 **else**
17 | Collect registrations to create Ω_r^i;
18 | $C_i = \sum_{j \in \Phi_r, j \neq i} S_{ij}^2$;
19 | $\Omega_r^i = \Omega_r^i \bigcup \{(i, C_i)\}$;
20 | Identify Node $c \in \Omega_r^i$, with $C_c \geq C_p, \forall\, p \in \Omega_r^i$;
21 | Advertise c to be a landmark node in Φ_{r-1};
22 **end**
23 r=r-1;
24 **if** $r > 0$ **then**
25 | GoTo Line 9;
26 **end**

Various clustering algorithms and protocols have been discussed in the literature to establish the hierarchical structure. Here we employ a very simple and effective scheme, while many other approaches are also applicable. Let M be a predetermined module number. The hierarchical layer of Node i, denoted by R_i, is defined as the maximum times that i can recursively modulo M to 0, i.e., $R_i = \max\{R_i | i \ mod \ M^{R_i} = 0\}$. Let Ψ_k denote the set of nodes in Layer k. For example, if $M = 10$, then the nodes in $\Psi_1 = \{10, 20, ..., 90\}$ all have $R_i = 1$; the nodes in $\Psi_2 = \{100, 200, 300, ..., 900\}$ are in the second layer; and most other nodes (in Ψ_0) are in Layer 0.

R_i is calculated by Node i in a distributed way. Given the size of the network (i.e., N), it is easy to estimate the size of each layer (i.e., the number of nodes in each layer), denoted by $|\Psi_k|$, $0 \leq k \leq R_{max}$ where $R_{max} = \lfloor \log_M N \rfloor$.

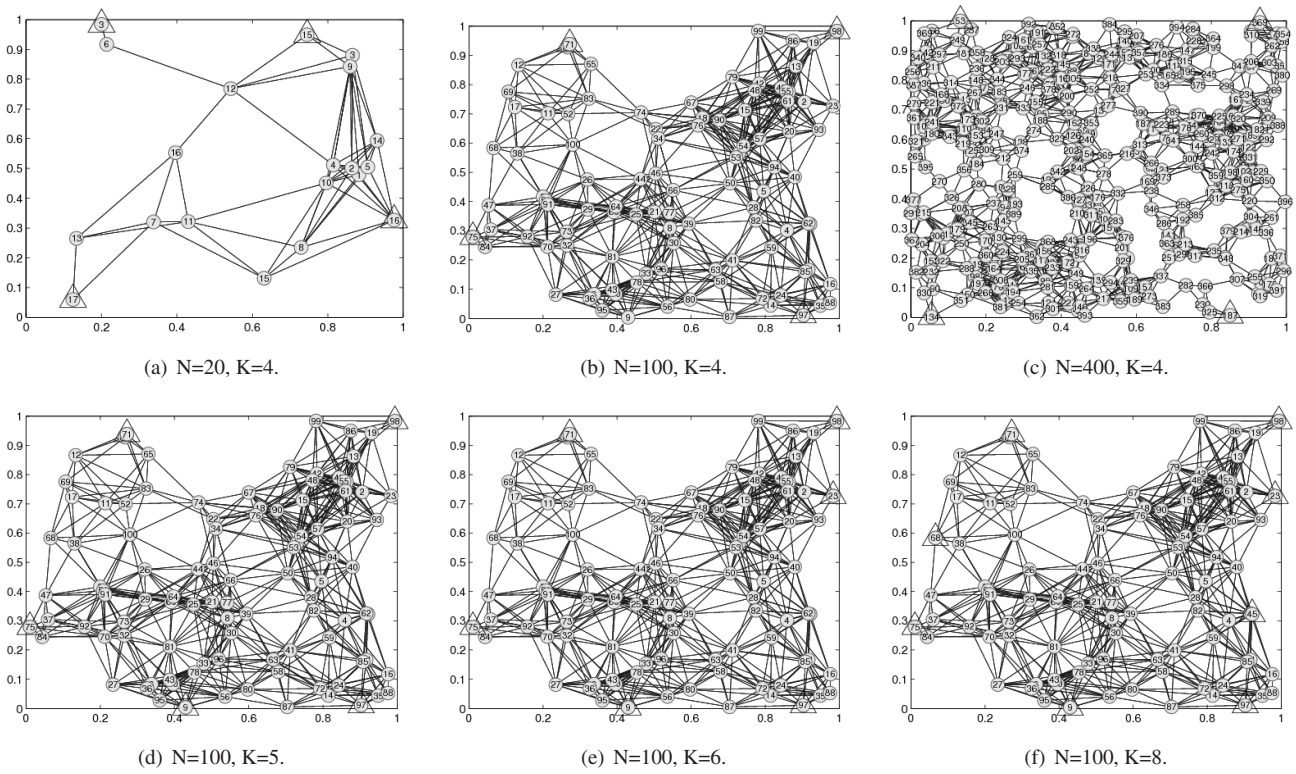
The hierarchy-structured approach is summarized in Algorithm 3. Its basic idea is to limit the number of nodes involved in the CNE algorithm, and thus reduce complexity and overhead. At the same time, we also notice that the accuracy may be degraded, if very few nodes participate in the algorithm. Thus, we define a threshold β, which is the minimum number of nodes that should be employed to run the algorithm. Therefore Ψ is initialized to be a set of nodes in the top r layers, so that the total number of nodes is greater than β, as shown in Lines 4-5 of Algorithm 3.

