Probabilistic Geographic Routing Protocol for Ad Hoc and Sensor Networks

Tanya Roosta
Department of EECS UC Berkeley
roosta@eecs.berkeley.edu

Mike Menzo
Department of EECS UC Berkeley
mpm@eecs.berkeley.edu

Professor Shankar Sastry
Department of EECS UC Berkeley
sastry@eecs.berkeley.edu

ABSTRACT

In this paper, we present Probabilistic Geographic Routing (PGR), a novel approach for the problem of power-aware routing in wireless ad hoc and sensor networks. Our protocol uses only local information to probabilistically forward the packet to the next hop. Every node relies on a beaconing process to keep track of the changes in the set of its neighbors. In order to forward a packet, the node selects a set of candidate nodes. These candidate nodes are then assigned a probability proportional to their residual energy and the link reliability. We have implemented PGR in NS-2 and compared the performance to two existing protocols, GPSR and Probabilistic Flooding. Based on the simulation results, PGR improves the throughput by 40%, increases the lifetime of the network by 30%, and decreases the overall end-to-end delay. In addition, we have implemented PGR on a real sensor network test-bed to verify our protocol.

1. INTRODUCTION

Ad hoc networks are infrastructureless, multi-hop wireless networks where every node can be either a host or a router, forwarding packets to other nodes in the network. Due to their potential uses in various situations such as battlefield, and emergency disaster relief, ad hoc networks have received extensive amount of attention. Wireless sensor networks represent a special type of ad hoc networks that are used for large scale distributed micro-sensing applications. The nodes in these networks are tiny devices that integrate sensors, wireless interface, and processing units in a single hardware platform which operates on a very limited battery power.

A key technical challenge in designing protocols for wireless ad hoc and sensor networks has been the problem of energy saving. Energy is a precious resource in any network with nodes that depend on a limited battery source for operation. The power dissipation in a network is due to the power consumption at different entities of the network. Therefore, it is necessary to develop power-aware schemes for different operations and at different networking layers. Recently many schemes have been proposed to improve the energy savings at the link layer, topology control, and routing layer. Our main focus in this paper is on the routing layer. Most of the routing protocols are designed with the goal of optimizing some cost along the selected routes. Examples of the cost function are the number of hops taken by a packet to reach its destination, end-to-end delay, or energy consumption.

Scalability is another important factor in designing a routing protocol for ad hoc and sensor networks. A good routing protocol has to be scalable and adaptive to the changes in the network topology. Scalability means that the protocol performs well as the network grows larger or as the workload increases. Scalability is best accomplished through decentralized algorithms, where nodes only need local information exchange to make routing decisions. It has been experimentally shown that the protocols which use geographical location of the nodes are more scalable than non-geographical protocols [12]. Geographical routing protocols make it possible for the routers to be stateless. Each node only needs to know the geographical positions of its neighbors. The position of a packet's destination is the other piece of information that is needed for selecting the next hop. In order to find the position of the nodes in geographical routing protocols, one needs to use a *location service* or *localization scheme* to learn the coordinates of the nodes.

In this paper we propose a probabilistic geographical routing protocol. We assume that each node is aware of its geographical coordinates through some localization scheme, such as GPS [18-22]. The protocol consists of two phases, the discovery phase and the routing phase which will be described in more detail later.

The rest of the paper is structured as follows. In section 2 we present the related work. In section 3 we describe PGR in more detail, and present simulation results in section 4. In section 5 we describe the implementation of the routing protocol on a real sensor test-bed, followed by discussion and conclusion in section 6.

2. RELATED WORK

2.1 Geographic Ad Hoc Routing

The appeal of geographical routing protocol lies in the fact that it is scalable, and the route selection process is localized. The node holding the message is only aware of its own location, its immediate one hop neighbors, and the destination location. There are three main forwarding strategies in position-based routing: greedy forwarding, restricted directional flooding, and hierarchical forwarding [4].

Most geographic routing protocols use greedy forwarding algorithms. Greedy algorithms send the message to the next hop which provides the most positive advancement toward the destination. One of the main problems with the greedy routing is handling voids in communication. The communication void happens when the current node is distance-wise closest to the destination than any of its neighbors, but has no direct connection to the destination to deliver the message. Handling these communication holes is where most position-based routing protocols differ. Finn has proposed one the earliest geographical

routing protocols in which restricted flooding is used to go around the void. Geographic Perimeter Stateless Routing protocol (GPSR) [3] avoids the problem of communication voids by forming a planar graph and routing the packets on the perimeter of this graph. Location-Aided Routing (LAR) [8] uses "Expected Zone" and "Request Zone" in order to limit the search for a new route. The geographical information is not directly used for routing but for limiting the route request flooding procedure.

2.2 Energy Aware Routing

In ad-hoc and sensor networks minimizing the energy consumption is of great importance. Therefore, instead of using traditional metrics, such as hop-count, new energy-aware metrics were introduced for route selection criteria [23]. An example of these metrics is time to network partition. In [24] a group of flow augmentation and flow redirection algorithms have been proposed. These algorithms aim at balancing the energy consumption rates among different nodes proportional to the residual energy. The problem with this approach is that one needs a prior knowledge of the source/destination pairs, and the data-generation rate between each pair. This information is not usually available in advance in ad-hoc and sensor networks. In [15] authors propose a clustering scheme, with a randomized cluster-head rotation in order to balance the energy consumption among nodes. However, this approach assumes adjustable transmission range. In addition, the cluster-head is assumed to be directly connected to the gateway node. These two assumptions differ from our assumptions in this paper.

