XLR: Tackling the Inefficiency of Landmark-based Routing in Large Wireless Sensor Networks

Xuetao Wei, Nicholas C. Valler, Michalis Faloutsos, Harsha V. Madhyastha, and Ting-Kai Huang
Department of Computer Science and Engineering, University of California - Riverside
Email: {xwei, nvaller, michalis, harsha, huangt}@cs.ucr.edu

***Abstract*—Landmark-based routing (LR) provides a promising approach for scalable point-to-point routing in wireless sensor networks (WSNs). Though various approaches have been proposed for landmark-based routing, they either introduce significant computational complexity or are inefficient in realistic, dynamic environments. In this paper, we identify three design principles that form the basis of efficiency: algorithmic simplicity, update efficiency, and application awareness. Motivated by these principles, we present *XLR*, a new, flexible and comprehensive framework that tackles the inefficiency of landmark-based routing. *XLR* consists of four components: Relay Selection (RS), Parametric P-Norm distance function (PPN), Efficient Update with Coordinate Difference (EUCD) and General Forwarding (GF). The key advantage of *XLR* is that any subset of *XLR*'s components can be independently incorporated into most landmark-based routing protocols. We perform extensive simulations to demonstrate that: (i) RS, a simple method, yields good performance comparable with previous methods, (ii) PPN increases LR performance considerably, (iii) EUCD reduces coordinate update overhead by up to 39%, and (iv) our GF outperforms previous approaches that consider factors such as link quality, delay and power consumption independently.**

I. INTRODUCTION

Point-to-point routing methods have become an important topic in emerging wireless sensor networks (WSNs), as evidenced by the recent formation of IETF working groups [1] and various research contributions [2], [3], [4], [5], [6], [7], [8], [9]. In this paper, we focus on landmark-based routing (LR)[1], a type of point-to-point routing protocol derived from geographic routing and virtual coordinate-based routing.

Fundamentally, this paper attempts to answer the following question: *Given a large wireless sensor network, what improvements to landmark-based routing protocols will increase overall WSN performance?* Our solution, XLR, is a new, flexible and comprehensive framework that tackles the inefficiencies of LR by providing components that may be incorporated in a piecewise fashion into existing LR protocols.

At the core of any LR protocol are 4 primitive operations: 1) landmark selection; 2) coordinate establishment and maintenance; 3) greedy forwarding; and 4) routing failure recovery. We will illustrate typical LR using in Fig. 1 as an example. Using some *landmark selection* algorithm, an initial set of landmarks $\mathbf{L} \subset N$ are selected from among the complete set of nodes N. Once selected, the *coordinated system* must be established. Each landmark will construct a shortest path tree, rooted at the landmark, typically using hop count as the distance metric. Each node i constructs its coordinate C_i using shortest distances to each landmark $L_i \in \mathbf{L}$. In Fig. 1, we see that node 7 has a coordinate of $C_7 = \{5, 4, 1\}$, where each element

[1]In this paper, the term "Landmark-based Routing" is distinct from virtual coordinate-based routing protocols [10] [11], which do not use hop count in the coordinate vector. Furthermore, certain hierarchical routing protocols [12] also use the term "landmark," but do not fall under the general category of LR.

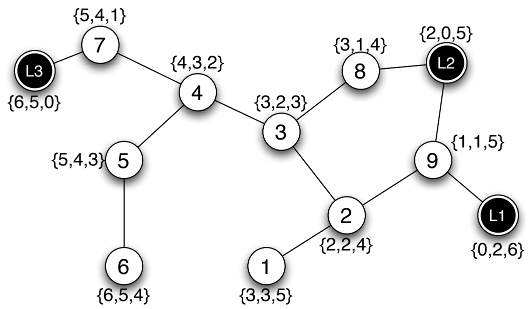


Fig. 1. An Example of Landmark-based Routing

is the distance to a landmark, e.g., 5 hops to landmark L_1, etc. The coordinates of nodes are typically published in some form of distributed *location service* [13], which is out of the scope of this paper. Furthermore, due to the dynamic wireless environment, the accuracy of coordinate system must be maintained. Therefore, LR use some form of *update message* to signal changes in a node's coordinate to its neighbors and the location service. In order to route packets from a source node i to a destination node j, each node on the path employs some form of *greedy forwarding* to identify the *next-hop* neighbor. This is done by calculating the distance from i's neighbors K_i. Thus, the next-hop node will be the neighbor $k \in K_i$ with the smallest distance to j, as calculated by the *distance function* $\delta(C_k, C_j)$. In our example, to route packets from node n_2 to node n_5, n_2 begins by looking up n_5 coordinate through location service [13]. Knowing C_2 and C_5, n_2 calculates the distance from all its neighbors $K_2 = \{n_1, n_3, n_9\}$ and computes $\delta(C_{K_2}, C_5)$. In this example, best solution for the next hop is n_3. But, if a routing error occurs, such as routing to n_1 (an obvious dead-end), the protocol could re-route the packets using costly *routing failure recovery* strategies, e.g. backtrack, fallback or scoped flooding [2], [3], [4]. Recovery strategy is beyond the scope of this paper and we do not address it herein.

