PROHET, LAYHET, AND EGYHET: ROUTING PROTOCOLS WITH ASSURED DELIVERY RATES IN WIRELESS HETEROGENEOUS SENSOR NETWORKS

THESIS

Presented to the Graduate Council of
Texas State University-San Marcos
in Partial Fulfillment
of the Requirements

for the Degree

Master of SCIENCE

by

Zanxun Dai, B.S.

San Marcos, Texas
August 2013

PROHET, LAYHET, AND EGYHET: ROUTING PROTOCOLS WITH ASSURED DELIVERY RATES IN WIRELESS HETEROGENEOUS SENSOR NETWORKS

Committee Members Approved:

Xiao Chen, Co-Chair

Hongchi Shi, Co-Chair

Qijun Gu

Approved:

J. Michael Willoughby

Dean of the Graduate College

FAIR USE AND AUTHOR’S PERMISSION STATEMENT

Fair Use

Duplication Permission

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisors and committee chairs Dr. Xiao Chen and Dr. Hongchi Shi for their support, guidance, and insightful advice on my research and thesis. They have been great advisors and friends to me.

I am also grateful to Dr. Qijun Gu for his encouragement and comments. I would like to thank my lab mates for their insights, discussions, and jokes in the past three years. The experience was full of joy and enlightening.

I thank my parents for their support. My wife, Jade, has been my pillar along the way. I could not come this far without her encouragement.

This manuscript was submitted on May 30th, 2013

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

ABSTRACT

PROHET, LAYHET, AND EGYHET: ROUTING PROTOCOLS WITH ASSURED DELIVERY RATES IN WIRELESS HETEROGENEOUS SENSOR NETWORKS

by

Zanxun Dai, B.S.

Texas State University-San Marcos

August 2013

SUPERVISING PROFESSORS: XIAO CHEN and HONGCHI SHI

Due to requirements in different applications, sensors with various characteristics are deployed. Data routing in such heterogeneous wireless sensor networks (HWSNs) poses challenges. First, the heterogeneous features create asymmetric links in the communication graphs which are not dealt with by conventional routing algorithms using undirected graphs. Second, it is required to have an assured delivery rate for mission critical applications even with sensors communicating with each other through lossy asymmetric links. In this thesis, we propose ProHet, LayHet, and EgyHet which take advantage of asymmetric links to deliver messages to the sink with an assured delivery rate. To show the advantage of the proposed protocols, we compare them with the existing performance-guaranteed protocol by simulation. The simulation results show that ProHet, LayHet, and EgyHet outperform previous routing methods in terms of average delivery ratio, average hops, average packet replication and average control message overhead. As sensor energy is consumed, the performance of LayHet and EgyHet eventually degrades more slowly than that of ProHet.

I. INTRODUCTION

Wireless sensor networks (WSNs) are having a significant impact on our daily lives. In WSNs, sensors gather information such as humidity, temperature and light intensity from the environment, process them locally, and then communicate with others or send the information to the sink for further processing. In WSNs, sensors may not have the same transmission range, and there will be asymmetric links in the communication graph. For example, if node A can reach node B but B cannot reach A, the link between A and B is asymmetrical. Asymmetric links can be the result of: noise sources near a device affecting packet reception at that device [38]; nodes powering down to conserve energy [39]; devices transmitting with different powers explicitly causing unidirectional links [40]; and intractable factors such as barriers and environmental conditions affecting signal propagation [41]. Therefore, asymmetric links are a fundamental problem in wireless sensor networks. Due to asymmetry, the common undirected graph generated after abstraction turns into a directed graph, and the off-the-shelf routing protocols for general WSNs do not work or only work with higher overhead [25]. Many existing protocols have not handled asymmetric links or handle them in a costly way. 'Bra' is a protocol that addresses asymmetric links using reverse paths, but it does not guarantee desired delivery rate.

In this thesis, we consider notification systems [42] that are widely used in battlefields, financial institutions, emergency services, information technology, weather forecast, government, education, sports, health care, and so on. The delivery rate determines the effectiveness of these applications. It also affects other metrics like throughput, latency, overhead, and energy, because they are measured based on the delivered packets. Therefore, in this thesis, we study new routing protocols for asymmetric wireless networks that can guarantee the desired delivery rate with a high probability using minimum energy consumption.

To satisfy the needs of different networks, dynamic and static, we study reactive and proactive protocols.

Our first protocol that can satisfy our requirements is ProHet. It is a reactive protocol that establishes a path in each communication step and is suitable for more dynamic networks.

Our second protocol is LayHet, which is a proactive protocol that finds the path before routing. It is suitable for more static networks.

EgyHet is an energy-conserving version of LayHet, which reduces and balances the energy consumption of sensors to extend the lifetime of the wireless sensor network.

II. RELATED WORKS

In this section, we give an overview of related existing routing algorithms in WHSNs and probabilistic routing strategies.

Routing in Heterogeneous Sensor Networks

Routing in homogeneous sensor networks has been well studied, and many routing protocols have been proposed [14], [16], [17], [20], [21], [26], [30], [31], [35]. In these protocols, all sensor nodes have the same capabilities in terms of communication, computation, energy supply, reliability, etc. However, in applications such as aforementioned, heterogeneous sensors with different capabilities may be deployed. It is reported in [34] that when properly deployed, heterogeneity can triple the average delivery rate and provide a five-fold increase in the network lifetime. Routing in WHSNs should be rethought about. Simply using the routing protocols in homogeneous sensor networks does not take advantage of the diversity of the sensors and does not work well.