2.3 Probabilistic Routing

Probabilistic flooding has been studied in [6]. The idea is to restrict the flooding by introducing a forwarding probability. The forwarding probability determines if a node that receives a message should forward it or do nothing. The forwarding probability is a fixed number for all the nodes in the network, and is selected in a way to ensure complete connectivity in the network.

GeRaF [7] is another protocol which uses restricted flooding. The goal of this protocol is to enable putting the nodes to sleep and waking them up without coordination. It also integrates routing, MAC and topology management into a single layer. The actual routing scheme in GeRaF is a probabilistic flooding. Each node upon receiving a message decided independently whether it should act as a relay or not. This decision is made based on the priority of the node, which is calculated based on the relative location of the node itself compared to the distance between the previous node and the destination.

In [5] Barrett et al. introduce a family of routing protocols which they term Parametric Probabilistic routing. These protocols are a variation of probabilistic flooding. However, instead of a fixed forwarding probability for all the nodes, each node determines its own retransmission probability using various parameters, such as the difference in distance between source and destination and distance from the current node to the destination.

In [10] the authors suggest an energy-aware probabilistic method for routing in sensor networks. In their scheme, the destination initially floods the network to setup the routes. During this initial phase, each route's overall energy consumption is determined. When a node is ready to send data to this destination, which is generally a fixed base station, it probabilistically chooses one of the available routes based on the determined overall energy consumption of the path. The problem with this scheme is that first the destination has to initiate the path setup procedure by flooding the network which makes it inefficient. Second, this approach is specific to the sensor networks where the main purpose of the network is to gather data and route it to the base station, i.e. many-to-one traffic. Therefore, this method can not be efficiently used in general ad hoc network setting, where there is many-to-many traffic flow.

Our contribution: In this paper we introduce probabilistic geographic routing protocol (PGR) for ad-hoc and sensor networks. PGR is an energy-aware decentralized routing protocol. Our approach differs from the previous work in a number of ways. First, the nodes do not need any global knowledge of the data flow, or any clustering scheme in order to accomplish energy balancing. Secondly, PGR probabilistically selects the next hop from a set of candidate nodes instead of the conventional deterministic approach. This helps eliminate the complexity of route selection introduced by most deterministic approaches. To our knowledge this approach has not been explored before. In the existing probabilistic schemes, each node decides whether it will relay a packet or not, and if it does, the packet is broadcasted to all the neighbors, such as in the probabilistic flooding.

We have implemented PGR in NS-2, and have compared its performance in terms of throughput, end-to-end delay, and network lifetime to GPSR and probabilistic flooding. In addition, we have verified PGR on a real sensor network test-bed.

3. PROBABILISTIC GEOGRAPHIC ROUTING

3.1 Algorithm Description

First we describe our assumptions throughout the paper. Each node in the network is aware of its (x,y) coordinates in the plane. The node can either be equipped with a GPS device, or use some other localization scheme, such as the signal-strength based localization [18-22].

Every node which has a packet to send, called the *source*, needs to know the location of the destination node. This could be accomplished using a location database [4]. Any intermediate node that forwards the packet toward the destination does not need to know the location of the target since this information is included in the message header.

We assume the wireless links are asymmetric. The existence of asymmetric links in wireless networks has been empirically shown in [16]. We denote the reliability from a node i to its neighbor j, r_{ij}, the *forward reliability*, and the reliability from neighbor j to node i, r_{ji}, the

backward reliability, where these two reliabilities are different from each other.

Now we are ready to explain PGR in more detail in the following sections.

3.1.1 Neighbor Discovery and Maintenance

At the beginning of the deployment, the nodes need to gather some initial information on who their neighbors are, and how good of a connection they have to each neighbor. The discovery time is decided at the deployment. The longer the discovery period is, the better the initial link reliability estimation is. During the discovery phase each node sends "hello" messages every *t* seconds. These messages contain the geographic location
of the sending node, its residual energy, and a list of its neighbors with the corresponding reliabilities. These hello messages are used during the discovery phase to estimate the link reliability from a node to *all* of its neighbors. For example, when node *j* sends a hello message, it includes a list of the nodes it can hear from with the reliability estimation r_{ji}.

The reliability is measured as the fraction of the number of messages that were received successfully in a given time interval over the number of messages expected to have been received during that time interval given the beaconing frequency is *t* seconds.

Different types of link estimators can be used, such as Exponentially Weighted Moving Average (EWMA), or Time Weighted Moving Average (TWMA) [16]. It has been shown in [16] that EWMA performs better than other estimators in terms of robustness and stability; therefore, we will use EWMA as the link estimator in our work. We will explain EWMA in more detail in section 4.2.