To optimize LR protocols to meet the needs of WSNs, we identify 3 necessary design principles 1) *algorithmic simplicity*; 2) *update efficiency*; and 3) *application awareness*. We incorporate these design principles into XLR. Our intent, however, is not to invent yet another landmark-based routing protocol. Rather, our key motivation is to tackle the inefficiencies of LR in WSNs using the design principles outlined above. Thus, XLR's methodologies can be applied to most landmark-based routing protocols by directly incorporating XLR's components, piecewise.

The primary contributions of this paper are as follows.

1. We provide a new, flexible and comprehensive framework, XLR, to tackle the inefficiency of landmark-based routing in a piecewise manner grounded by our three guiding principles:

algorithmic simplicity, update efficiency, and application awareness, in large wireless sensor networks.

2. XLR consists of four components that facilitates efficient landmark-based routing.
   a) A simple, yet efficient, relay selection (RS) algorithm that effectively adjusts in dynamic environments.
   b) PPN, a new distance function to efficiently compute distances.
   c) A forwarding table update scheme, EUCD, that reduces the required update overheads.
   d) An application-aware forwarding scheme, GF, that closely couples application requirements with LR properties.
3. A comprehensive evaluation of the above through extensive simulations.

The rest of paper is organized as follows. Related work are presented in section II. Then, we address the framework XLR in section III. In section IV, we evaluate the framework XLR. Finally, we conclude in section V.

II. Related Work

Landmark-based routing is derived from geographic routing [14] and virtual coordinate-based routing [10]. Geographic routing protocols, e.g., GPSR [14], were proposed to perform scalable routing using true positions from GPS or localization techniques. However, it is not practical to equip each WSN nodes with expensive GPS technology. Inaccurate localization techniques also largely degrades geographic routing performance. Virtual coordinate based routing, e.g. NoGeo [10], performs geographic routing based on virtual coordinates that are computed from certain algorithm [10]. However, it is complicated to compute and maintain these virtual coordinates [2].

Landmark-based routing protocols, e.g. BVR [2], HopID [3], LCR [4] build the coordinate system using a set of landmark nodes selected by techniques such as random selection [2], [3] and iterative selection [4]. Though the algorithmic method improves landmark-based routing performance, it comes at the cost of increased packet flooding of the network. Various distance functions, such as Weighted Manhattan [2], Euclidean [4], and Minkowski function [3] are used to compute the distance for greedy forwarding. GLIDER [5] configures landmarks manually to obtain a global shortest path topology structure and uses both local greedy routing and global shortest path routing. In S4 [7], compact routing is applied into landmark-based routing. Each node maintains an optimal distance vector routing table for its local cluster of nodes, which could incur considerable cost in large wireless sensor networks; even worse when there is a small number of landmarks. GLDR [6] uses r-Sampling to improve landmark selection. However, it is difficult to determine the number of landmarks in advance and control or manage the number of necessary landmarks. In [15], AVCS is proposed to improve the success of greedy forwarding. The coordinate of each node is averaged together with the coordinates of its neighbor nodes in order to make the coordinate a smoothed Euclidian space. Similar to virtual coordinate based routing, coordinate update control traffic can be significant. In geographical routing, either a product of distance and link quality [8] or a normalized metric [9] is used in forwarding, which is found to be optimal. A more detailed related work can be found in our technical report [16].

III. The XLR Framework

In the framework XLR, we seek to address our three principles in turn.

Algorithm 1 Relay Selection

Require: K randomly distributed landmark candidates.

procedure INITIALIZE LANDMARKS(K,β, Baton=Null)
  if $Baton == Null$ and $LM_{id} == 1$ **then** ▷ LM_{id}: the ID of landmark
    $Baton \leftarrow 1$
    FLD($LM_{id} = 1$) ▷ Construct shortest path tree rooted at $LM_{id} = 1$.
  end if
end procedure
procedure DUTY(Baton) ▷ Handles Batons upon receipt.
  if $Baton = LM_{id} - 1$ **then**
    if $h_m \geq \beta, \forall m \in \{1, 2, ..., LM_{id} - 1\}$ **then** ▷ Enter into landmark status
      $Baton \leftarrow LM_{id}$
      FLD(LM_{id})
    else ▷ Current landmark is too close
      LRS($\beta, Baton, 1$)
    end if
  else if $Baton \geq LM_{id}$ **then**
    if $LM_{id} == 1$ **then** Terminate ▷ Failure to find K landmarks, end
    else
      LRS($\beta, Baton, 0$)
    end if
  end if
end procedure
procedure LRS(β, $Baton, flag$) ▷ Find a node with limited scope of flooding.
  $maxhop \leftarrow (flag = 1)?\beta - \mathrm{MIN}(\mathrm{Hop}(LM_{id})) : \beta$
  Limited range searching with radius of maxhop to find a node that $\mathrm{Hop}(ID_i, h_m) \geq \beta, \{(flag = 1)?\forall m \in \{1, 2, ..., LM_{id} - 1\} : \forall m \in \{1, 2, ..., LM_{id}\}\}$
  if Such a node is found **then**
    Transfer($ID_i, LM_{id} = Baton + 1$) ▷ Transfer "landmark" status to ID_i.
  else
    ReportBaton($Baton, LM_{id} - 1$)
  end if
end procedure