In the literature, there are a few routing protocols designed for WHSNs [1], [7], [9], [11], [13], [36]. The sensors in WHSNs are categorized into powerful and less powerful ones. Sensors form clusters, with the powerful ones being the cluster heads. There are two routing protocols used: Intra-cluster and inter-cluster. The intra-cluster protocol is used to route messages between less powerful nodes and their cluster heads, and the inter-cluster protocol is used to route messages between cluster heads. In these routing protocols, the capability of each individual sensor is not distinguished, and the asymmetric links are not fully utilized. Therefore we have proposed a protocol in [15] that differentiates diverse transmission ranges of different sensors and takes advantage of the asymmetric links to achieve assured delivery rate. However, our preliminary work does not study the relationship between the assured delivery rate and the network parameters. In this thesis, we enhance that in our analysis, give a more comprehensive description of ProHet, and conduct more simulations to justify our design idea and calculate the overhead more accurately.

Probabilistic Routing Strategies

The probabilistic routing strategy in WSNs is not a new topic, and there are various studies about it. Paper [28] uses probabilistic routing to disseminate information in a wireless sensor network without maintaining any routing table: The sensor nodes simply forward the received packets with some probability. Thus, it reduces traffic in the network and mitigates the broadcast storm problem. The authors in [4] propose Parametric Probabilistic Sensor Network Routing Protocols, a family of light-weight and robust multi-path routing protocols for sensor networks in which an intermediate sensor decides to forward a message with a probability that depends on various parameters, such as the distance of the sensor to the destination, the distance of the source sensor to the destination, or the number of hops a packet has already traveled. Probabilistic Flow-based Spread Routing Protocol in [32] makes the intermediate nodes forward packets with a probability based on neighboring nodes' traffic load and tries to achieve the balance of energy consumption when forwarding packets. In [8], the information obtained by sensors from the environment has different delivery probabilities according to their levels of importance to the end user. For example, the information that there is a chemical leak is more important than the information that everything is fine and should have a higher delivery probability. The authors propose a new method for information delivery at a desired reliability using hop-by-hop schemes.

In the above work, the probability is not based on a node's historical information of its delivery capability. Using historical delivery information may result in better performance. In this thesis, we explore historical statistics and propose a probabilistic routing protocol with assured delivery rate.

III. PRELIMINARY

Definitions of Nodes' Neighbor Relationships

A WHSN can be represented by a directed graph G = {V, E}, where V is the set of sensors (also called nodes) and E is the set of links (also called edges) in the network. For example, if sensor B is in the transmission range of sensor A, then there is a directed link from A to B. We assume graph G generated from the sensor network is a strongly-connected directed graph. Therefore, the sensor network is strongly-connected, too.

We categorize the neighbor relationships of sensors into four categories: (1) In-out-neighbor, (2) In-neighbor, (3) Out-neighbor, and (4) Non-neighbor. For two nodes A and B, as shown in Figure 1(a), if $A \rightarrow B$ and $B \rightarrow A$, then A and B are In-out-neighbor of each other. If only $A \rightarrow B$ (or $B \rightarrow A$) as in Figure 1(b) (or 1(c)), then A (or B) is an In-neighbor of B (or A), and B (or A) is an Out-neighbor of A (or B). If neither $A \rightarrow B$ nor $B \rightarrow A$, they are non-neighbors of each other, as shown in Figure 1(d).

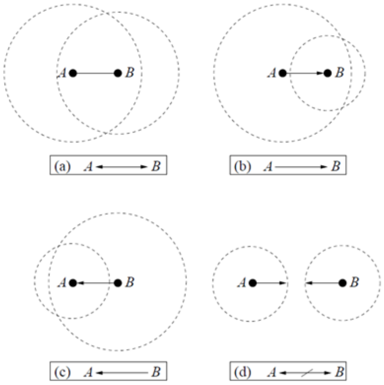


Fig. 1. The neighbor relationships between two nodes A and B. (a) A and B are each other's in-out-neighbor; (b) A is the in-neighbor of B and B is the out-neighbor of A; (d) A and B are non-neighbors.

Definitions of 1-hop, 2-hop, and n-hop Receivers

A node’s 1-hop receiver is the node’s Out-neighbor or In-out-neighbor.

A node’s two-hop receiver is the 1-hop receiver of the node’s 1-hop receiver (excluding the node’s 1-hop receiver and the node itself).

A node’s n-hop receiver is the 1-hop receiver of the node’s (n-1)-hop receiver (excluding the node’s 1-hop receiver, 2-hop receiver, … , (n-1)-hop receiver and the node itself)

Definition of Two-hop Neighborhood Information Model

In WSNs, the two-hop neighborhood information model, in which a node knows the information of its neighbors and the neighbors of its neighbors, is used by some researchers [2], [6], [33] to guide the routing process. This model is very attractive to large-scale WSNs because only local information is needed. This model is still helpful in WHSNs to steer the routing in the right direction. But because of the asymmetric links, the definition of the two-hop neighborhood information model in WHSNs should be changed to: A node knows its one-hop receivers and the one-hop receivers of its one-hop receivers. Still, the two-hop neighborhood information can be obtained by exchanging “Hello” messages between nodes in WHSNs.

Theoretically speaking, any k-hop ($k \geq 1$) neighborhood information model can be used. However, if one-hop neighborhood information is used, it is more like flooding which will cause large number of redundant data packets. If k-hop ($k \geq 3$) neighborhood information is used, it will introduce much more communication overhead among neighbors without bringing much benefit comparing with the two-hop neighborhood information model. Our later simulation confirms these.

Definition of Delivery Probability

A node's delivery probability $P_{delivery}$ is defined as the ratio of the number of packets successfully delivered by the node denoted by N_d and the number of packets forwarded by it, denoted by N_f. It can be expressed as:

$$P_{delivery} = N_d / N_f \tag{1}$$

N_d and N_f for a node will be recorded in the routing process so that $P_{delivery}$ can be calculated locally and timely. At the beginning of routing when N_d and N_f do not exist, a routing protocol can work in a flooding manner for a while to establish these values. After some rounds of packet delivery, every node's delivery probability will become stable. Thus, the historical information of the network has been established and can be used.

