
©2008 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.
Abstract

One of the key challenges in geographic routing is how to deal with dead-ends, where greedy routing fails to find a neighbor node which is closer to the destination. Most existing geographic routing algorithms just switch to the deterministic face routing or limits its face searching range. In this paper, we demonstrate that we can improve routing performance by considering local connectivity status at each node before making routing decision. We present a protocol, Density Ripple Exchange (DRE), that maintains local density information at each node, and a new geographic routing algorithm, Geographic Ripple Routing (GRR), that achieves better routing performance in both hop stretch and transmission stretch than existing geographic routing algorithms by exploiting available connectivity information. Our simulations demonstrate that we increased the performance for GRR over Greedy Perimeter Stateless Routing (GPSR) by about 15%. The cost of this improved performance is a small amount of additional local connectivity information required for our algorithm.

1. Introduction

Ad-hoc routing protocols used today, such as DSDV [12], OLSR [3], AODV [13], and DSR [7], only scale reasonably to dozens or hundreds of nodes. Geographical routing protocols [8, 2] make use of the geographical location of a node to make routing decisions and do not require the establishment or maintenance of routes. Those characteristics eliminate the overhead of frequent topology updates and route acquisitions which are required by DSDV, AODV and DSR. For these reasons, Geographic routing protocols are attractive compared to the traditional ad hoc routing algorithms [12, 3, 13, 7] due to their scalability and robustness to changes in the network topology. Currently the geographic routing protocols are preferred choice for large and highly dynamic networks.

In geographic routing protocols, the nodes’ geographical positions are used to make routing decisions. The location information could be acquired either from GPS satellites. Each node forwards packets greedily [4] that is to find a neighbor node which is closest to the destination. This process is repeated until the packet reaches the destination. If a node does not have any neighbor closer to the destination, it switches to the face routing [9] through which the packet is routed along the faces of a extracted planar subgraph by applying algorithms, the Gabriel Graph (GG) [5] or Relative Neighborhood Graph (RNG) [16]. Karp and Kung [8], and Bose et al. [2] proposed the idea of combining the greedy forwarding and face routing on a planar graph. When a packet gets stuck at a node, it is routed by the “right-hand rule” along a face until it reaches a node that is closer to the destination. At this point, the packet returns to greedy forwarding phase.

In geographic routing protocols, as a node only knows its immediate neighbors, there is often insufficient information for it to make a good decision on the forwarding direction. When a packet gets stuck at a node in which no neighbors are closer to the destination, Greedy Perimeter Stateless Routing (GPSR) [8] just arbitrary chooses a direction, e.g., “right-hand rule”, to forward the packet along the face. But this choice could be the wrong one. Greedy Other Adaptive Face Routing (GOAFR+) [10] deals with that problem by bounding the search in each direction within an expanding ellipse to avoid the full wrong choice.

This paper proposes a different way of tackling the problem. Each node will consider not only its neighbor’s geographic location but also their local connectivity information to make its routing decision. We believe that the proximity of two nodes is based on two factors: geographic location and connectivity status. We developed a new routing algorithm, Geographic Ripple Routing (GRR), which
collects local connectivity and geographic information and assigns them with different weight to calculate the next forwarding node. Thus the algorithm still retains scalable and distributed benefits which are important to apply to a large wireless sensor network.

The contributions of our work are as follows:

- We show and analyze that an optimal routing is tied with geographic location and connectivity;
- We propose a practical distributed algorithm, Geographic Ripple Routing (GRR) that computes the forwarding node based on geographic location and its connectivity information;
- Through the simulations we evaluate the performance of GRR and show that it achieves significantly better performance in terms of both hop stretch and transmission stretch than GPSR.

The remainder of this paper is organized as follows. Section 2, we provide a review of existing and related work. Followed that, the key contribution, Geographic Ripple Routing, is analyzed and presented in Section 3. In Section 4, the performance of the algorithm is evaluated. We conclude with a few remarks and the implication of our future work in Section 5.

2. Related work

Finn [4] proposed the early geographic routing concept which was a simple greedy forwarding scheme. The idea is to find a neighbor node which is closest to the destination node. This process is repeated until the packet reaches the destination. If a node does not have any neighbor closer to the destination, the packet will be dropped. So the scheme did not have any guarantees of packet delivery in a connected network. The first geographic routing algorithm to provide guaranteed delivery was Face Routing [9].