A. Algorithmic Simplicity

Relay Selection: As indicated in Sect. §II, the prior works have examined the landmark selection and coordinate system establishment problems. Though these algorithmic methods, e.g. iterative selection [4] and r-Sampling [6], improve landmark-based routing performance, it comes at the cost of either increased packet flooding of the network or complicate landmark management. In this paper, we ask: *Can we relax landmark placement while the performance of landmark-based routing is promising?* Inspired by both iterative selection and r-Sampling, we propose a new landmark selection method: Relay Selection (RS) and detailed in Algorithm 1.

Intuitively, the key to RS is to select landmarks as to avoid clustering landmarks too closely together, e.g., two hops away from each other. This process is initiated by a user designated landmark, then runs autonomously in a distributed fashion to ultimately select a set of landmark nodes. In combination with the distance function presented in the following section, we obtain promising results (shown in Section §IV).

First, let us introduce some notations. Let K be the desired number of landmarks. A constant β is the minimum hops between the landmarks. Batons are embedded into flooding packets and act as a token during the landmark selection process. N is the total number of nodes in the network. $\texttt{Hop}(ID_i, h_1, h_2, .., h_j)$ is the coordinate of each node, where $i \in \{1, 2, ..., N\}$ and $j \in \{1, 2, ..., K\}$, ID_i is the ID number of node i, and h_j is the hop distance of node ID_i to the j-th landmark. `FLD()` is a flooding function that establishes the shortest path tree from the initiating node to every other node in the network. `MIN()` is used to return the minimum element of one coordinate vector. `Transfer()` lets the new found node be a landmark. If a new landmark can not be found, `ReportBaton()` reports failure and returns the baton to the previous landmark[2].

The New Distance Function: Next, we consider how to measure the distance between two nodes. This plays a vital role in determining the performance of routing. In a continuous Cartesian space, the coordinates are continuous and smooth. The distance between any pair of nodes can be determined accurately by the traditional distance function, i.e., Euclidean distance. However, in landmark-based routing, the coordinate of a node is represented by a vector that is composed of hop counts (integral values) to each landmark. It is non-trivial to measure the distance between two nodes in this situation. The discrete values of the hop count coordinate affects the accuracy of the coordinates by introducing the "noise"[15]. Furthermore, the coordinate is a vector with high dimensionality. Traditional distance functions, e.g., Manhattan, Euclidean and Minkowski distance, cannot accurately and effectively compute the distance.

In the following, our goal is to develop an effective distance function, which is suitable for landmark-based routing. Given a destination node $D(d_1, d_2, ..., d_k)$ and a forwarder's neighbor node $F(n_1, n_2, ..., n_k)$, where d_k and f_k are the hop distances to the K^{th} landmark.

We propose a new distance function: Parametric-P-Norm distance function (PPN):

$$PPN(\overline{F}, \overline{D}, \lambda_1, ..\lambda_k) = (\textstyle\sum_{i=1}^{k} \lambda_i |\overline{f}_i - \overline{d}_i|^{p_2})^{1/p_1}$$

where λ is the coefficient of weight for each dimension in the coordinate, P_1 and P_2 are two parameters, $\overline{F}$ and $\overline{D}$ are modified coordinate vectors, discussed below.

Next, we will introduce the interpretable issues about PPN. Since the coordinates are integral values, e.g., hop counts, they cannot accurately represent the real distance among the nodes in the network. There are two options to remove the noise from coordinates: *averaging* [15] and *transformation* [5]. Briefly, averaging the coordinate of one node depends on both itself and the coordinates of its n-hop neighbors, e.g. one hop or two hops. It reaches the equilibrium state by averaging the forces from its neighbor nodes [10]. However, it is similar to coordinate calculation in virtual coordinate based routing, thus averaging breaks the relative independence of coordinates and coordinates are extremely sensitive to wireless dynamics, such as channel uncertainty. Furthermore, averaging may introduce additional update cost, as we discuss in Sect. §II.

Considering these limitations, we opt for transformation. For a node F($f_1, f_2, ..., f_k$), we firstly calculate a mean value $\sum_{i=1}^{k} f_i^j/k$ of F, where the depth of j=1,2,...,n. Then, the modified coordinate of node F is $\overline{F} = (f_1^j - \sum_{i=1}^{k} f_i^j/k, ..., f_k^j - \sum_{i=1}^{k} f_i^j/k)$. Note that, in Sect §IV, we find that PPN works very well as j=2, which is also proven in [5].