IV. PROHET PROTOCOL

In this section, we present the ProHet protocol, which has two parts: preparation and routing. The preparation part identifies neighbor relationships and finds a reverse path for an asymmetric link, and the routing part selects nodes, forwards messages, and sends acknowledgement. The details are as follows:

Preparation Part

First each node needs to identify its In-out-neighbors and In-neighbors (if there is any) by sending each other "Hello" messages (see algorithm Identifying Neighbor Relationships). The identification of a node's Out-neighbors has to wait until a reverse path is found.

Algorithm: Identifying Neighbor Relationships

1: Every node in the network broadcasts a "Hello" message.

2: If two nodes A and B can receive each other's "Hello" message and the corresponding "Acknowledgment" of the "Hello" message, then each adds the other to its In-out-neighbor list.

3: If A receives B's "Hello" message, but not the "Acknowledgement" of its own "Hello" message, then A knows that B is its In-neighbor and adds it to its In-neighbor list. Then, A will find a reverse routing path to B.

Fig. 2. The algorithm of identifying neighbor.

Next, for each node that has an In-neighbor, it is necessary to find a reverse path using the Finding a Reverse Path algorithm. Finding a reverse path can fully utilize the asymmetric links in the HWSNs. The study of [25] shows that a significant percentage of links in WHSNs are asymmetric and the connectivity of the network can be up to 97% when the maximum reverse routing path length (here "length" means the number of hops) is set to 3. Based on their observation, we can find reverse paths for most asymmetric links by tracing back three hops.

After a reverse path is found, most nodes will establish their reverse routing paths to their In-neighbors. If a node receives more than one reverse routing path to an In-neighbor, it chooses the shortest one.

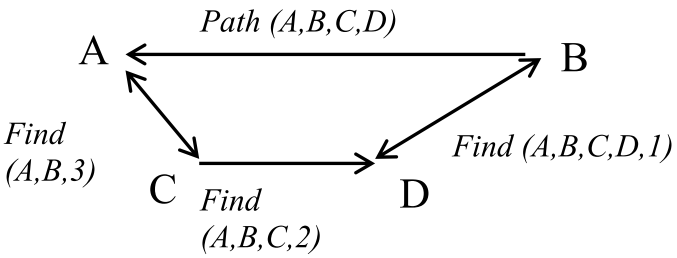


Fig. 3. Finding a reverse path example

Let's use the WHSN in Fig. 3 to explain the preparation part. In this network, A, B, C, and D are sensors with different transmission ranges. The directed links in the graph represent their neighbor relationships. After broadcasting "Hello" messages, sensors B and D can receive each other's "Hello" message and "Acknowledgement." So can sensors A and C. Thus they identify each other as In-out-neighbor. However, sensor A gets sensor B's "Hello" message, but does not receive B's "Acknowledgement" to its own "Hello" message. It knows that B is its In-neighbor. Then, it starts to find reverse routing path to B by broadcasting a “Find” message (A, B, 3). The number in “Find” represents the expiration length, initially set to 3. The “Find” message is received by sensor C. Sensor C is not the destination node and the expiration length is 3, so it will rebroadcast the message by changing it to (A, C, B, 2). After sensor D receives the message, it is not the destination either and the expiration length is 2, so it will rebroadcast the message by changing it to (A, C, D, B, 1). When B receives the message, it sees that it is the destination. It knows by now that source A is its Out-neighbor and adds A to its Out-neighbor list. Also, it builds a “Path” message (A, C, D, B) and sends it to A. After A receives the “Path” message, it gets its reverse routing path to B: A → C → D → B.

Algorithm: Finding a Reverse Path

```
Node A tries to find the reverse routing path to each of its In-neighbors by
broadcasting a “Find” message containing the source ID (“A”), the
destination ID (the ID of the In-neighbor to which it wants to find the
reverse path (e.g. “B”)), and an expiration length of 3 hops.

if some node C receives a “Find” message, then

    if it is the destination node listed in the message, then
        it adds the source node to its Out-neighbor list
        and send the identified reverse routing path to the source node by a
        “Path” message containing the reverse route.
    end if

    if it is not the destination node and the expiration length is greater than 0
    then
        it rebroadcasts the message after the following modifications:
            decrease the expiration length by one;
            append its own ID to the message.
    end if

    in all other cases, it drops the message.
end if
```

Fig. 4. The algorithm of finding a reverse path.

Routing Part

The nature of wireless communication is broadcasting. So the easiest and most reliable way to transmit a packet to the sink is flooding. However, flooding will cause serious communication overhead known as “flooding storm.” In order to reduce overhead and achieve assured delivery rate, we only choose a number of forwarding nodes based on the historical statistics. Compared with conventional routing protocols in WSNs, which ignore the existence of large numbers of asymmetric links, ProHet takes advantage of asymmetric links to route packets with high delivery ratio assurance.

In ProHet, two-hop neighborhood information model is used, although information in one-hop or more than two-hop neighborhood can also be used, which we will justify in the simulation section. The basic idea is as follows: We choose a subset of two-hop receivers of a node with high delivery probabilities as forwarding nodes and choose the one-hop receivers that can cover the selected two-hop receivers to relay the message. The ProHet protocol contains three phases/algorithms: Selecting Nodes, Forwarding Messages, and Sending Acknowledgement. The Selecting Nodes algorithm chooses the subset of two-hop receivers and the corresponding one-hop receivers; the Forwarding Message algorithm forwards messages to the destination; and the Sending Acknowledgement algorithm sends back an “Acknowledgement” message for a successful transmission and updates the delivery probabilities of forwarding nodes. The details are given in the following:

In the Selecting Nodes algorithm, notation N1(v) denotes v’s one-hop receivers, and N2(v) denotes v’s two-hop receivers. Node u covers v if u is an In-out-neighbor or In-neighbor of v. SN2(v) and SN1(v) denote v’s selected two-hop and one-hop receivers, respectively.

Let's use an example to explain the Selecting Nodes algorithm in Figure 5.