At the end of the discovery phase, every node has a list of all the nodes it can hear from with the corresponding link reliability estimation. For example, node *i* has a list of all nodes *j* that it can hear from, with r_{ji}. Lets assume that the size of the neighbor table of node *i* is N. At the end of the discovery phase, node *i* picks N of its neighbors with the highest values for the reliability, i.e. highest r_{ji}. Note that we use the backward reliability since it corresponds to the link quality from node *i* to its neighbor *j*, i.e. how well node *j* hears node *i*.

It is also possible to pick all the nodes with r_{ji} greater than a given threshold, if the neighbor table size is not a limitation. Every entry in the neighbor table consists of the corresponding node id, the geographical location, the corresponding link reliability, and the residual energy of the neighbor.

After the initial setup phase, the nodes enter the maintenance phase. During the maintenance, each node continues to send "hello" messages as before. The frequency of the beaconing can be decreased in the maintenance phase to reduce the energy consumption and the protocol overhead. In order to keep the neighbor table up-to-date, the nodes refresh their neighbor table every T seconds. If a neighbor's link reliability has dropped over time, that node will be replaced by a neighbor with higher link reliability. The procedure used to refresh the neighbor table is explained in detail in section 4.2.

3.1.2 Probabilistic Geographic Routing Protocol

As mentioned before, we assume the nodes know their geographical location and the location of the destination node. If the source does not know the destination's location, it can use a location service scheme [4] to determine the coordinates of the target. We explain the routing algorithm using figure 1.

Let us assume that node S is the source, D is the destination, and nodes 1 through 8 are in S's neighbor table. When S wants to send a packet to D, it looks in its neighbor table, and selects the neighbors that fall within an angle θ from D, as shown in figure 2. The initial value for θ can be picked arbitrarily. If S can not find more than one neighbor within the initial angle, it will increase θ until it finds at least two neighbors. If S has to open the angle θ beyond 180˚ to find a neighbor, it stops at that point, and drops the packet. The motivation behind this approach is to guarantee an "almost" loop-free protocol. Given the angle θ will never be opened up beyond 180˚, a packet will never be sent in the "backward" direction, and will always have a positive progress toward the destination. However, to ensure that a packet does not cycle in a loop, every time the packet is forwarded, the id of the forwarding node is included in the packet header. Once a packet is received by a node, its header is examined, and if the current node is listed in the header, the packet is dropped.

Now let us assume that S was able to find at least two neighbors within the angle θ, nodes 1 through 4 in our example of figure 2. These four nodes are the candidates for the next hop. Source S then proceeds to assign probabilities to each of these candidate nodes using their corresponding residual energy and backward link reliability.

It is worth mentioning that the inverse of the link reliability is approximately equal to the number of retransmissions required on a given link, if we think of each transmission as a Bernoulli trial. In a Bernoulli trial, the average number of trials before success is equal to *1/p*, where *p* is the probability of success. We use this fact along with the residual energy to calculate each candidate's probability. Let *R(j)* be the number of retransmissions required over a link from *j* to another node. The probability associated with a given candidate node *j* is calculated as follows:

$$p(j) = E_{res}(j)/R(j)$$

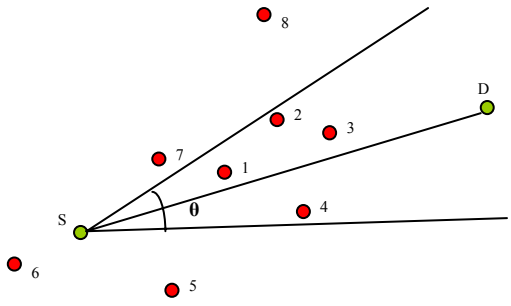

Figure 1: Example of how PGR works.

Therefore, if the backward reliability to one candidate node is not as good as another candidate, the probability corresponding to that neighbor will be lower. Also, if the residual energy of a candidate node is lower than another candidate neighbor, its corresponding probability will be smaller. This approach will ensure that the energy consumption is balanced among nodes, and a node with a reliable link will not be drained out of energy due to being selected continuously. Once the probabilities are assigned to each of the candidate neighbors, a Roulette wheel selecting algorithm is used to pick a node proportional to its assigned probability.

3.2 Analysis of the Selection Metric

In this section we explain the motivation behind using $E_{res} \times r_{ji}$ as the cost metric. It is obvious that a node with less residual energy should not be used as frequently since this will cause it to run out of energy faster. Therefore, the cost metric has to be proportional to the residual energy of a node.

As explained before, the number of retransmissions on a link is the inverse of the link reliability. Therefore, by using the link reliability in our metric, we can account for the number of retransmissions. Since the algorithm selects the next hop probabilistically, it does not explicitly try to minimize the number of retransmissions. However, a node with higher link quality has a higher probability of getting selected. Therefore, the algorithm implicitly minimizes the number of retransmissions locally at every hop.

In order to see how our algorithm performs in terms of the overall number of retransmissions compared to the global minimum, which is obtained using Dijkstra's algorithm, we simulated three different cases.

Dijkstra's shortest path algorithm is used to find the global minimum value for the total number of retransmissions from a source to destination.

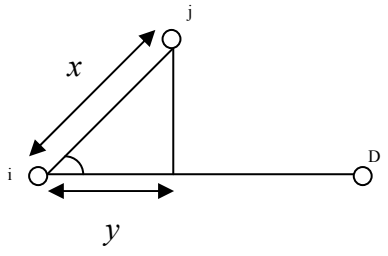

Figure 2: Maximizing the value of $\frac{y}{x^{\alpha}}$ at each step.