The success ratio of greedy forwarding always increases when the packet is forwarded towards landmarks closer to the destination node. These landmarks are also called *routing* landmarks. This could not only reduce the interference from irrelevant dimensions, but also reduce the computation cost of the distance function. For example, we can put more weights on the dimensions that have positive values of difference $(\overline{f}_i - \overline{d}_i)$. In our simulation, we only consider 10 landmarks that are closest to the destination node and set λ_i=1 when the ith landmark is one of those closest landmarks. Otherwise, we set λ_i=0.

In the PPN, the difference formula $|\overline{f}_i - \overline{d}_i|$ dominates the computation. Note that when p_2 is fixed, different values of p_1 do not affect the results from PPN. Therefore, we set p_1=1 in our paper. When we increase p_2, if the difference in one dimension is bigger, the distance function will reflect more difference in that dimension.

[2]Node synchronization issues are handled through timers, not explicitly detailed in our algorithm.

B. *Opting for Update Efficiency*

In this paper, we construct a coordinate system using the shortest path trees rooted in landmarks as previous efforts did [4], [2], [3], [7]. Link quality is used to assure that the shortest path trees do not traverse long and unreliable links (detail in subsection C). In order to eliminate count-to-infinity problems and loops, each node maintains the highest sequence number and a parent node of the tree to each landmark. We assume these landmarks are stable, and only non-landmark nodes and links may fail as previous efforts did [4], [2], [3], [7]. In addition, each node only needs to maintain its direct neighbors (one hop away).

Due to the dynamic wireless environment, the WSN's coordinate system undergoes constant change. Various control messages are required to maintain coordinate system consistency and availability between nodes and the location service. In this paper, we focus on two types of control traffic: *periodic update traffic* and *passive update traffic*. Periodic update traffic includes control traffic exchanged between a node, its neighbors and the location service at regular intervals that is necessary to maintain and update node coordinates, the coordinate system and the location service[13]. During the interval, passive update traffic is generated by link fluctuations, link and/or node failures. Passive updates include both neighbor discovery broadcast and location service update messages, generated due to a coordinated change.

To achieve our *update efficiency* design principle, we seek to minimize the amount of control traffic exchanged to maintain system coordination. We propose a new method: Efficient Update with Coordinate Difference (EUCD), which lowers packet size by generating update payload data from the difference between coordinate changes. Let K be the number of landmarks in the network and h_{ik} is the hop count from node i to landmark k. Node i's coordinate is a vector $C_i = \{h_{ik}\}$, where $|C_i| = K$. Now, $\Delta_i = (\texttt{INC}, \texttt{DEC})$ is a 2-tuple of increment and decrement vectors, denoting the changes in subsequent coordinates of node i. Furthermore, we define two special functions, $|\Delta_i|$ and $||\Delta_i||$, where $|\Delta_i|$ returns the maximum element in Δ_i and $||\Delta_i||$ that returns the number of non-zero elements in Δ_i. For example, say node i's coordinate updates from $C_i = \{5, 2, 1\}$ to $C_i = \{4, 2, 2\}$, then $\Delta_i = (\{0, 0, 1\}, \{1, 0, 0\})$, $|\Delta_i| = 1$ and $||\Delta_i|| = 2$. Now, let us consider three possible scenarios for the value of $|\Delta_i|$ and the subsequent update messages necessary:

Case 1: $|\Delta_i| = 0$. If there are no changes to the coordinate, a node simply sends an special keep alive update packet to its neighbors. Additionally, no update packet is sent to the location service. As no changes are necessary, the keep alive packet need not encode any payload data.

Case 2: $|\Delta_i| = 1$. In this case, EUCD uses `INC` and `DEC` to indicate the hop differences between the `old` and `new` coordinate and announces $\texttt{UP}(\texttt{INC}, \texttt{DEC})$ to the node's neighborhood. In this case, `INC` and `DEC` are binary vectors at least K bits long, where K is the number of landmarks. As a result, at most $2 \cdot \lceil \frac{K}{8} \rceil$ bytes are necessary to encode the update message payload data.

Case 3: $|\Delta_i| > 1$. Here, we further consider two subcases. First, if $||\Delta_i|| < K/2$, EUCD only announces the changed dimensions of coordinates `UP(CD,CC)`, where `CD` and `CC` represent the changed dimensions and the changed coordinates, respectively. For example, say C_i changes from $\{3, 4, 5, 6, 7, 8, 9, 10\}$ to $\{3, 6, 8, 6, 7, 8, 9, 3\}$. Here, $\Delta_i = (\{0, 2, 3, 0, 0, 0, 0, 0\}, \{0, 0, 0, 0, 0, 0, 0, 7\})$, $|\Delta_i| = 7$ and $||\Delta_i|| = 3$. We set `CD`=(1,2,7) and `CC`=(6,8,3) and send `UP(CD,CC)=UP((1,2,7),(6,8,3))`. The resulting update packets encode between 2 and K bytes of data. Secondly, if $||\Delta_i|| > K/2$, EUCD would announce the new coordinates as done in previous approaches [4], [2], [3], [7]. In these methods, the update packets encode K bytes as payload data.