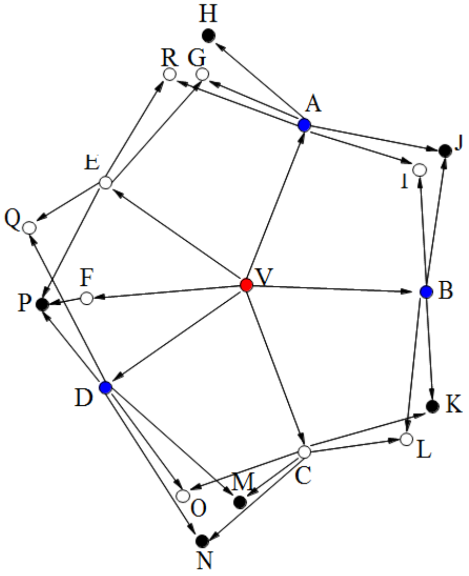


Fig. 5. Forwarding message process.

Suppose V (marked in red) is the node that has the packet to send. We use the algorithm to select v's two-hop (marked in black) and one-hop receivers (marked in blue). If there is a directional link *A* → *B* or a bidirectional link *A* ↔ *B*, it means *A* covers *B.* First, suppose six of *V* 's two- hop receivers *H, J, K, M, N, P* are selected into *SN2* (*v*) because their delivery probabilities are higher than or equal to the probability threshold *Pth* given the delivery rate *ρ*. We will explain the calculation of *Pth* in the ProHet analysis section. Next, we select the minimal set of *V* 's one-hop receivers to cover all of the nodes in *SN2(v)* using the greedy method. Node H is only covered by one one-hop receiver *A*. So, *A* is selected into *SN1 (v).* Node *A* also covers *J*. Next, the one-hop receiver that covers the most of the remaining nodes in *SN2(v)* is node *D*. So, it is also put into *SN1(v).* Now, the only node left in *SN2(v)* is *K* . It is covered by both *B* and *C*. Since neither *B* nor *C* covers any other remaining nodes in *SN2 (v),* we can choose either one of them to cover *K*. Choosing *B*, we have *SN1 (v)* = {*A*, *B*, *D*}.

Algorithm: Selecting Nodes

```
Node $v$ calculates the probability threshold $P_{th}$ according to Condition (4) in subsection V-C given desired delivery rate $\rho$.
$v$ selects a subset of its two-hop receivers whose delivery probability $P_{delivery}$ is higher than or equal to $P_{th}$ into the set $SN_2(v)$;
$v$ finds the minimal set of its one-hop receivers to cover all the nodes in $SN_2(v)$ by the following:
repeat
    Add every $v \epsilon N_1(v)$ to $SN_1(v)$, if there is a node in $SN_2(v)$ covered only by $v$;
    Add $v \epsilon N_1 (v)$ to $SN_1(v)$, if $v$ covers the largest number of nodes in $SN_2(v)$ that have not been covered;
    If there is a tie, the choice is random;
until all the nodes in $SN_2 (v)$ are covered.
```

Fig. 6. The algorithm of selecting nodes.

Next, any source node and forwarding node will run the Forwarding Messages algorithm, where the forwarding number N_f is recorded.

After the message reaches the sink, the sink will send back an acknowledgement P_{ack} to all the forwarding nodes on the path using the Sending Acknowledgement algorithm. Because of the asymmetric links, the reverse paths may be used. On the way to send back P_{ack}, the delivery number N_d is recorded, and the node's delivery probability $P_{delivery}$ can be obtained using Formula (1). The value of $P_{delivery}$ is refreshed in every forwarding node each time a message is sent from a source to the sink, and then the sink sends back an acknowledgement to the source.

Algorithm: Forwarding Messages

```
The current forwarding node $v$ broadcasts the packet $P$ containing $SN_1(v)$, $SN_2(v)$, and the message to be delivered to the sink; the forwarding number $N_f$ of $v$ is increased by one;
Any node $u \epsilon N_1(v)$ rebroadcasts $P$ if it is in $SN_1(v)$ and increases its forwarding number $N_f$ by one and attaches $u$'s ID in $P$ as a forwarding node in the path;
repeat
    Set node $t$ in $SN_2(v)$ as the new "source" node "$v$" and apply Selecting Nodes and Forwarding Message algorithms;
until $P$ reaches the sink.
```

Fig. 7. The algorithm of forwarding messages.

Algorithm: Sending Acknowledgement

```
When the first copy of a packet $P$ reaches the sink, the sink sends an acknowledgement $P_{ack}$ of $P$ to all the forwarding nodes on the path from the sink back to the source. The later arrived copies of $P$ are dropped.
When an intermediate node $m$ receives $P_{ack}$, it increases its $N_d$ by one, and
if its previous node $t$ is its In-out-neighbor, then
    it sends $P_{ack}$ directly to $t$;
else if $m$ has a reverse path to $t$, then
    $m$ sends $P_{ack}$ to $t$ via the reverse path of the asymmetric link $t \rightarrow m$;
else
    $m$ simply drops $P_{ack}$
end if
```

Fig. 8. The algorithm of sending acknowledgement.

Also note that at the initial stage of running the routing protocol, every node's delivery probability does not exist. So, the ProHet protocol will work in a flooding manner. After some rounds of packet delivery, each node's delivery number N_d and forwarding number N_f have values, so every node's delivery probability can be computed locally and timely. After the routing protocol has been running for some time in the network, every node's delivery probability will become stable. Thus, the historical information of the network will be established and used.

Analysis

The key point for a node u to select its two-hop receivers is the value of the probability threshold P_{th} given a delivery rate ρ ($0 \leq \rho \leq 1$). In this section, we first give an upper-bound of ρ, then show that the delivery rate ρ can be achieved if it is within its upper-bound, and finally present a method to calculate P_{th}.

A. Upper-bound of ρ

Obviously, if the delivery rate ρ is set too high and the delivery probabilities of nodes in the network are too low and the network is sparse, then the delivery rate ρ cannot be achieved. So, we need to find out the upper-bound of ρ to make it possible to achieve the desired delivery rate.