The second case is maximizing the value of $\frac{y}{x^{\alpha}}$ at each step, where y is the net advancement toward the destination, as shown in figure 3, and x^{α} is the number of retransmissions on a link of length x. At each step the neighbor with maximum value of $\frac{y}{x^{\alpha}}$ is chosen as the next hop. This is a local scheme that attempts to maximize the advancement toward the destination while minimizing the number of retransmissions at each step.

The third case is the greedy algorithm, where at each step the neighbor which is the closest to the destination in Euclidean distance is chosen as the next hop.

We have used x^{α} as the number of retransmissions on a link since the path loss model in a wireless fading channel is given as $\frac{1}{x^{\alpha}}$. We simulated a network consisting of 150 nodes in a square area of side $1000m$. The nodes are placed uniformly at random in the square area, and each node has a transmission range of $150m$. Table 2 lists the values for $\sum_{Source}^{Destination} x^{\alpha}$ in each case obtained from the simulation. This summation gives the total number of retransmissions required to go from a random source to a random destination. The values outlined here have been averaged over 20 runs of the simulation.

As seen from table 2, the total number of retransmissions in the greedy algorithm is the highest. The second scheme has a value on the same order as the greedy algorithm but smaller.

Table 1: Comparison of the overall number of retransmissions using three different schemes.

Exponent α	Dijkstra's Algorithm	Max $\frac{y}{x^{\alpha}}$	Greedy Algorithm
2	1.21e+007	1.55e+008	2.25e+008
4	3.94e+008	3.00e+016	7.20e+016

Although maximizing $\frac{y}{x^{\alpha}}$ at each step is a local scheme with no global knowledge of the network topology, its performance in terms of minimizing the retransmission cost comes closer to the global optimum than the greedy algorithm.

4. SIMULATION RESULTS AND EVALUATION

To implement and compare the performance of PGR with the prior work, we simulated probabilistic flooding (PF) [6] and GPSR [3] as well. These two protocols are the closest to our protocol; the first one has the probabilistic nature, and the second protocol uses geographical routing.

4.1 Simulation Environment

We simulated our protocol using NS-2 and the wireless extension to NS-2 which was developed at Carnegie Mellon [25]. Ns-2 is a discrete network simulator developed at UC Berkeley. We also used the codes for PF

and GPSR that already existed in NS-2 [26]. In the following we describe NS-2 network model in some detail.

Data Link Layer Model: NS-2 implements the complete IEEE 802.11 physical and MAC layers with *virtual and physical carrier sensing*. It uses the sequence of Request to Send (RTS), Clear to Send (CTS), and Acknowledgement (ACK) for packet transmission.

Radio Propagation Model: The physical model we used is called *shadowing model*. The shadowing model takes into account channel fading and is the most realistic model to use for ad hoc wireless network simulations. The advantage of the shadowing model is that it extends the ideal circle model to a statistical model where the nodes near the edge of the circle can only probabilistically communicate.

The shadowing model is implemented in NS-2 using the following equation.

$$\left[\frac{P_r(d)}{P_r(d_0)}\right]_{dB} = -10\beta\log\left(\frac{d}{d_0}\right) + X_{dB}$$

Where β is the path loss exponent, P_r is the received power at a distance d, d_0 is the *reference distance*, and X_{dB} is a Gaussian random variable with mean zero and standard deviation σ_{dB}.

Energy Model: NS-2 has an implementation of a simple energy model in which every time a packet is received, the total energy of the node decreases by the value:

$$DecEnergy = P_{rcv} \times rcvTime$$

The same formula applies for decreasing the energy when a packet is transmitted, except instead of P_{rcv} we have P_{tx}.

4.2 Algorithm Implementation

As mentioned earlier, PGR has a setup phase in which the nodes gather link quality information. The discovery time is passed to the algorithm as an input. The longer the discovery time, the better the link estimates will be for the initial setup.

Link Estimation: In order to perform the link estimation, every node in the network sends "hello" beacons periodically. The link estimator we have used is Exponential Weighted Moving Average. EWMA is a linear combination of infinite history, each with exponential weights. Let $\overline{R}_t$ be the current reliability estimate, n the number of known missed packets based on the sequence number of the received packets, m the number of received packets, and w the window size. Then the reliability is updated as follows,

$$\overline{R}_{t+1} = \overline{R}_t \times \alpha + (1-\alpha) \times R_{t+1}$$

Where $0 \prec \alpha \prec 1$, $R_{t+1} = \dfrac{m}{m+n}$ and m and n get reset to 0 when $m+n \succ w$.

Neighbor Table Management: Nodes are added to the neighbor table after the discovery period is over. Neighbor tables are refreshed every T seconds to ensure that only "good" nodes populate the table at all times where the goodness of a node is defined by the user. For example, a "good" neighbor can be defined to be one with link reliability higher than 90%. In order to implement the neighbor table management, every node keeps a list of all the nodes it can hear from at all times. We call this list the *background table*. The reliability estimates are updated in this background table every time a node is heard from, or the node is added if it does not already exist in the background table. Every node in the background table has a timer associated with it, and when this timer expires, the corresponding node is deleted from the background table. This will ensure that the background table remains up-to-date. In order to populate and manage the routing table, we can choose one of the following schemes, based on the memory limitations of a node.