Unlike previous approaches that cope with link uncertainty, EUCD will not immediately send update packets to its neighbors or the location service databases upon the disappearance of a link or node. Rather, since EUCD does not distinguish a permanent link outage from a temporary outage, EUCD first locates a new parent node from among its the impacted node's neighbors, during which the node keeps its coordinates unchanged. If a new parent node is located, the node's coordinates remain unchanged and only the node's parent is updated. Otherwise, the node initializes a timer to keep this coordinate unchanged, disregarding periodic updates, until timer expires.

C. Catering to Application Awareness

In this section, we investigate an application-oriented forwarding model that integrates realistic factors, such as link quality, delay, and residual energy.

In landmark-based routing, link quality is the primary factor in the success of any forwarding strategy: In geographic routing, the forwarding formula is *Distance* $\times$ *Factor* (found to be optimal in [8], [9]), where factor may be link quality, delay, or energy consumption and the policy is to maximize it. However, it can not be applied into the landmark-based routing directly, and should be coupled with the unique properties of landmark-based routing. As discussed in previous subsection B, using link quality as the primary metric in constructing the coordinate system results in better performance. Then, the hop count coordinates are derived from those trees. Distance (in hops) in landmark-based routing is obtained from a threshold based link quality scheme, not always the best link [17], [2], [7]. Furthermore, we do not consider the factors independently as done in previous efforts. We would also like to explore their inter-dependent relationships. In addition, it is worth noting that for both delay and energy consumption, retransmission plays an important role. Therefore, link quality should be considered as the prime factor in forwarding strategy.

In the following, we will present the specifications about how to implement link quality, delay and energy consumption. A passive link estimator layer [17], [2], [7] is adopted to measure these factors.

We use ETX [17] to estimate link quality. In brief, the passive link estimator uses ongoing packets from neighbor nodes to estimate the properties of incoming links.Tagging the packets with a sequence number, the node could check the sequence numbers in the packets and compute the ratio of received packets to sent packets during an successive time interval. In order to estimate the outgoing links, each node could periodically broadcast beacons to its neighbors, which contains the qualities of incoming links.

It is apparent that a node with a good link is involved in more communications. The result is that this node could be drained out quickly, which could cause the partition of the network. In our paper, we take the residual energy of the next hop nodes into account. The forwarding strategy favors the node who has higher residual energy. In our model, $C_{energy} = C_{residual-energy}$. We use the beacons from the passive link estimator for each node to announce its residual energy to its neighbor nodes.

We only consider transmission delay, from the current node to the next hop, and experienced delay, the interval from receiving the packet to sending the packet to the next hop node. Namely, $C_{delay} = C_{experience} + C_{tx}$. Experienced delay is broadcasted by the beacons used in passive link estimator. Only when the node is processing packets, the experienced delay should be updated.

From the statement above, we derive our General Forwarding (GF):

$$GF = Distance \times LinkQuality \times (\mu Energy + \rho Delay)$$

where $LinkQuality = \frac{1}{ETX}$, $Energy = C_{energy}$ and $Delay = \frac{1}{C_{delay}}$. $\mu\ and\ \rho$ are the coefficients. Distance is the distance between forwarder and one of its neighbor nodes. LinkQuality, Energy and Delay are nomalized into the same scale. The major policy is to *maximize* the results from GF.