Then we run the CNE algorithm based on Ψ, which results in a set of landmark nodes. This initial set of landmark nodes is denoted by Φ_r, because the algorithm is based on the nodes in Layers r and above. Since the number of nodes in Ψ is around the constant β, the computational complexity and communication overhead of running the CNE algorithm are both $O(1)$.

Clearly, the landmark nodes obtained so far are not necessary to be the best, because they are selected from a subset of nodes in the network. Hereafter, we try to update Φ_r by considering other nodes in lower layers. If Node i is not in Φ_r (i.e., it is currently not a landmark node) but it is in Ψ_{r-1} (i.e., in Layer $r-1$), it is associated with the nearest node in Φ_r, e.g., Node k. It calculates its candidacy degree, C_i, and sends to Node k a registration message that includes (i, C_i), as illustrated in Lines 11-15. Note that, here, the calculation of C_i is similar to but different from Eq. (1) used in CNE. Eq. (1) is motivated by Lemma 2, which shows that a node is at the center of the network if the total distance from itself to all other nodes in the network is minimum. Here we try to find whether there exists a better candidate near the current landmark node (i.e., Node k) which can replace Node k as the landmark node.More specifically, Node i in Ψ_{r-1} calculates its C_i according to $C_i = \sum_{j \in \Phi_r, j \neq k} S_{ij}^2$, which is the total distance from Node i to all current landmark nodes except Node k. In the mean time, the current landmark node collects the registrations from its nearby nodes in Ψ_{r-1} and elects the one with the largest candidacy degree as the new landmark node (see Lines 17-21). Clearly both the computational complexity and the communication overhead of the Lines 10-22 are in the order of $O(N)$.

The above procedure for refining the set of landmark nodes repeats until the lowest layer is reached, which produces the final result, Φ_0. The number of iterations is up to $\lfloor \log_M N \rfloor + 1$. Thus the total computational complexity and the communication overhead of Algorithm 3 are $O(N \log N)$.

Figs. 4(a)-4(c) demonstrate the results after each iteration of the HS algorithm. The network consists of 400 nodes. Fig. 4(a) shows the first iteration, where twenty two nodes (represented by the solid circles) are in top layers that form Φ. As we can see, the four landmark nodes elected based on this small set of nodes are not really close to the corner of the network. In the second iteration (see Fig. 4(b)), better candidates are identified to replace the previous landmarks. After the third iteration, the results shown in Fig. 4(c) are very close to the results when we directly apply Algorithm 2 on the whole network.



(a) N=20, K=4. (b) N=100, K=4. (c) N=400, K=4.