Karp and Kung proposed Greedy Perimeter Stateless Routing (GPSR) [8] and Bose et al. proposed Greedy Other Adaptive Face Routing (GOAFR+) [2]. Both were the combination of the greedy forwarding and face routing. These algorithms also provided delivery guarantees and were more efficient than Face Routing. However, their proposals are likely to yield a wrong forwarding decision when packets gets trapped in a local minimum and lack a mechanism to adapt and learn from the experience. In such scenarios, the performance of GPSR/GOAFR+ may degrade severely as greedy routing fails and a recovery mechanism has to be applied, where packets are forwarded according to the face routing algorithm. The followed path may then be very suboptimal as shown in an example in Figure 1.

Kuhn et al. proposed an algorithm called Adaptive Face Routing (AFR) [2] that bounds the searching area in each direction during routing. Their algorithm also achieves the optimal worst-case result. Kuhn et al. studied the performance of a family of geographic routing algorithms that combined Greedy forwarding and AFR in different ways. A clustering technique was applied to GOAFR to produce GOAFR+ [2]. But the proposal just may avoid the full consequences of a wrong forwarding decision.

Stojmenovic and Lin proposed extending existing geographic routing schemes to two-hop neighborhoods [14]. The proposal also used greedy forwarding and two-hop neighborhood information to reduce the chances of routing queries to dead-ends. The idea motivated our work. The difference between our work and the two-hop neighborhoods is that GRR uses the density information to decide the “rippling” range rather than the sole two-hop which was used in [14].

There have been some other complex geographic routing algorithms proposed which are not based on the concept of face routing. In EASE [6], each node caches all positions of previous nodes and associates a time-stamp with these positions. A node consults its cache to obtain estimates of the destination’s current location. In GRA [15], each node does not only have knowledge of its neighbors, but also stores the positions of all other nodes it is aware of, together with the next hop to reach these nodes. In DREAM [1], it includes a location service in order to determine the position of destination. Each node proactively disseminates its location through the network.

All above described geographic routing algorithms route packets based solely on geographic location without considering nodes’ connectivity status which is addressed in our paper.
3. Geographic Routing with Connectivity Ripping Information

Current geographic routing protocols rely on the assumption that the geographic distance between two nodes coincides with their proximity in the network topology. In Figure 1 where this is not true. The resulting path from geographic routing is much longer than the optimal path because the geographic distance is only sole factor in determining next hop in geographic routing.

We believe that the proximity of two nodes is based on two factors: geographic distance and connectivity status. In Figure 1 the sole geographic distance fails to determine the best next forwarding node without considering the connectivity information. The consequence of ignoring “connectivity” is that may miss an opportunity to bypass the void area in advance. Based on the observation and study of existing geographic routing algorithms, we try to:

- To extract local connectivity information in a distributed manner;
- To design a proximity function in which the weighted sum of geographic distance with respect of connectivity effect to the routing decision;
- To develop a new geographic routing algorithm.

We developed three additional modules to implement the above goals: (a) Density Ripping protocol which uses ripple function to disseminate connectivity status; (b) proximity function which allocates geographic distance and connectivity status with appropriate weight factors to compute the next forwarding node; and (c) a new geographic routing algorithm.

Below, we describe functions and components in details. Also, we provide pseudo-code for the routing algorithm.

3.1. Ripple function: connectivity status disseminating

We have observed the following connectivity status in sensor network. First, nodes close to the void area or on the edge of the void area have less neighbor density. Second, nodes isolated with their neighbors have less neighbor density. The nodes in the above two scenarios are being treated the same with other nodes if they have same geographic distance to the destination in the traditional geographic routing. But they may result into different situations. The nodes with low neighbor density likely lead to stuck position. In contrast, the nodes with high neighbor density have more chances to find alternative path to the destination. Based on the above observation, we proposed a ripple disseminating protocol which uses a special packet, beacon, to exchange information with neighbors. In addition to the geographic information, the ripple disseminating protocol also enables nodes to exchange their connectivity information with their neighbors. On receiving the beacons from its neighbors, a node extracts the geographic and connectivity information and applies to our ripple function to decide next dissemination node or just stop the dissemination.

The general idea for ripple function is as follows: a node with less neighbor density will introduce a large ripple. Consequently, the node with high neighbor density introduces a little ripple. More formally, with the concept of “ripple effect”, nodes calculate a certain ripple power depending on their neighbors density before disseminating the “ripple” message. We assume the maximum neighbor’s density is $d_{max}$ which indicates the maximum number of neighbors a node can have. There is one parameter taken by the algorithm called $P_{Max}$ which indicates the maximum ripple power a node can generate. With the concept of “ripple” function, nodes calculate a certain ripple power depending on their density and disseminate to their neighbors. A node collects the number of its neighbors as its local density or connectivity parameter, denoted as $d$.

$$P_{ripple} = P_{max} \frac{d_{max} - d}{d_{max}} \quad (1)$$

$$P_{ripple} = P_{max} e^{-\frac{d}{d_{max}}} \quad (2)$$

We designed two different functions to calculate the “ripple” power. The first function, Equation 1, maps the ripple power linearly to the value of the node’s density. The second function, Equation 2, uses an exponential function to map the ripple power to the node’s density. The difference is that the second function which uses exponential function results in faster/exponential power changes when the node’s density changes.