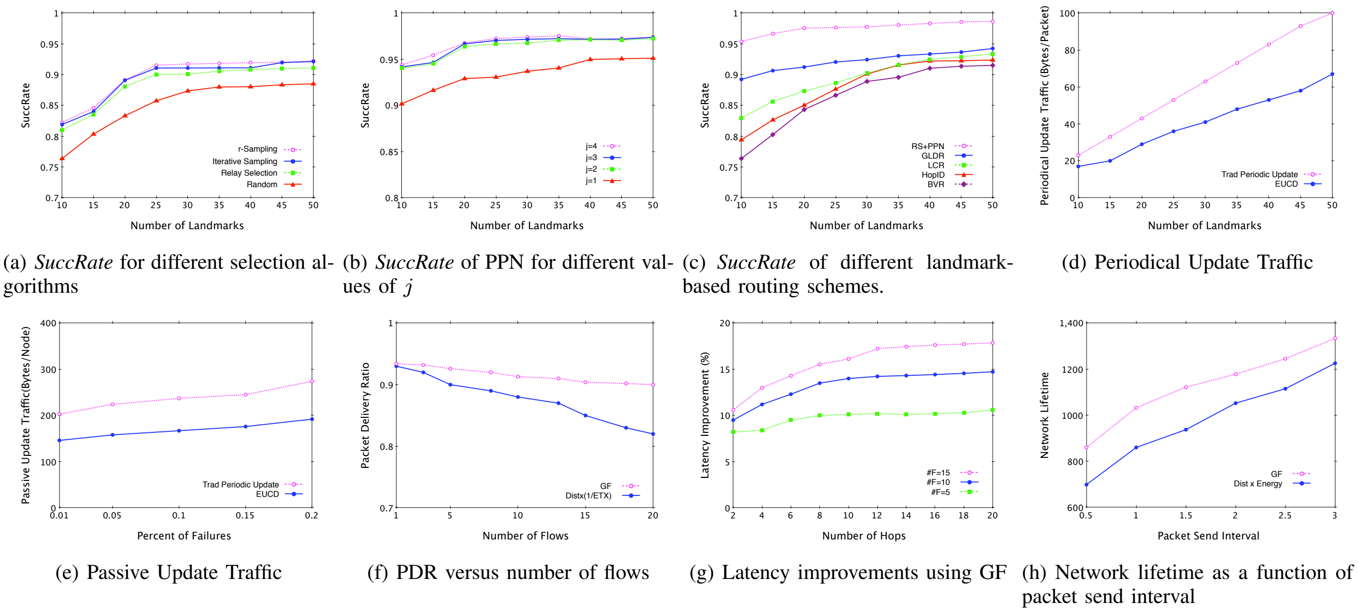
IV. Performance Evaluation

In this section, we present a simulation-based evaluation of XLR. We begin this evaluation using TOSSIM [18], a TinyOS-based discrete time simulator that captures many realistic radio properties, yet does not scale adequately in networks with more than 1000 nodes. To address this scalability limitation, we developed a custom C++-based simulator. Our simulator does not account for realistic radio properties, so we use this simulator to focus on verifying correctness and scalability issues of XLR. We evaluate each metric by averaging over 10 simulation executions.

We consider the following performance metrics:

- Routing Success Rate (SuccRate): Success rate is tested purely from a routing perspective and any recovery schemes are not used when packets hit a dead end or local minima that leads to a failure.
- Periodical Update Traffic: The amount of traffic generated by periodic beacon packets and the update packets to location service if the coordinate is changed. We assume that each node would send the location service update packet to its closest landmark.
- Passive Update Traffic: This traffic is generated between two consecutive routing updates due to external factors, such as node or link failures and link fluctuation.
- Packet Delivery Ratio (PDR): The ratio of the total number of packets received by destination nodes to the total number of packets sent by source nodes.
- Latency: The latency is defined as the average amount of time between the start of sending a packet and its arrival at the destination node. Transmission time is based on the CC2420 transceiver data transfer rate 250Kbps and the size of the packet 40 bytes.
- Network Lifetime: We define the lifetime of the network as the simulation time at which the first outage in the network occurs.

Custom Simulator Settings: We generate 4000 nodes placed uniformly at random within a 250×250 square units. The radio range is set to 8 units. The default number of landmarks is 10. The average node degree is approximately 12. In this simulator, as previous works



(a) *SuccRate* for different selection algorithms

(b) *SuccRate* of PPN for different values of j

(c) *SuccRate* of different landmark-based routing schemes.

(d) Periodical Update Traffic

(e) Passive Update Traffic

(f) PDR versus number of flows

(g) Latency improvements using GF

(h) Network lifetime as a function of packet send interval

Fig. 2. Simulation Results

Algorithm	Distributed	Feasibility	Overhead (#floodings)
Random	$\surd$	$\times$	$O(K)$
r-Sampling	$\surd$	$\times$	$O(m), m \gg K$
Iterative Selection	$\surd$	$\surd$	$O(K^2)$
Relay Selection	$\surd$	$\surd$	$O(K)$

TABLE I

COMPARISONS OF DIFFERENT SELECTION ALGORITHMS, K IS THE NUMBER OF LANDMARKS

did [2], [7], [4], [6], [5], we test the following metrics: routing success rate, periodic update traffic and passive update traffic

TOSSIM Settings: We inject n concurrent flows into the network. We uniformly at random choose the source and destination for each flow. We generate 1000 nodes and place them uniformly at random in the area of 200×200 square meters and 30 landmarks are used in the simulation. Lossy links are generated by a widely used default radio model [19]. We use TOSSIM to evaluate metrics impacted by radio dynamics, such as packet delivery ratio, latency and network lifetime.

A. Simulation Results

We evaluate all the components of our XLR framework for landmark-based routing using the above metrics.