Every time the table is refreshed, N nodes with the highest link reliability are picked and added to the routing table. If this scheme is used, it is possible that the routing table will become unstable due to deleting and adding N nodes every T seconds.

Each node in the routing table has a counter associated with it which is initialized to a constant when the node is added to the table. Every time a "hello" beacon is heard from some node, all the counters are updated. The update rule is explained in the following:

If the "hello" beacon is from a node already in the routing table, increase its counter by one. If the message is from a node not in the routing table, decrease *all* the counters by one. If one of the counters becomes zero, remove the corresponding neighbor from the routing table. In order to fill up the vacant spot in the routing table, a new node from the background table is picked. This new node must have the highest link quality among all the nodes in the background list.

This second scheme makes the routing table more stable, due to a lower turnover rate, but at the same time it requires more memory for implementation.

4.3 Comparison Metrics

In order to compare the performance of PGR, GPSR, and PF, we choose the following three metrics:

Throughput or packet delivery ratio: This is defined as the ratio of the number of packets received by the destination, to the number of packets originated by the source.

Delay: The average time taken between when a packet was initially sent by the source, and the time it was *successfully* received at the destination.

Lifetime of the network: The time it takes until the first node in the network is completely drained out of energy. The more complete definition for the lifetime of the network is "time to network partition". Network partition occurs when there is a cut-set in the network.

Path Length: Path length is defined as the number of hops a packet takes to reach its destination.

Number of Retransmissions: The total number of valid retransmissions required to go from a source to destination. Valid retransmissions are defined as retransmissions due to not receiving an acknowledgement (ACK) for a transmitted data packet.

Another metric that is generally used is the routing overhead. However, we do not compare the three protocols based on this metric since PGR has a constant overhead due to the beaconing process, and is not load-dependent. The same is true for GPSR, and PF does not use beaconing.

4.4 Simulation Setup

In our simulations we have compared PGR, GPSR, and PF using the metrics described in section 4.3. Here we summarize the simulation parameters and scenarios.

We use the shadowing model with parameters $\beta = 2$ and $\sigma = 4$ which correspond to the outdoor environment [25]. The simulations are run on networks with 50-110 nodes, where the nodes are placed uniformly at random in a square area of 1000 m^2. The nodes have MAC 802.11 with 914 MHz Lucent WaveLAN DSSS radios. The nominal transmission range is 200 *m*. The initial energy of the nodes is set to 1000 J, and transmit and receive powers are equal and set to 0.281 J. The idle power is set to 0.035 J.

We simulate 3, 10, and 20 Constant Bit Rate (CBR) traffic flows. The source and destination of these flows are chosen at random. Each CBR flow has a rate of 4 packets per second, a packet size of 512, and max packet size of 10000. The simulation time is set to 500 seconds.

The discovery time for PGR is set to 40 seconds, the beaconing interval is set to 1.5 seconds, and the window for the EWMA is set to 4.

The probability of forwarding a packet in PF protocol is set to 0.9. It is possible to lower this probability to restrict the flooding more; however, by lowering the forwarding probability the number of paths from source to destination decreases which can cause the network to be disconnected.

4.5 Simulation Results

The results we present here have been averaged over 10 runs for each scenario.

Delay: The delay for 3, 10, and 20 CBR flows are shown in figures 5, 6, and 7. PGR has the smallest delay, on the order of 10^{-2} seconds. In the 3 CBR flows, PF and GPSR have almost the same delay, which is 0.4 seconds. As the number of CBR flows grows, PF experiences the most increase in delay, up to 12 seconds in the 20-CBR flow.

Throughput: The average throughput for PGR is about 95% over different network sizes. PF protocol has about 85% throughputs and GPSR about 67% for 3 CBR flows, figure 8. From figure 9, the throughput of PGR for 10 CBR flows is 95%, for GPSR 75%, and for PF 75%. The throughput of PF has dropped from the 3 CBR scenario by 10%. The reason for this drop is the increase in the network load. Since PF broadcasts the packet to all its neighbors, there is more congestion in the network as the number of flows increases. From figure 10, the throughput for PGR stays about 95% for 20 CBR flows, while PF and GPSR have an average throughput of 73%.

Based on the simulation results PGR has a consistent throughput over different networks sizes. This fact confirms that our protocol is both scalable in the network size and the workload.

Network Lifetime: We define the lifetime of the network to be the time it takes for the first node to run out of energy. In order to find the lifetime of the network under the three protocols, we simulated networks of sizes 50-110 with 10 CBR flows. The initial energy of the nodes was set to 50 J, and the simulation time was 1500 seconds. The numbers shown in figure 11 are averaged over 10 simulation runs. As seen from the figure, PF has the smallest network lifetime of about 250 seconds. This is expected since flooding is a very energy-consuming task. Even though there is a forwarding probability associated with the flooding in PF, it does not increase the lifetime of the network by a considerable amount since 90% of the nodes still broadcast their packets.