3.2. Proximity Function

In this section, we present a new proximity function which takes weighted sum of geographic location and connectivity status. The function exploits the strong correlation between geographic distance and ripple power involved in computing the next routing node.

More formally, a network topology is represented as a graph $G(V,E)$, where $V$ denotes the set of network nodes and $E$ denotes the set of links. For a node $i$, each link $(i,j) \in E$ is associated with the cost of using the link. In traditional geographic routing, the cost is indicated as the distance between the two nodes. Our proximity function falls into another category which incorporates distance and local density information. In future, we would like to investigate combining with other network information, such as energy status, congestion, interference and reliability.
Our proximity function works as follows. For a given node, \( i \), which has links with its neighbors, \( \{i,j\} \in E \), where node \( j \) is one of its neighbors. Each node \( i \) is labeled with a tuple \( \{w_{ij}, j \in G_i\} \), where node \( j \) is in its neighbor set, \( G_i \). The appropriate weight factors are assigned to the Ripple Power (RP) (reflects the connectivity status) and Geographic Distance (GD) to the destination (reflects the geographic location). The node with greater weight has higher priority to be selected as next routing node. We have investigated Geographic Distance only function, Connectivity only function, Geographic Distance and Connectivity weighted sum function, and Adaptive geographic distance and connectivity status function. Geographic routing functions based on distance to destination (D) also were used in [8, 2].

**Geographic Distance only function:**
This represents the basic case where only geographic distance is only metric. In this case the weight factor, \( \alpha \), is set as 1. The path is selected based solely on distance to the destination. Intuitively, this scheme (used by GPSR, GOFAR+, etc) results in shortest distance path in a dense network.

\[
w_{ij} = \alpha \times |R - d_j| \quad \forall j \in G
\]  

(3)

\[
R = \max\{d_{ij}, \quad \forall j \in G\}
\]

(4)

where \( d_j \) represents distance from node \( j \) to the destination and \( R \) is the maximum isotropic range in the sensor field which can be set as maximum distance between any two nodes in the network.

**Connectivity only function:**

\[
w_{ij} = \begin{cases} 
-\alpha \times \log\left(\frac{p_{\min}}{p_j}\right) & \text{if } p_j \geq p_{\min} \\
1 & \text{if } p_j \leq p_{\min}
\end{cases}
\]

(5)

where \( p_{\min} \) is minimum ripple power threshold which is defined as \( p_{\min} = \min\{p_j, \forall j \in G_i\} \). \( p_{\min} \) denotes the measure of resistance a node offer for forwarding the packet in terms of connectivity status. \( \alpha \) is pre-defined constant value. The greater the ripple power, the lower is the willingness of forwarding. We use negative log in the function as a weight function. This function ignores geographic distance but translates the connectivity status to the selection priority. Also, this function is a dynamic weight function that adapts the changes with network connectivity states.

**Geographic Distance and Connectivity weighted sum function:**

\[
w_j = \begin{cases} 
\alpha \times |R - d_j| + (1 - \alpha) \times p_j & \text{if } p_j \geq p_{\min}, \alpha \leq 1 \\
|R - d_j| & \text{if } p_j \leq p_{\min}
\end{cases}
\]

(6)

In this function, we use weighted sum of geographic distance and connectivity status (ripple power) to calculate the selection priority. The node, with the highest selection priority, \( w \), is selected to be the next forwarding node. \( \alpha \) is a pre-defined weight factors associated with geographic distance and \( 1 - \alpha \) is a weight factor allocated to the connectivity status. Ripple Power. \( p_{\min} \) is the minimum ripple power threshold which is defined as \( p_{\min} = \min\{p_j, \forall j \in G_i\} \). We assume the maximum isotropic range with radius \( R \) which is defined as \( R = \max\{d_{ij}, \forall (i,j) \in G\} \). In experiments, we try to give more importance to the connectivity status by setting weight factor as 0.6. This ensures that nodes with high density have more chances to be selected.