1. Relay selection (RS): First, to evaluate relay selection, we compare our selection algorithm (RS) with r-sampling, iterative selection, and random selection from the perspective of routing success rate. For the simulation results in Fig. 2(a), RS's β parameter is set to 5. Though not detailed in this paper, we evaluated RS's performance given β and found that as we increase β, RS performs better. Figure 2(a) shows that the routing success rates with iterative selection, r-Sampling, and relay selection are significantly better than random selection. The reason is that random selection can not avoid the worst case of landmark placement. Iterative selection and r-Sampling yield comparable performance, but both perform slightly better than our relay selection algorithm. However, the iterative algorithm requires prohibitively costly computation to locate the landmarks and obtain the coordinate for each node, as shown in Tab. I. Alternatively, though r-Sampling reduces the need for complex computation, it is difficult to control the exact number of landmarks, especially in lossy networks. As shown in Tab. I and Fig. 2(a), RS is significantly simpler in comparison, and yet, achieves promising routing performance.

2. Distance function: Next, we evaluate the performance of PPN for different values of the parameter j. In Fig. 2(b), we see that the success ratio of PPN, when j=2, is higher than that of PPN when j=1. This is because PPN have more effects to remove "noise" from coordinates when parameter j=2. When we further increase the value of j, the success ratio only increases slightly. Based on these results, we use j=2 in the rest of our evaluation.

3. Combining relay selection and distance function: We next evaluate routing performance after we combine relay selection and PPN; we refer to this scheme as RS+PPN. We compare RS+PPN with previous routing protocols such as BVR, LCR, HopID, and GLDR [2], [4], [3], [6]. We do not consider GLIDER and S4 for the SuccRate evaluation, since routing in these cases is aided significantly by either global shortest path structure from the manual configuration of landmarks or optimal distance vector routing for each node. In Fig. 2(c), we see that the routing performance of RS+PPN outperforms previous approaches. Although RS slightly degrades routing performance, overall routing performance is boosted significantly by the PPN.

4. Update traffic: As discussed before, the traffic of update includes periodical update traffic and passive update traffic. We refer to the scheme of periodical update used in previous routing protocols such as BVR, LCR, HopID, GLDR, GLIDER, and S4 as Traditional Periodical Update. In Fig. 2(d), we observe that EUCD significantly reduces the periodical update traffic comparing with traditional update. When the number of landmarks is the range 10–50, EUCD can eliminate 23-39% of the periodic update traffic compared with traditional update techniques. EUCD reduces update traffic by propagating only the differences of coordinates instead of announcing all coordinates. We randomly selection non-landmark nodes and links to fail or links to fluctuate by the percentage of total number of nodes and links. The passive update traffic is generated between two consecutive routing updates. The traditional method for a node to update its coordinate in such scenarios is to immediately broadcast request packets and update its coordinate based on the replies received

from its neighbor nodes. If the coordinate is changed, the node sends an update packet to its location service. Fig. 2(e) shows that EUCD generates much less traffic than prior approaches due to hysteresis from the Timer setting. These results show that EUCD not only reduces the overhead of update traffic, but also copes better with network dynamics.

5. Packet delivery ratio: We compare the packet delivery ratio obtained with GF with that obtained using Dist$\times \frac{1}{ETX}$, especially in high-traffic scenarios. In this simulation, we set $\mu = 0$ and $\rho = 1$. (Note that we do not intend to optimize the parameters μ and ρ in our paper, and we leave it as future work.) In Fig. 2(f), packet delivery ratios with both schemes decrease with increasing number of flows, but the decrease with Dist$\times\frac{1}{ETX}$ is more marked. This is because Dist$\times\frac{1}{ETX}$ only considers routes with high link quality, but the high quality links involve more communications that cause network contention and hotspot in the network. GF not only considers link quality but also considers delays that could indicate contention. GF lets traffic detour hotspot and avoid collisions and packet losses. In summary, GF achieves a better packet delivery ratio.

6. End-to-end latency: We next compare the average end-to-end latency yielded by GF with the corresponding value using Dist$\times\frac{1}{C_{delay}}$. Here, we use $\mu = 0.5$ and $\rho = 1$. In Fig. 2(g), we plot the percentage improvement in latency with GF in comparison with that obtained with Dist$\times \frac{1}{C_{delay}}$, as a function of the average number of hops per data flow. Note that Dist (in hops) in landmark-based routing is obtained from a threshold based link quality scheme, not always the best link. If only considering the factor of delay, the forwarding function may choose lower quality links that could invoke more retransmissions. Moreover, GF works better when the flow is longer. Longer flows have a higher probability of passing through hotspot in the network. However, GF can detour the hotspot with the help of residual energy of the next hop nodes. Because the nodes which involve more communications in hotspot have less residual energy, GF favors the nodes that have higher residual energy to avoid such contention and congestion. Therefore, GF yields significant latency benefits, for different values of the number of flows.

7. Network lifetime: Next, we compare network lifetime with GF with that using Dist$\times C_{energy}$, when $\mu = 1$ and $\rho = 0$. In Fig. 2(h), we see GF increases the lifetime (simulation time) of the network beyond that obtained when considering only energy. The underlying reason is that Dist$\times C_{energy}$ does not guarantee links that are always high quality as discussed before. Poor quality links invoke more retransmissions that results in higher energy consumption. This in turn decreases network lifetime.