The lifetime of GPSR is on average about 700 seconds, and for PGR is about 900 seconds. This is a 30% improvement in the lifetime of the network.

Path Length: Table 2 shows the path length for GPSR and GP with 1 CBR flow. We have not included the path length for the PF since a packet can go along multiple paths in flooding.

As seen from table 2, the path length for GPSR is less than PGR. This is expected since PGR does not try to optimize for path length directly, but GPSR uses the greedy algorithm.

Table 2: Comparison of path lengths

Number of Nodes	50	60	70	80	90	100	110
GPSR	1.5	1.6	1.5	1.8	1.75	2.0	2.1
GP	1.75	1.8	1.60	2.0	2.01	2.8	3.5

Number of Retransmissions: This is the number of valid retransmissions on a link due to loss of ACK. We simulated networks of sizes 50-110, with 1 CBR flow. If we have more CBR flows, the number of retransmissions will increase due to the congestion and collision effects. We simulate one flow in order to find the overall number of retransmissions purely due to the probabilistic nature of the physical layer.

Figure 12 shows the total number of retransmissions required to reach the destination for GPSR and PGR. We did not include PF due to the multiple-path problem, as mentioned before. We observe that the number of retransmissions in the GPSR case is on average 90, and for PGR is 63 retransmissions, a 30% decrease.

Since PGR explicitly includes the link reliability into the cost function, it will locally minimize the number of retransmissions on a link. This in effect minimizes the total number of retransmissions from source to destination.

PGR will not obtain the global minimum since the optimization decision is made per-hop with no global knowledge of the network topology.

5. IMPLEMENTATION

In order to verify PGR, we have implemented our protocol on a sensor network test-bed. The test-bed uses the Telos motes from Berkeley [27], shown in figure 4. There are 29 nodes, 28 of which gather data and send it to a node called the "base station". The software used on the motes is Tinyos, which is an even-driven operating system for "wireless embedded sensor networks" [28].

Figure 3: Telos ultra low power mote with IEEE. 802.15.4 wireless transceiver.

6. DISCUSSION AND CONCLUSION

In this paper we introduced Probabilistic Geographic Routing protocol (PGR) which is a decentralized energy-aware routing protocol for wireless ad hoc and sensor networks. PGR uses geographical location along with residual energy and link reliability information to make routing decisions. Instead of deterministically choosing the next hop, PGR assigns probabilities to the candidate next-hop nodes. The probability assigned to each node is a function of its residual energy and the corresponding link reliability estimation. Using the residual energy in the cost function ensures that nodes with more reliable links are not drained out of energy too quickly. This will in turn increase the lifetime of the network. In addition, PGR attempts to locally minimize the number of retransmissions. Reducing the number of retransmissions contributes to saving energy, and increases the overall lifetime of the network. The other advantage of PGR is that it does not require keeping a state per route. This will reduce the routing protocol overhead and the amount of routing information that needs to be stored at each node. This makes PGR simple to use. Given the next hop is chosen probabilistically, it is also possible to pass down a list of the candidate nodes along with their probabilities to the MAC layer, and let the MAC layer choose a next hop. If the transmission to one of the candidate nodes fails, the MAC can choose another neighbor from the list, and so on. This will help expedite the routing process; if transmission to a neighbor fails, instead of the trying to retransmit on the same link multiple times, the MAC layer has the freedom to choose a different neighbor for retransmission. Finally, as argued in [security and stochastic routing papers], a probabilistic routing scheme has a built-in security advantage. Since the next hop is chosen randomly and not based on a deterministic rule, it is more difficult for an adversary to attack and intercept a message.

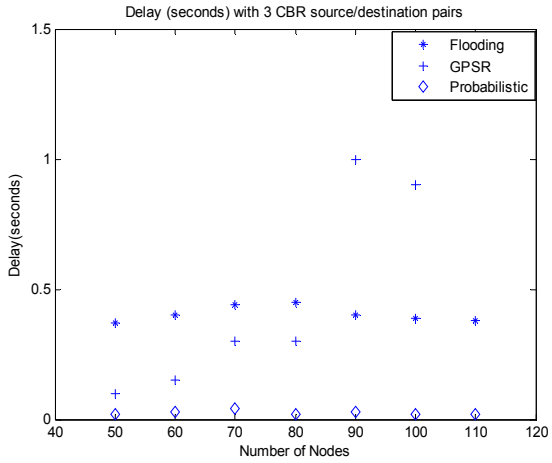

Figure 4: The packet delivery delay in seconds for 3 CBR flows.

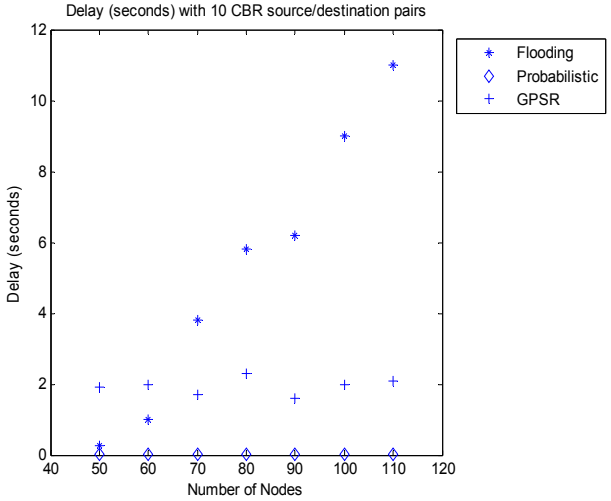

Figure 5: The delay in seconds for the three protocols. The delay for PGR is on the order of 10^{-2}. PF has the largest delay due to multiple-routes problem. PGR and GPSR have a relatively constant delay with the network size growing.