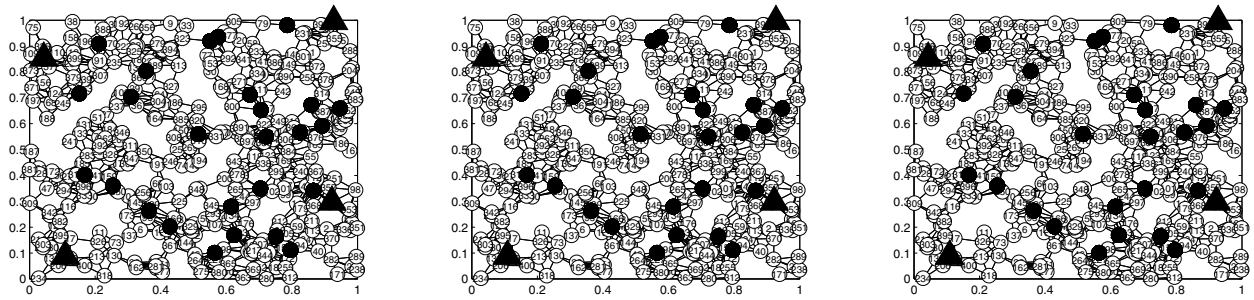
(d) N=100, K=5. (e) N=100, K=6. (f) N=100, K=8.

Figure 3. Examples of employing the CNE algorithm in networks with different N and K. The circles stand for wireless nodes, while the triangles are the landmark nodes identified by CNE.

2.5. Simulation Results

Simulations are carried out in Matlab to evaluate our proposed landmark selection algorithms. We consider a rectangular area, in which a number of nodes are randomly distributed. We have implemented the three proposed algorithms under various distance estimation schemes, including ideal distance, statistic model-based distance, shortest path distance, and hop count distance, as we have discussed in Sec. 2.1. For comparison, we also include results based on random selection of landmarks, which is denoted by RND. The network topologies used in our simulations are randomly generated graphs. According to Definition 1, the landmark selection algorithm aims to identify K landmarks that form the polygon with the maximum area. To judge the effectiveness of our proposed approaches, we define the *coverage ratio*, i.e., S/S', where S denotes the area of the polygon that comprises the landmarks selected by our algorithms and S' is the area of the maximum K-polygon found by brutal search. The coverage ratio serves as a general performance indicator of the landmark selection algorithms. Their performance in specific applications such as coordinates calculation, border detection, and landmark-based routing will be discussed later in Sec. 3. In addition to the coverage ratio, we are also interested in overhead of our proposed algorithms, which is measured by the number of message transmissions during the algorithm execution.

Fig. 5 compares the coverage ratio of different landmark selection algorithms, based on various distance estimation methods. All results are obtained in a random network topology with $N = 100$ nodes and $K = 4$ landmarks. As can be seen, when the ideal distance is assumed, CHB achieves perfect coverage ratio; CNE also performs very well; while the coverage ratio of HS is lower, mainly due to the approximation in its hierarchical refinement of landmark selection. The results under statistic model-based distance are similar to those under the ideal distance, because the statistic model yields estimated distance very close the actual distance. Thus the results are omitted in Fig. 5. When the shortest path or hop count is used for distance estimation, however, the performance of CHB degrades dramatically, since this algorithm is sensitive to distance errors. If hop count is used for distance estimation, only 40% coverage ratio can be achieved by CHB. On the other hand, CNE yields constant and high coverage ratio under all distance estimation methods, exhibiting excellent tolerance to dis-



(a) Top layer. (b) Middle layer. (c) Lowest layer.

Figure 4. Examples of hierarchy-based approach. $N = 400$**,** $K = 4$**.**

tance errors. The performance of HS is lower than but still comparable with CNE.

At the same time, HS results in the lowest overhead. HS builds the hierarchical structure and runs the CNE algorithm at the top layers only. The landmark refinement process in the lower layers are done locally, thus significantly reducing overhead. As depicted in Fig. 6, the overhead of HS is about one order of magnitude lower than that of CNE. Based on the above observation, we deem HS as the best approach that effectively strikes the balance between efficiency and overhead. In addition, Fig. 7 shows the coverage ratio when the number of selected landmarks (i.e., K) varies from 4 to 8. As can be seen, K doesn't significantly affect the coverage ratio in our simulations. All algorithms work effectively to identify any number of landmarks.