Suppose node u has a total of m two-hop receivers whose delivery probabilities in non- increasing order are $p_1, p_2, \cdots, p_m$ which are obtained by Formula (1) based on historical data. The highest delivery rate a node u can achieve is when $P_{th} = p_m$, which means all of its m two-hop receivers are selected into the forwarding set. Then the following is true:

$$\begin{aligned} & 1-(1-p_1)(1-p_2)\cdots(1-p_m) \\ \geq\ & 1-(1-p_{min})^m \\ \geq\ & 1-(1-p_{min})^{out\text{-}d_{min}} \\ \geq\ & \rho \end{aligned}$$

In the above, p_{min} is the minimum delivery probability of nodes in the whole network. Thus $p_1, p_2, \cdot\cdot; p_m \geq p_{min}$. The value out-$\mathrm{d}_{\min}$ represents the minimum m in the whole network. So $m \geq out\text{-}d_{min}$. The values of p_{min} and out-$\mathrm{d}_{\mathrm{min}}$ can be known after a network has been set up and several rounds of packet delivery have been done. So ρ is upper-bounded by

$$\rho \leq 1 - (1 - p_{min})^{out\text{-}d_{min}} \tag{3}$$

That means, the delivery rate ρ that can be achieved depends on the nodes' delivery probabilities and the network density.

V. LAYHET PROTOCOL

In this section, we present the LayHet protocol which also has two parts: preparation and routing. The preparation part includes finding the reverse paths for asymmetric links, assigning layer numbers to the nodes, and adjusting the layer numbers periodically. The routing part includes the sender broadcasting H times to guarantee an assured delivery rate, each receiver calculating its probability to forward the message to reduce the number of replicated messages, and the updating of packet loss rates of links. The details are as follows:

Preparation Part

1) Finding a reverse path for each asymmetric link: We use the same algorithm as in ProHet.

2) Deciding initial layer numbers: In this part of the LayHet protocol (see Algorithm DILN), each node will find out its layer number which represents its shortest hop count to the sink. First a node u broadcasts an exploration packet EP containing a hop count c initialized to 0 and its ID to the sink. On the way, the hop count is incremented, and the path is recorded. After the sink receives EP, it waits for a while for more copies of EP to arrive. Then it picks the EP with the smallest hop count. This is because multiple EPs can arrive at the sink due to the nature of broadcast. The smallest hop count represents the shortest hop distance from u to the sink. The sink increments the smallest hop count by 1,

which is the final hop count c from u to the sink. Then the sink sends back an ACK of EP containing c to u via all the forwarding nodes on the path. Because of the asymmetric links, the reverse paths may be used in the process. After u receives c, it knows its layer number to the sink is c. A good point of the DILN algorithm is that each node may have multiple chances to adjust its initial layer number: once by the ACK from the sink addressing itself and other times by the ACKs from the sink addressing other nodes if it is the relay node on the paths. Multiple adjustments are necessary because of lossiness in the links. The closer the node to the sink, the more accurate its layer number can be, because it is more likely to be a relay node and thus has more chances to adjust its layer number. The accuracy of the layer numbers of lower layer nodes is more important than that of the higher layer nodes, because lower layer nodes are more likely to relay messages for others.

Algorithm DILN: Deciding Initial Layer Numbers

```
Node u broadcasts an exploration packet $EP$ containing a hop count $c = 0$ and its ID.
if a node v receives $EP$ then
    if it is the sink node then
        it waits for a while for more copies of $EP$ to arrive. Then it picks an $EP$ with the
        smallest hop count. It increments the hop count by 1 and generates an
        acknowledgement $EP_{ACK}$ containing the value of the current hop count $c$ and the
        path to all the forwarding nodes on the path back to the source $u$. The later arrived
        copies of $EP$ are dropped.
        When an intermediate node m on the path receives $EP_{ACK}$, it adjusts its own layer
        number according to hop count c and its location on the path.
        if its previous node $t$ is its In-out-neighbor then
            it sends $EP_{ACK}$ directly to $t$;
        else if m has a reverse path to $t$ then
            $m$ sends $EP_{ACK}$ to $t$ via the reverse path of the asymmetric link $t$ -> $m$;
```

Fig. 9. The algorithm of deciding initial layer numbers (DILN).

```
        else
            $m$ simply drops $EP_{ACK}$
        end if
    else
        it increments the hop count by 1, appends its ID to $EP$ and rebroadcasts $EP$
    end if
end if
After u receives $EP_{ACK}$, it knows its layer number to the sink is $c$.
```

3) Adjusting layer numbers periodically: After applying Algorithm DILN, because of the lossy links, some nodes may not be put into the right layers. But they still have chances to adjust their layer numbers later. To reduce the overhead, the adjustment of layer numbers can be embedded in Algorithm UPR-P. When a node u communicates with its In-out-neighbor or Out-neighbor v to find out the packet loss rate of link uv, besides sending back the number of messages that v receives, v will also send back its layer number. If u’s layer number is at least 2 more than v’s layer number, u will adjust its layer number to v’s layer number plus 1.

Routing Part

The routing part of LayHet contains three phases: Broad-casting H times, Forwarding messages, and Updating packet loss rate periodically (see Algorithms BRD-H, FWD-M, and UPR-P, respectively). The assured delivery rate is preset to Δ. In Algorithm BRD-H, before any routing in the network begins, the packet loss rates of the links between a source node u and its K lower layer In-out-neighbors or Out-neighbors are generated randomly because the network does not have any routing history. Later the packet loss rates are updated by Algorithm UPR-P. After u knows the packet loss rates of the links, it broadcasts the message it wants to send to the sink H times so that at least one of its K lower layer In-out-neighbors or Out-neighbors can receive the message in order to achieve the assured delivery rate Δ. Next in the FWD-M algorithm, a receiving node v will forward the message at a probability of Γ to avoid flooding the network with unnecessary messages. The formulas to calculate H and Γ are presented later in the analysis section. The UPR-P algorithm updates the link packet loss rates periodically so that the next routing can be guided by more accurate information in the network.