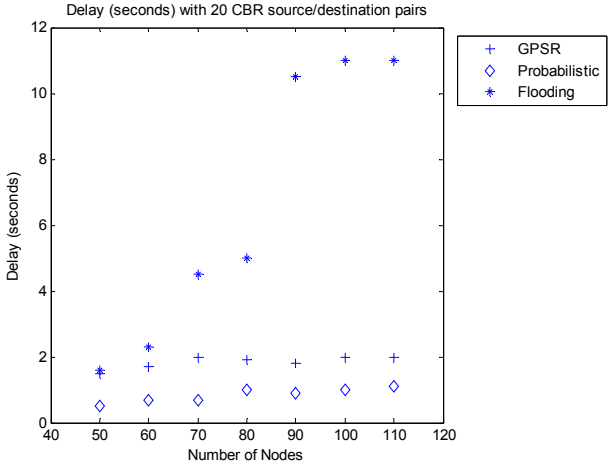

Figure 7: The average delay for the 20 CBR flow scenario. PF has the highest delay, and PGR has the lowest delay.

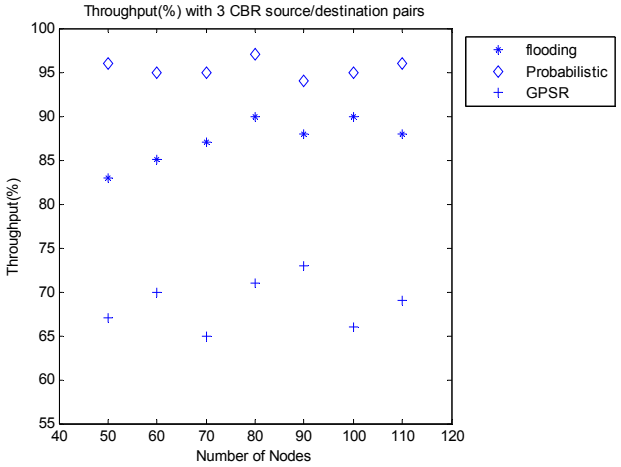

Figure 8: PGR has a very high packet delivery ratio. GPSR has the lowest value for the throughput.

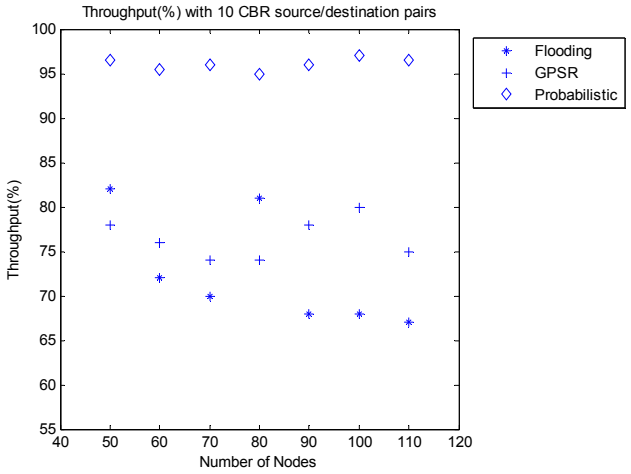

Figure 9: The packet delivery ratio for 10 CBR flow is shown in this figure. PGR on average has a 95% throughput for different network sizes.

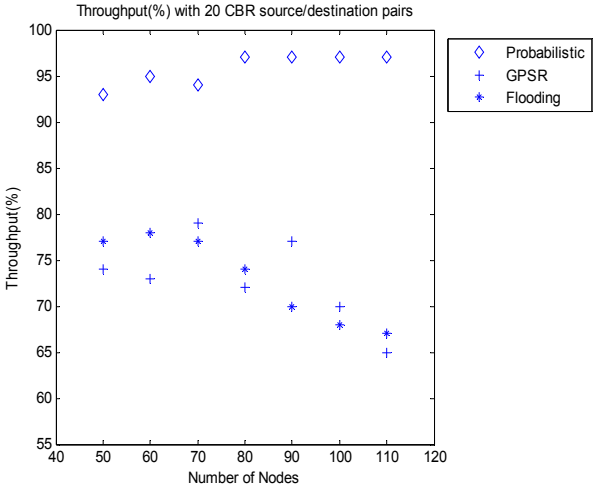

Figure 10: Throughput for 20 CBR flows. PGR has about 95% throughput on average in this scenario as well.