As we have discussed in Sec. 2.4, while the hierarchical structure reduces overhead, it pays the cost of possibly degraded coverage ratio. To study this tradeoff, we have carried out simulations by varying the layer structure. For simplicity, we run CNE at the highest layer only. Thus the more layers in the hierarchical structure, the smaller number of nodes are involved in the CNE algorithm. As shown in Fig. 8, the coverage ratio decreases about 10% when the number of layers increases from 1 to 5. Meanwhile, there is a dramatic overhead reduction of up to 90%. The choice of the hierarchical structure depends on the application, particularly, its accuracy requirement and overhead tolerability. Based our extensive simulations, we observe that a 3-layer structure fits most scenarios, with only marginal loss of coverage ratio but significant save of signaling overhead.

3. Applications of Landmarks

Once the landmarks are identified, they can be employed for coordinates calculation as well as border node detection and landmark-based routing, as outlined below.

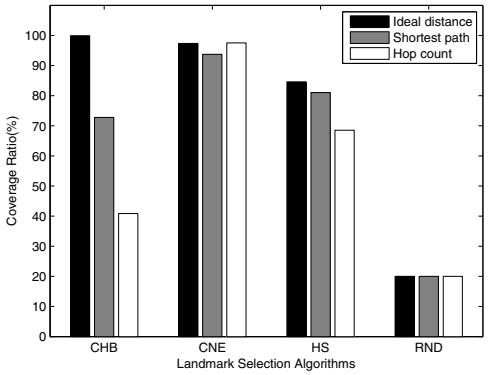


Figure 5. Coverage ratio under different landmark selection algorithms.

3.1. Coordinates Calculation

A few infrastructureless localization schemes have been proposed recently, mainly to provide location information in mobile ad hoc networks and wireless sensor networks [5]. Here, we focus on the method proposed in [2], which can make efficient use of the border landmarks to establish the coordinates system. Any landmark (i.e., a node in Φ_0) can initiate the coordinates calculation, by making an announcement to all other nodes in Φ_0 and acquiring their distance information.[2] More specifically, each node in Φ_0, e.g., Node i, sends $\{S_{ij}|j \neq i, j \in \Phi_0\}$ to the initiator.

Denote the coordinates of Node i as (x_i, y_i), where x_i and y_i are valuables to be determined. Then the distance between two landmark nodes i and j in the established coordinates system can be written as $L_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. Our objective is to assign the appropriate coordinates to the landmark nodes, such that

[2] If more than one landmarks intend to be the initiator, other nodes yield to the one with the lowest ID.

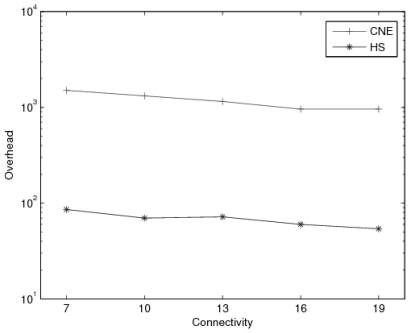


Figure 6. Overhead Comparison.

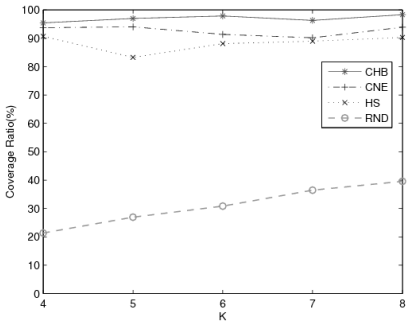


Figure 7. Coverage ratio vs. number of landmarks.

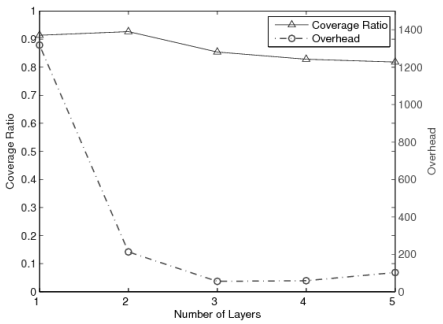


Figure 8. Impact of hierarchical layers.