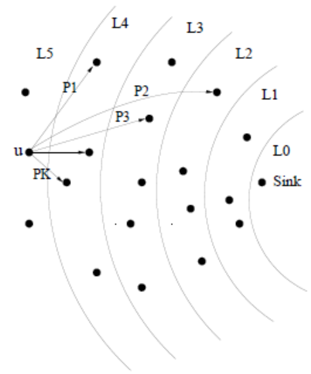


Fig. 10. LayHet data forwarding scenario.

We use an example in Fig. 10 to explain the LayHet protocol. Each black dot represents a sensor which is responsible for collecting data. The rightmost node is the sink which is responsible for processing data after collection.

In the beginning, each node applies Algorithm DILN to determine its layer number to the sink. After the initialization, the nodes are put into different layers relative to the sink. Because of the lossy links in the network, some nodes may not be put into the right layers. But the nodes can use Algorithm UPR-P to adjust their layers later.

After the node layers are identified, routing can be carried out. Suppose a source node u in Layer L5 wants to send a message to the sink. It has K In-out-neighbors or Out-neighbors in the lower one-hop, two-hop, and three-hop layers. A node may have a one-hop In-out-neighbor or Out-neighbor in the lower two- or three-hop layers because we consider opportunistic communications exploiting the nature of broadcast. Based on the packet loss rates of links to these neighbors, u broadcasts the message H times calculated by formula (2). This guarantees that at least one of these neighbors will receive the message with a high probability. Then each of these receivers will decide the probability Γ to forward the message by formula (3). The purpose of the forwarding probability is to avoid flooding the network with replicated packets. If a node chooses to forward, it becomes the new source and reapplies the routing protocol. Every T period of time, a sender will update the packet loss rates of its links so that the calculations of H and Γ can be more accurate next time.

Algorithm BRD-H: Broadcasting H Times

```
Source node u finds out the packet loss rates $p_1; p_2;\dots ; p_K$ with its K lower layer In-out-neighbors or Out-neighbors. Before any routing in the network starts, the packet loss rates are generated randomly. Later, they are updated by Algorithm UPR-P periodically.
Node u calculates the number of times H it should broadcast using formula (2).
Node u broadcasts the message plus its link packet loss rates $p_1; p_2; \dots ; p_K$ H times.
```

Fig. 11. Algorithm BRD-H.

Algorithm FWD-M: Forwarding Messages

```
repeat
    If a node v receives a message from a higher layer neighbor u along with the packet loss rates of u's links, it uses formula (3) to decide its probability $\Gamma$ to forward the message.
    If it forwards, it becomes the new source and reapplies the BRD-H algorithm.
    If it does not forward, it will simply drop the message.
until the message reaches the sink.
```

Fig. 12. Algorithm FWD-M.

Algorithm UPR-P: Updating Packet Loss Rate Periodically

```
1: Each node u updates the packet loss rate of each of its links with its K lower layer In-out-neighbors or Out-neighbors every T time period.
2: Suppose node u sends out $N_s$ messages to node v during T time period. At the end of T, node u sends a message to v asking "How many messages out of $N_s$ have you received?"
3: After v receives the inquiry, it replies directly or through the reverse path with the answer "$N_d$". Also, it attaches to the message its layer number for u to adjust its layer number.
4: After u receives the answer, it updates the packet loss rate of link uv to $1-N_d/N_s$. Also u may adjust its layer number $N_s$ based on v's layer number.
```

Fig. 13. Algorithm UPR-P.

Analysis

In this section, we provide an analysis of LayHet to show that if H and Γ are properly selected, there is a high chance that the routing can achieve an assured delivery rate Δ and reduce the number of replicated messages in the network. In order to explain that, it is easier to do it reversely, that is: Given the assured delivery rate Δ, decide the number of broadcasts H and the forwarding probability Γ to meet the assured delivery rate Δ and reduce the number of replicated messages.

For a node u at layer i, it forwards data to the lower layer nodes by broadcasting. Node u may need to broadcast several times so that at least one node in the lower layers receives the message. The number of broadcasts H depends on the link qualities from u to its lower layer In-out-neighbors or Out-neighbors. When a lower layer node v receives the message from u, it will forward the message with probability Γ, or in other words, with probability 1- Γ, it will drop the message.

If node u in layer i in Fig. 10 needs to send a message to the sink. In the worst case, the message needs to travel i hops to reach the sink. Assume transmission in each layer is the same. To guarantee the overall assured delivery rate Δ, in each layer, we should guarantee the success rate of the transmission to be at least $\Delta^{1/i}$. We first decide the number of broadcasts H to satisfy the requirement. We assume the packet loss rate of the link from u to its j-th In-out-neighbor or Out-neighbor is p_j. A transmission is successful if at least one of the lower layer nodes receives the message. The probability is:

$$Pr\{at\ least\ one\ lower\ layer\ node\ receives\ the\ message\ after\ H\ transmissions\}$$
$$= 1 - (\prod_{j=1}^{K} p_j)^{H}$$

Let $$1 - (\prod_{j=1}^{K} p_j)^{H} \geq \Delta^{\frac{1}{i}}$$

We have $$H \geq \frac{\ln(1 - \Delta^{\frac{1}{i}})}{\sum_{j=1}^{K} \ln(pj)} \qquad (2)$$