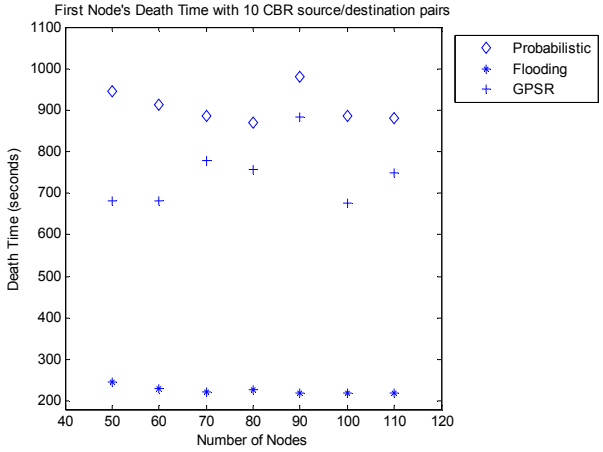

Figure 11: This graph shows the death time for the first node in the network. We see that the lifetime of the network is noticeably longer under PGR protocol.

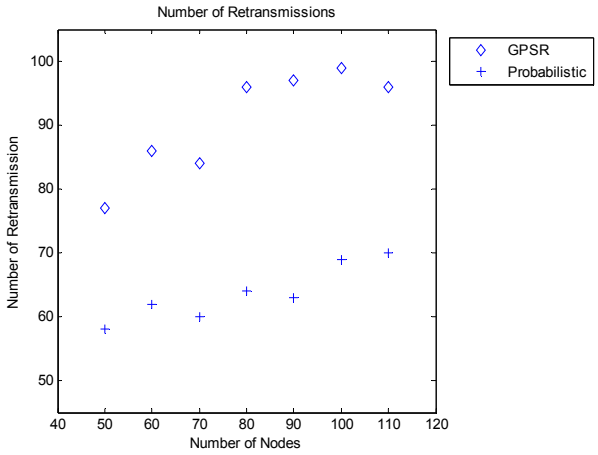

Figure 12: the overall number of retransmissions for going from a source to destination is higher for GPSR than PGR.

7. REFERENCE:

[1] Barrett, S. J. Eidenbenz, L. Kroc; Parametric Probabilistic Sensor Network Routing; WSNA 2003

[2] Broch, D. Maltz, D. B. Johnson, Y. Hu, J. Jetcheva; A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocol; Proceedings of Mobicom 1998

[3] Bulusu, John Heidemann, Deborah Estrin; GPS-less Low Cost Outdoor Localization for Very Small Devices; IEEE Personal Communication Magazine, October 2000

[4] Chang, L. Tassiulas; Energy Conserving Routing in Wireless Ad-Hoc Networks; Infocom, 2000

[5] Girod, D. Estrin; Robust Range Estimation Using Acoustic and Multimodal Sensing; IEEE/RSJ International Conference on Intelligent Robots and Systems, October 2001

[6] Hsieh, Raghupathy Sivakumar; Performance Comparison of Cellular and Multi-Hop Wireless Networks: A Quantitative Study

[7] Ju, Ramesh Givindan, Deborah Estrin; Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks

[8] Karp, H. T. Kung; GPSR: Greedy Perimeter Stateless Routing for Wireless Networks; Mobicom 2000

[9] Ko, Nitin Vaidya; Location-Aided Routing (LAR) in Mobile Ad Hoc Networks; Wireless Networks 2000

[10] Mauve, Jorg Widmer, Hannes Hartenstein; A Survey on Position-Based Routing in Mobile Ad-Hoc Networks

[11] Melodia, Dario Pompili, Ian Akyilidiz; Optimal Local Topology Knowledge for Energy Efficient Geographical Routing in Sensor Networks; IEEE Infocom 2004

[12] Michele Zorzi, Ramesh R. Rao; Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Energy and Latency Performance; IEEE WCNC 2003

[13] Priyantha, A. Chakraborty, H. Balakrishnan; The Cricket Location-support System; Proceedings of ACM Mobicom, 2000

[14] Rabiner Heinzelman, A. Chandrakasan, Balakrishnan; Energy-Efficient Communication Protocol for Wireless Microsensor Networks; Proceedings of the Hawaii International Conference on System Sciences; Jan 2000

[15] References:

[16] Sasson, David Cavin, Andre Schiper; Probabilistic Broadcast for Flooding in Wireless Mobile Ad Hoc Networks; Technical Report

[17] Savvides, C. Han, M. Strivastava; Dynamic Fine-grained Localization in Ad-hoc Networks of Sensor Networks; Proceedings of ACM Mobicom 2001

[18] Seada, Ahmed Helmy, Ramesh Govindan; On the Effect of Localization Errors on Geographic Face Routing in Sensor Networks

[19] Shah, Jan M. Rabaey; Energy Aware Routing for Low Energy Ad Hoc Sensor Networks

[20] Singh, M. Woo, C. S. Raghavendra; Power-Aware Routing in Mobile Ad Hoc networks

[21] Stojmenovic; Position-Based Routing in Ad Hoc Networks; IEEE Communication Magazine; July 2002

[22] Ward, A. Jones, A. Hopper; A New Location Technique for the Active Office. IEEE Personal Communications, October 1997

[23] Woo, Suresh Singh, C. S. Raghavendra; Power-Aware Routing in Mobile Ad Hoc Networks; ACM/IEEE Mobicom, 1998.

[24] Woo, T. Tong, David Culler; Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks; Sensys 2003

[25] Woo; Evaluation of Efficient Link Reliability Estimators for Low-Power Wireless Networks