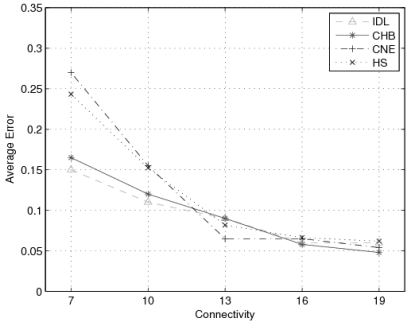


Figure 9. Coordinates errors.

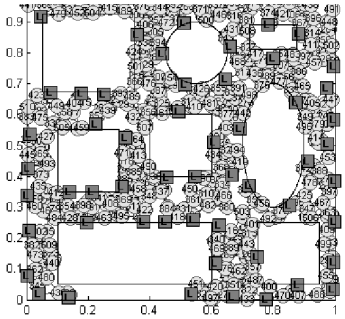


Figure 10. Border node detection ($K = 64$).

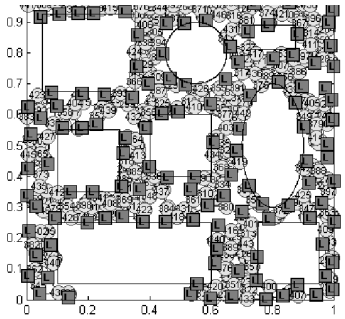


Figure 11. Border node detection ($K = 128$).

the error between L_{ij} and S_{ij} is minimized. To this end, we define the error objective function to be

$$\epsilon = \sum_{i \in \Phi_0} \sum_{j \in \Phi_0, j \neq i} (\frac{S_{ij} - L_{ij}}{S_{ij}})^2. \quad (4)$$

Then, a Simplex method [6] is used to minimize ϵ and give the coordinates for each landmark node. The Simplex program has been implemented in many softwares. Our simulation (to be discussed in Sec. 2.5) adopts the Simplex module available in Matlab [7] to minimize the error objective function. After being determined by the initiator, the coordinates of the landmark nodes are advertised over the network. Note that such coordinates system is stand-alone. But it can be easily translated into the global positioning system if at least one node in the network is equipped with a GPS receiver.

The node that is not in Ψ_0 (i.e., not a landmark node) can obtain its coordinates through a similar error minimization method. Consider Node p whose coordinates are denoted by (x_p, y_p), we define the error objective function to be $\epsilon_p = \sum_{i \in \Psi_0} (\frac{S_{ip} - L_{ip}}{S_{ip}})^2$, where $L_{ip} = \sqrt{(x_i - x_p)^2 + (y_i - y_p)^2}$. Again, the Simplex method is used to minimize the error function ϵ_p and determine the coordinates (x_p, y_p).

We have simulated the above coordinates calculation algorithm, based on different landmark selection algorithms where the statistic model-based method is employed for distance estimation. The average errors of coordinates are shown in Fig. 9. HS and CNE always result in similar performance. Under low connectivity, HS and CNE exhibit higher errors compared with the centralized CHB and the ideal solution (obtained via brutal search). But with the increase of connectivity, their performance becomes almost identical. This again verifies our observations discussed in Sec. 2.5, showing the effectiveness of the proposed distributed algorithms in coordinates calculation.

3.2. Border Detection

In addition to the establishment of GPS-free coordinates systems, the proposed CNE and HS algorithms are also useful to identify boarder nodes in general networks without location information, because the algorithms tend to discover the nodes at the border of the network (either exterior border or interior border if there is any void space inside the net-

work coverage area). For example, Figs. 10-11 illustrate the results for border detection in a sensor network with "holes" in various shapes. In Fig. 10, $K = 64$ nodes are identified to outline the borders, while Fig. 11 shows the scenario with $K = 128$ border nodes. As can been seen, our landmark selection algorithm can effectively identify both exterior and interior borders, achieving similar results as [8]. A larger K usually yields more complete contour of the network and the holes. These results are obtained by using the CNE algorithm based on hop-count distance, while similar results have been observed under other proposed algorithms and distance estimation methods.