After node u broadcasts the message H times, the message is transmitted to one or more lower layer nodes with high probability. To reduce the number of replicated messages, not all the nodes receiving the message will forward the message to a lower layer. A receiver only forwards the message with probability Γ. Given that a message has been received by some lower layer nodes, we should make sure that at least one node will forward the message. That probability is:

$$Pr\ \{\ at\ least\ one\ lower\ layer\ node\ will\ forward\ the\ message\}$$
$$= \ 1 - \ Pr\ \{no\ one\ will\ forward\ the\ message\}$$

$$= 1 - \prod_{j=1}^{K} \begin{pmatrix} Pr\{\text{the jth node not recieved the message}\} \\ +Pr\{\text{the jth node receives the message}\} \\ \cdot Pr\{\text{the jth node does not forward the message}\} \end{pmatrix}$$

$$= 1 - \prod_{j=1}^{K} (P_j{}^{H} + (1 - P_j{}^{H})(1 - \Gamma))$$

$$= 1 - \prod_{j=1}^{K} (1 - \Gamma + P_j{}^{H}\Gamma)$$

Let $$1 - \prod_{j=1}^{K} (1 - \Gamma + P_j{}^{H}\Gamma) \geq \Delta^{\frac{1}{i}}$$

Then, $$1 - \Delta^{\frac{1}{i}} \geq \prod_{j=1}^{K} (1 - \Gamma + P_j{}^{H}\Gamma) \geq (1 - \Gamma + P_{min}{}^{H}\Gamma)^{K}$$

in which p_{min} is the minimum value of p_j ; ($1 \leq j \leq K$).

Solving this inequality yields

$$\Gamma \geq \frac{1-(1-\Delta^{\frac{1}{i}})^{\frac{1}{K}}}{1-p_{min}{}^{H}} \qquad (3)$$

Since H and Γ are obtained based on our preset value Δ, it is guaranteed that our routing algorithm LayHet can deliver messages with an assured delivery rate Δ and reduce the number of replicated messages.

VI. EGYHET PROTOCOL

LayHet can be more energy-efficient by considering the remaining energy of nodes. The changes made in the upgraded protocol EgyHet are as follows: Algorithm BRD-H calls algorithm CAL-H to select only a subset of K lower layer out-neighbors according to their remaining energy as forwarders to satisfy the desired delivery rate. In algorithm CAL-H, S represents the selected forwarding nodes with the highest remaining energy. First, the K lower layer out-neighbors of node u are ordered in non-increasing order based on their remaining energy levels. Then starting from the node with the highest remaining energy, we add node one by one to S. After adding a new node, we use Formula (4) to calculate H. The H value may be reduced with the increase of the number of nodes in S. The algorithm stops if the newly added node does not reduce H any more or if all of the K nodes are added. After the H value is known, u broadcasts the message containing the packet it wants to send to the sink, the selected forwarding nodes in S, and its link loss rates to the forwarding nodes H times so that at least one of the lower layer out-neighbors can receive the message with a high probability in order to achieve the desired delivery rate Δ. In Algorithm FWD-M, only the selected nodes will decide their probabilities to forward. The unselected ones will simply drop the message.

Algorithm CAL-H: Calculating H value and selecting forwarders

```
Order node u's K lower layer out-neighbors in non-increasing order according to their remaining energy levels. Here we use a node's remaining energy level to represent the node. Suppose the sequence $E = \{E_1, E_2, \ldots, E_K\}$.
$S = \{E_1\}$, $i = 0$.
Calculate H based on sequence S using Formula (4).
repeat
    $H_{pre} = H$, $i = i + 1$.
    if $i == K + 1$ then
        return H and S
    end if
    $S = S \cup \{E_i\}$.
    Calculate H based on sequence S using Formula (4).
until $H == H_{pre}$
return H and $S = S - \{E_i\}$.
```

Fig. 14. Algorithm CAL-H.

VII. SIMULATION

In this section, we first justify several design choices for ProHet and then evaluate its performance by comparison with the following three protocols using a self-written simulator in Java language:

- Flooding, the conventional algorithm.
- Random-K, in which random *K* one-hop receivers are selected to forward packets.
- TopRatio-K, in which *K* one-hop receivers that have the highest delivery probabilities are selected to relay packets.

Then we compare them with ProHet because to the best of our knowledge, ProHet is the only one that handles asymmetric links with performance guarantee. The BRA protocol in [41] deals with asymmetric links but does not consider delivery rate and is shown not to guarantee delivery rate. Therefore, we compare our algorithms with ProHet.

Simulation Setup

We used the following metrics to evaluate the performance of the protocols:

- Delivery ratio: the ratio of the number of packets successfully delivered to the total number of packets generated.
- Average hops: the average hops of a packet successfully sent from a source to the sink.
- Average packet replication overhead: the average number of packet replications

needed to successfully deliver a packet.

- Average control message overhead: the average number of control messages needed to successfully deliver a packet. The control message includes all the communication messages except the main packet to identify neighbors, find a reverse path, and update nodes' delivery probabilities.

In our experiments, nodes were deployed in a $500m \times 500m$ area. To diversify the transmission ranges of nodes, we used the idea in [25]. That is, a node can have one of the three transmission ranges: the *minimum*, the *normal*, and the *maximum* transmission ranges. The normal transmission range is the average of the minimum and the maximum transmission ranges. Here, we set the normal transmission range, which is also the default transmission range to $50m$. *Node transmission diversity* is defined as the difference between the maximum and the minimum ranges. The link loss rate of each link was randomly set between 0% and 20%. In both Random- K and TopRatio-K algorithms, the value of *K* was set to 5. To implement message sending and receiving, a virtual concept of *time slots* was used. In each time slot, we randomly chose a sensor to generate a new message and let it send the message to the sink. Each node used a buffer to cache packets from other nodes. We assumed that all packets in the buffer can be transmitted to the next-hop node within one time slot. The simulation time was set to 10,000 time slots. During the experiments, we randomly generated 20 different deployments of heterogeneous sensor nodes and calculated the average performance in the simulation results.

Simulation Results

Asymmetric link and reverse path

We first studied the percentage of the asymmetric links in the network and the percentage of these asymmetric links that have a reverse path within 3 hops. Fig. 15 shows that about 30% of the total links in the network are asymmetric links when the node transmission diversity was set to 20*m* and the node number varied from 200 to 360. Fig. 16 indicates that over 90% of the asymmetric links can find their reverse paths within 3 hops using our algorithm, which justifies that setting the expiration length to 3 in our algorithm is good enough to find most of the reverse paths of asymmetric links.