**Adaptive Geographic Distance and Connectivity function:**
Geographic Distance and Connectivity weighted sum function relies on the pre-defined weight factors, \( \alpha \) and \( 1 - \alpha \), which associate to geographic distance and connectivity status. In a dynamic network, it is desirable to change weight factors dynamically. Adaptive Geographic Distance and Connectivity function uses the ripple power, \( \{p_j, \forall j \in G\} \), as a measure for the dynamic changes in network.

\[
w_j = \begin{cases} 
d_j \times \min\{|R - d_j|, \frac{-p_j}{1-p_j}\} & \text{if } p_j \geq p_{\min} \text{ and } p_{\max} > p_j \\
d_j \times \min\{|R - d_j|, \frac{-p_j}{1-p_j}\} & \text{if } p_j \leq p_{\min}
\end{cases}
\]

(7)

We use exponential function of this product as a weight function and try to minimize the weight when the ripple power becomes greater. \( p_{\max} \) is pre-defined maximum ripple power which can be generated in ideal situation. \( p_{\min} \) is pre-defined minimum ripple power which can be ignored in a dense situation.

**3.3 Geographic Ripple Routing (GRR)**

When greedy forwarding works, it is usually the most efficient forwarding strategy. It was reported [8] that in dense network geographic routing achieves almost optimal routing path to the destination. Geographic Ripple Routing (GRR) still follows the idea of geographic routing except using weight \( w_{ij} \) to replace sole geographic distance. The idea of GRR is that GRR tried to bypass the void area as soon as possible and keeps using greedy forwarding in the dense area. More formally, GRR is detailed in Algorithm 1.

**4. Simulation results**

In this section we evaluate the performance of Geographic Ripple Routing (GRR) through simulation-based experiments. We use simulator, ns-2 [17], to quantify the effects of our Geographic Ripple Routing strategy. We use the simulation topology is shown in Fig. 1 with nodes deployed with different density.
Our implementations of GRR are based on the algorithms as described in previous Section. We use GPSR [8] code in ns-2 [17] provided by Mr Ke Liu [11]. The configuration parameters for GRR are set as $p_{\text{max}} = 20.5, p_{\text{min}} = 0.8, \alpha = 0.34$ and $\beta = 0.56$. Other physical transmission parameters are the same with the code [11].

### 4.1 Routing Performance

We investigated the performance of the routing algorithms for the scenarios in which the sensor nodes had been deployed with different density. Source and destination are marked as $S$ and $D$ respectively in the Figure 1.

#### 4.1.1 Hop Stretch Performance

Figure 2 show the average values of hop stretch achieved by the different routing algorithms with increasing density. $GRR(R)$ denotes Geographic Ripple Routing by considering only connectivity status. $GRR(DR)$ denotes GRR by considering Geographic location and connectivity, $GRR(ADR)$ denotes GRR by using Adaptive Distance and connectivity.

The results show that GRR with geographic distance and connectivity, $GRR(DR)$, and with adaptive geographic distance and connectivity, $GRR(ADR)$, outperform GPSR. In contrast, GRR with only connectivity, $GRR(R)$, is worse than the performance of GPSR. So that it is understandable that the routing protocols which consider two factors, Geographic distance and connectivity status, can achieve better performance because the connectivity status could be used to bypass the void area in advance. By giving the different weight for distance and connectivity the node has different performance in different situation which will be investigated in our future work. It is surprised to see the GRR(DR) outperforms GRR(ADR). The likely explanation for this is careful chosen static parameters plays important role for performance.

#### 4.1.2 Transmission Stretch Performance

We also investigated the transmission stretch performance of the GRR and GPSR. The corresponding result is shown in Figure 3 which compares transmission stretch among the four routing protocols. As shown in the figure, GPSR performs best when the node number is less than 110 in all the routing algorithms. However, both GRR (DR) and GRR(ADR) outperform GPSR when node number is increasing. Only GRR(R) performs worst compared to other algorithms. One interesting observation is when the extra information is useful in helping nodes to make better decision. In our experiments, we found that on an average, GPSR performs better when nodes are less populated, e.g., less 100 nodes, in the field.
5. Summary and Conclusion

In this paper, we addressed the problem that is how to improve geographic routing performance in sensor network. We demonstrate that by using local connectivity information, we can achieve better routing performance in terms of hop stretch and transmission stretch. The extra connectivity information helps to bypass the void area.

Our paper makes two main contributions: (i) we have shown that local connectivity can help to alleviate the problem of the void area, and (ii) we developed Geographic Ripple Routing (GRR), considering local connectivity information and geographic distance. Through simulations we have shown that GRR(DR) and GRR(ADR) achieves better routing performance in terms of both hop stretch and transmission stretch than GPSR with only a small local connectivity information. In future, the adaptive algorithms will be tested to investigate the adaptive parameters setting.

References