3.3. Landmark-based Routing

Another interesting application of the landmarks is in routing. Our proposed border landmark selection algorithms can be applied in many landmark-based routing protocols [9–11]. The basic idea of landmark-based routing is to define virtual coordinates according to the landmarks and then apply greedy forwarding based on the virtual coordinates. Here we adopt the basic idea of [11], but with modified distance formulation to achieve higher efficiency. More specifically, let Φ_0 denote the set of landmarks identified by the algorithms discussed in Sec. 2. The landmarks periodically broadcast advertisements, which are used by the nodes in the network to estimate their distances to each of the landmarks. Any types of distances that are discussed in Sec. 2.1 can be employed here. Our implementation is based on "hop count distance", i.e., the number of hops along the shortest path between two nodes, which needs least hardware/software support. Let S_{il} denote the distance from Node i to Landmark l, where $l \in \Phi_0$. The virtual coordinates of Node i is defined as a vector, denoted by $V_i = [S_{il_1}, S_{il_2}, \cdots, S_{il_k}, \cdots]$, where $l_k \in \Phi_0$. Based on the virtual coordinates, the virtual distance from Node i to Node j is defined in a way similar to Mahalanobis distance:

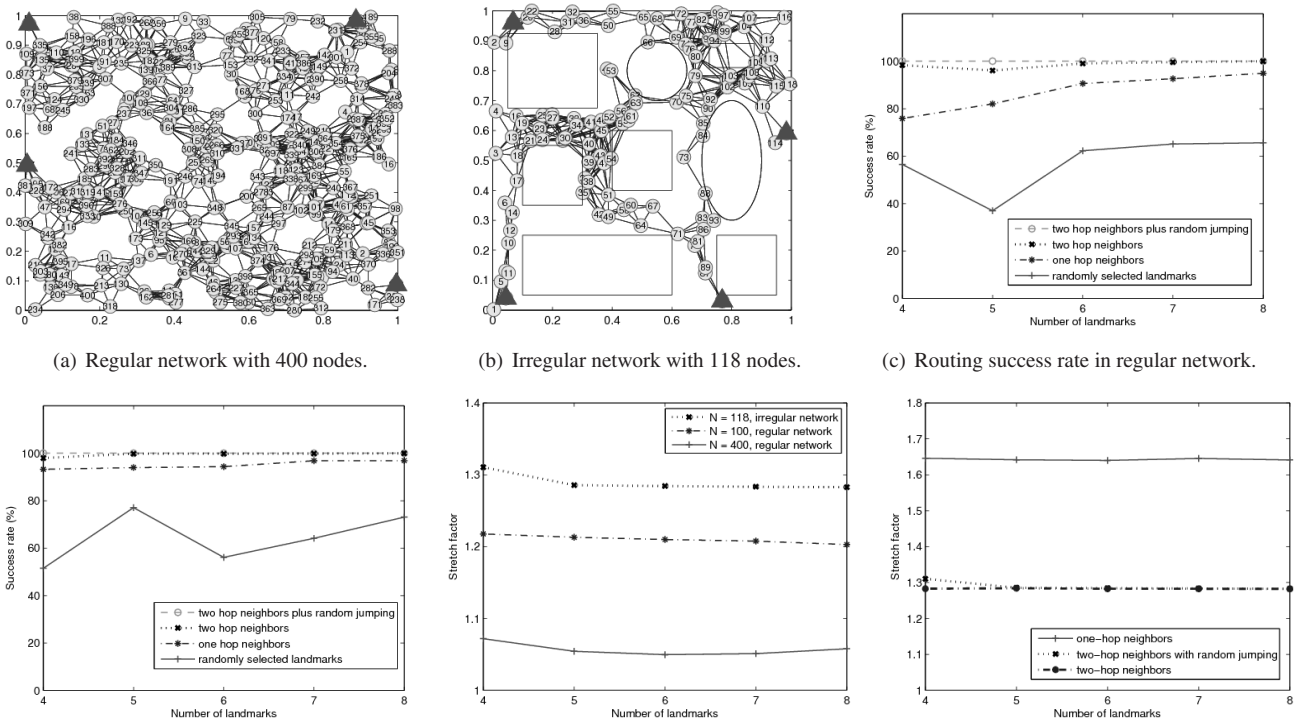
$$D_{ij} = ||V_i - V_j|| = \sqrt{\sum_{l_k \in \Phi_0} (\frac{S_{il_k} - S_{jl_k}}{S_{jl_k}})^2}. \quad (5)$$