Remark: The key point to take away from our simulations comparing GF with the previous approach is that in landmark-based routing, link quality is the key factor not only for a stable coordinate system, but also for designing forwarding strategies.

V. Conclusion

Efficiency is a pressing concern to landmark-based routing in large wireless sensor networks. In this paper, we present *XLR*, a new, flexible and comprehensive framework that realizes our three design principles necessary for efficient landmark-based routing: algorithmic simplicity, update efficiency, and application awareness. Our framework *XLR* is composed of four components: relay selection (RS) to choose landmarks, parametric-p-norm distance function (PPN) to compute distances, efficient update with coordinate difference (EUCD), and a systematic model of forwarding (GF). We have evaluated *XLR* with extensive simulations and our results show that our framework yields significant performance benefits.

Acknowledgements

This material is based upon work supported by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053 and the National Science Foundation under Grant No. NETS-0721889. Any opinions, findings, and conclusions or recommendations in this material are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory, the U.S. Government, the National Science Foundation, or other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] J. P. Vasseur and D. Culler, “Routing over Low Power and Lossy Networks.” [Online]. Available: http://tools.ietf.org/wg/roll

[2] R. Fonseca, 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 *USENIX NSDI*, 2005, pp. 329–342.

[3] Y. Zhao, Y. Chen, B. Li, and Q. Zhang, “Hop ID: A Virtual Coordinate-Based Routing for Sparse Mobile Ad Hoc Networks,” *IEEE Trans on Mobile Computing*, pp. 1075–1089, 2007.

[4] Q. Cao and T. Abdelzaher, “Scalable Logical Coordinates Framework for Routing in Wireless Sensor Networks,” *ACM Trans. Sen. Netw.*, pp. 557–593, 2006.

[5] Q. Fang, J. Gao, L. Guibas, V. de Silva, and L. Zhang, “GLIDER: Gradient Landmark-based Distributed Routing for Sensor Networks,” in *IEEE INFOCOM*, 2005, pp. 339–350.

[6] A. Nguyen, N. Milosavljevic, Q. Fang, J. Gao, and L. Guibas, “Landmark selection and greedy landmark-descent routing for sensor networks,” in *IEEE INFOCOM*, 2007, pp. 661 –669.

[7] Y. Mao, F. Wang, L. Qiu, S. Lam, and J. Smith, “S4: Small State and Small Stretch Compact Routing Protocol for Large Static Wireless Networks,” *IEEE/ACM Transactions on Networking*, pp. 761 –774, 2010.

[8] M. Z. n. Zamalloa, K. Seada, B. Krishnamachari, and A. Helmy, “Efficient Geographic Routing over Lossy Links in Wireless Sensor Networks,” *ACM Trans. Sen. Netw.*, pp. 12:1–12:33, 2008.

[9] S. Lee, B. Bhattacharjee, S. Banerjee, and B. Han, “A General Framework for Efficient Geographic Routing in Wireless Networks,” *Computer Networks*, vol. 54, no. 5, pp. 844 – 861, 2010.

[10] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, “Geographic Routing without Location Information,” in *ACM MobiCom*, 2003, pp. 96–108.

[11] N. James and D. Song, “GEM: Graph EMbedding for Routing and Data-Centric Storage in Sensor Networks without Geographic Information,” in *ACM SenSys*, 2003, pp. 76–88.

[12] P. F. Tsuchiya, “The Landmark Hierarchy: a New Hierarchy for Routing in Very Large Networks,” *SIGCOMM Comput. Commun. Rev.*, pp. 35–42, 1988.

[13] J. Ortiz, C. R. Baker, D. Moon, R. Fonseca, and I. Stoica, “Beacon Location Service: A Location Service for Point-to-Point Routing in Wireless Sensor Networks,” in *ACM IPSN*, 2007, pp. 166–175.

[14] B. Karp and H. T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” in *ACM MobiCom*, 2000, pp. 243–254.

[15] K. Liu and N. Abu-Ghazaleh, “Aligned Virtual Coordinates for Greedy Routing in WSNs,” in *IEEE MASS*, 2006, pp. 377 –386.

[16] X. Wei, N. C. Valler, M. Faloutsos, H. V. Madhyastha, and T.-K. Huang, “XLR: Tackling the Inefficiency of Landmark-based Routing in Large Wireless Sensor Networks (Extended Version),” University of California Riverside, Tech. Rep. UCR-CS-2010-12100, 2010. [Online]. Available: http://www1.cs.ucr.edu/research/publications/

[17] A. Woo, T. Tong, and D. Culler, “Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks,” in *ACM SenSys*, 2003, pp. 14–27.

[18] P. Levis, N. Lee, M. Welsh, and D. Culler, “TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications,” in *ACM SenSys '03*, 2003, pp. 126–137.

[19] M. Z. n. Zamalloa and B. Krishnamachari, “An analysis of unreliability and asymmetry in low-power wireless links,” *ACM Trans. Sen. Netw.*, vol. 3, June 2007.