Each node in the network maintains a set of its neighbors and their virtual coordinates. The "neighbors" can be the nodes within one hop, or two hops, or multiple hops. Let $N^{(k)}(i)$ denote the set of k-hop neighbors of Node i, including Node i itself. In our landmark-based routing scheme, the virtual coordinates of the destination is enclosed in the packet header. Whenever a node receives a packet, it identifies the best next hop for routing the data packet to its destination. The routing strategy is similar to the greedy forwarding algorithm, except that the geographic coordinates are replaced by the virtual coordinates. More specifically, assume Node i receives a packet for destination Node d. It then chooses the node with the shortest distance to Node d as next hop, i.e., $\nu = \arg\min_{j \in N^{(k)}(i)} D_{jd}$. This procedure repeats until the packet reaches its destination or $\nu = i$ (which indicates a failure in routing).

Similar to all greedy forwarding protocols in the literature, failures may occur in the landmark-based routing, especially when the network contains void areas. We propose two schemes to address this problem. The first scheme is to adjust $N^{(k)}(i)$, which involves a tradeoff in routing success rate and overhead. Choosing a larger k will increase the probability to select the best next hop and to avoid the "dead end" problem (i.e., $\nu = i$ for a non-destination node i), with the cost of higher overhead in maintaining the neighbor information. Our simulation results (to be discussed next) show that one can achieve close to 100% success rate by choosing $k = 2$. The second approach is random jumping. More specifically, when Node i finds $\nu = i$, it randomly forwards the packet to a node k-hops away, which in turn tries to route the packet to its destination. This approach is particularly useful in scenarios with extremely complex network topologies.

We have carried out simulations for evaluating the landmark-based routing protocol. We have considered two network scenarios: a regular network with nodes uniformly distributed in a unit square (see Fig. 12(a) for the regular network of 400 nodes) and an irregular network with void areas in various shapes inside the unit square (see Fig. 12(b) for the irregular network of 118 nodes). All landmark selection algorithms are simulated to evaluate their performance in landmark-based routing. Our simulation results show that the CHB algorithm is not effective here, due to the use of hop count distance, which leads to poor landmark selection or even failure in CHB as discussed in Sec. 2.2. CNE and HS algorithms yield almost identical routing performance. Thus the simulation results to be discussed next are based on either of these two algorithms. Four landmarks are employed in all of our simulations. Note that, although CNE and HS performs significantly better than CHB and random landmark selection, the selected landmarks may still deviate from true landmarks as we have discussed in Sec. 2.5, since the simple hop count distance is employed (see the landmarks indicated by solid triangles in Figs. 12(a) and 12(b)). Such deviation, however, does not affect the effectiveness of our landmark-based routing protocol as evident by our simulation results elaborated below, although better landmarks result in higher routing performance.

We are mainly interested in the routing success rate. Fig. 12(c) shows the results in the regular network. As can be seen, landmark selection is crucial for the routing performance. Under randomly selected landmarks, the success rate is merely about 60%. Our proposed landmark selection algorithms lead to significantly improved performance. With the basic landmark-based routing (using one-



(a) Regular network with 400 nodes. (b) Irregular network with 118 nodes. (c) Routing success rate in regular network.

(d) Routing success rate in irregular network. (e) Impact of network topology on stretch factor. (f) Impact of routing schemes on stretch factor.

Figure 12. Simulation results of landmark-based routing.

hop neighbors only), the routing success rate approaches 98% when the number of landmarks increases from 4 to 8. If two-hop neighbors (i.e., $N^{(2)}(i)$) are considered in making routing decisions, the performance will be further improved to around 100% success rate. This improvement is particularly notable when a small number of landmarks (e.g., four landmarks) are employed. We have also incorporated the "random jumping" technique and observed that it helps the landmark-based routing protocol to achieve 100% success ratio in all scenarios we have tested. In the irregular network, our landmark-based routing protocol performs similarly, as illustrated in Fig. 12(d).

In addition to routing success rate, we have also studied routing overhead. To this end, we define a metric dubbed *stretch factor*, which is the ratio between actual routing length (in terms of number of hops) and the shortest path length. It serves as an indicator of routing efficiency and overhead. In general, the stretch factor is greater than one. First we study the impact of network topology on stretch factors. Three network scenarios are considered in this endeavor: a regular network with 100 nodes, a regular network with 400 nodes, and an irregular network with 118 nodes. As shown in Fig. 12(e), the stretch factor is smaller in regular networks, compared with the irregular network, because the void areas in the irregular network may lead to less efficient routing decisions and longer routing paths. The stretch factor also decreases when the network becomes larger. Second, we study the stretch factors under different routing techniques, including routing with one-hop neighborhood, two-hop neighborhood, and two-hop neighborhood plus random jumping. The simulation is carried out in the irregular network of 118 nodes. As revealed by the simulation results in Fig. 12(f), using two-hop neighborhood in routing can significantly reduce the stretch factor. This is understandable, because a better route can usually be chosen by considering more nodes in making the routing decisions. On the other hand, random jumping leads to minor or no increase in stretch factor. This, together with our earlier result of 100% routing success rate achieved by random jumping, suggests that it is an effective routing technique with minimum overhead and should be incorporated in the landmark-based routing.

4. Conclusion

In this paper, we have proposed three algorithms for border landmark selection, namely the Convex Hull-Based (CHB) algorithm, the Center Node Elimination (CNE) algorithm, and the Hierarchy-Structured (HS) algorithm. CHB works perfectly in theory and provides a deep insight into

the landmark selection problem. At the same time, it has been noticed that CHB is centralized and sensitive to errors in distance estimation. The CNE algorithm is a distributed approach, devised to gradually exclude the nodes in the "center" of the network till the desired number of nodes left, which are employed as landmarks. While CNE works effectively in a small network, its high order computation complexity and communication overhead may eventually lead to scalability problem when it is applied in very large networks. To address this problem, we have proposed the HS algorithm for striking the balance between accuracy and complexity/overhead. In HS, we establish a hierarchical structure with multiple layers, and apply the CNE algorithm in an appropriate layer to identify an initial set of candidate nodes. The outcomes are then rectified through a recursive process, yielding the final landmarks. Furthermore, we have introduced three applications of the landmarks in establishing GPS-free coordinates systems, detecting border nodes, and efficient routing in general networks without location information. We have carried out extensive simulations to compare the performance of our landmark selection algorithms and demonstrate their effectiveness in the applications.

References

[1] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices," *IEEE Personal Communications Magazine*, vol. 7, no. 5, pp. 28–34, 2000.

[2] H. Wu, C. Wang, and N. Tzeng, "Novel self-configurable positioning technique for multi-hop wireless networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 3, pp. 609–621, 2005.

[3] F. P. Preparata and S. Hong, "Convex hulls of finite sets of points in two and three dimensions," *Communications of the ACM*, vol. 20, no. 2, pp. 87–93, 1977.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, pp. 947–957. MIT Press and McGraw-Hill, 2001.

[5] D. Perkins, R. Tumati, H. Wu, and I. Ajbar, *Localization in Wireless Ad Hoc Networks*, ch. Resource Management in Wireless Networking. Springer, 2005.

[6] J. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308–313, 1965.

[7] Matlab. `http://www.mathworks.com/`.

[8] Y. Wang, J. Gao, and J. S. Mitchell, "Boundary recognition in sensor networks by topological methods," in *Proceedings of the 12th Annual ACM International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 122–133, 2006.

[9] R. Fonesca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon vector routing: Scalable point-to-point routing in wireless sensornets," in *Proceedings of the 2nd Symposium on Networked Systems Design and Implementation*, 2005.

[10] Q. Fang, J. Gao, L. Guibas, V. de Silva, and L. Zhang, "Glider:gradient landmark-based distributed routing for sensor networks," in *Proceedings of the 24th Conference of the IEEE Communication Society (INFOCOM)*, 2005.

[11] Y. Zhao, B. Li, Q. Zhang, Y. Chen, and W. Zhu, "Efficient hop id based routing for sparse ad hoc networks," in *Proceedings of the 13TH IEEE International Conference on Network Protocols*, pp. 179–190, 2005.