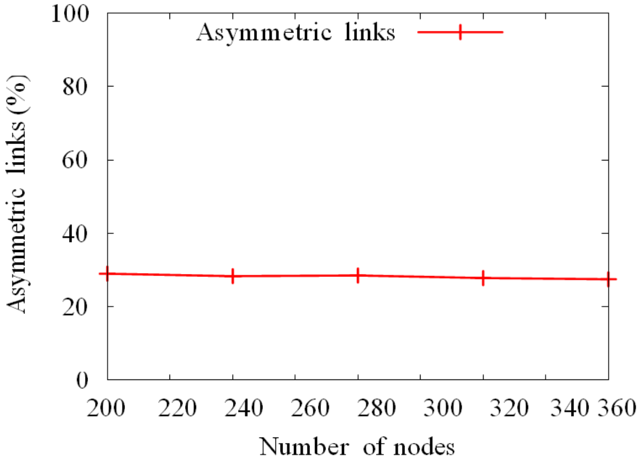


Fig 15. The percentage of asymmetric links in the network.

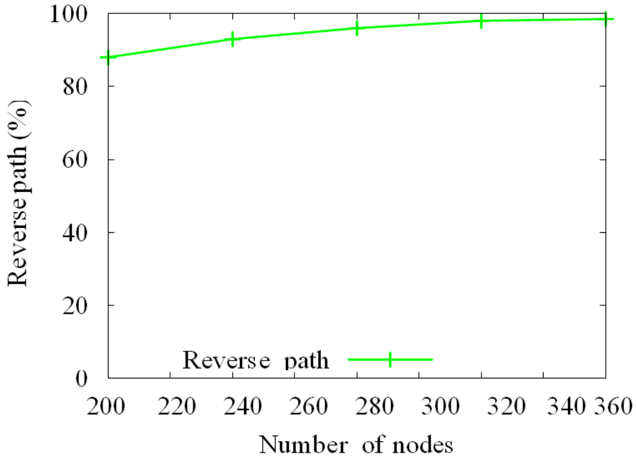


Fig. 16. The percentage of reverse paths found within 3 hops.

Comparison of ProHet performance using 1-hop, 2-hop, 3-hop neighbors

In order to explain that using the two-hop neighborhood information model in our algorithm is reasonable, we compared the performance of the one-hop, two-hop, and three-hop information models. We used the same three transmission ranges for the nodes, set the node transmission diversity to 20*m*, and set the assured delivery rate to 80%. We found that the delivery ratio of the one-hop information has a marginal improvement over those of the two-hop and three-hop information models because it is more like flooding. However, the packet replication overhead of the one-hop information is significantly higher than those of the two-hop and three-hop information models as shown in Fig. 17. Considering the significant replication overhead in the one-hop model and the communication overhead among neighbors in the k-hop $(k \geq 3)$ model, we think using two-hop neighborhood information model is appropriate.

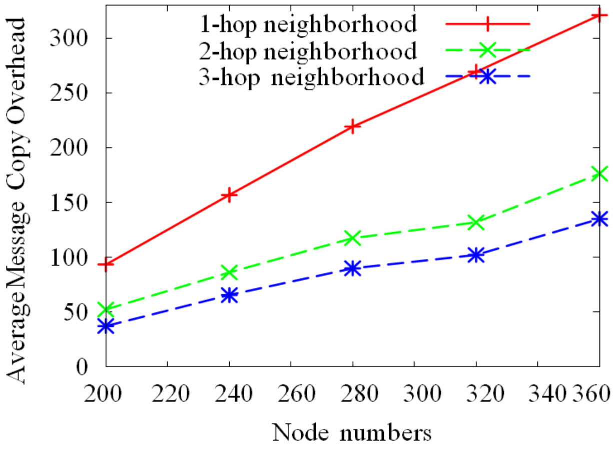


Fig. 17. Comparison of replication overhead using one- hop, two-hop, and three-hop neighborhood information models in ProHet.

Comparison of ProHet with AODV

In order to illustrate the improvement in delivery ratio if it is considered in the design, we compared the delivery ratio of ProHet with that of AODV. Though they have several differences: AODV is for *ad-hoc* wireless networks while ProHet is for heterogeneous sensor networks; AODV assumes symmetric communication links, while ProHet deals with asymmetric links, but both of them use reverse path in routing and have some similarity in design methodology. We used the same three transmission ranges for the nodes, set the transmission diversity to 20*m*, and set ProHet's assured delivery rate to 80%. From the results in Fig. 18, we can see that the delivery ratio of AODV cannot be guaranteed because it does not use asymmetric links and does not set achieving assured delivery rate as its design goal whereas in ProHet, with the increase of node numbers and thus more connections, it can reach the assured delivery rate and exceed.

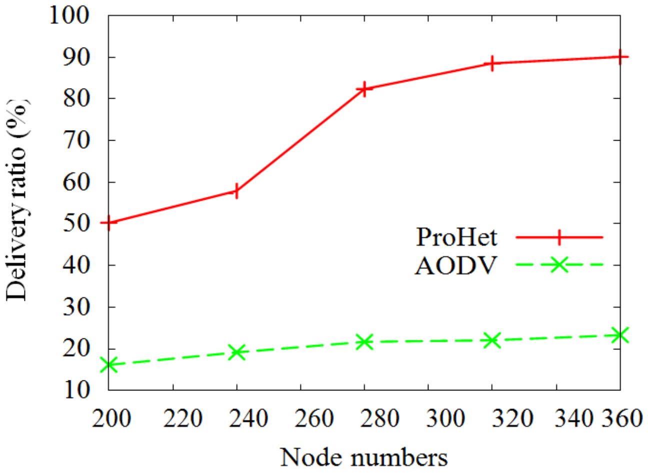
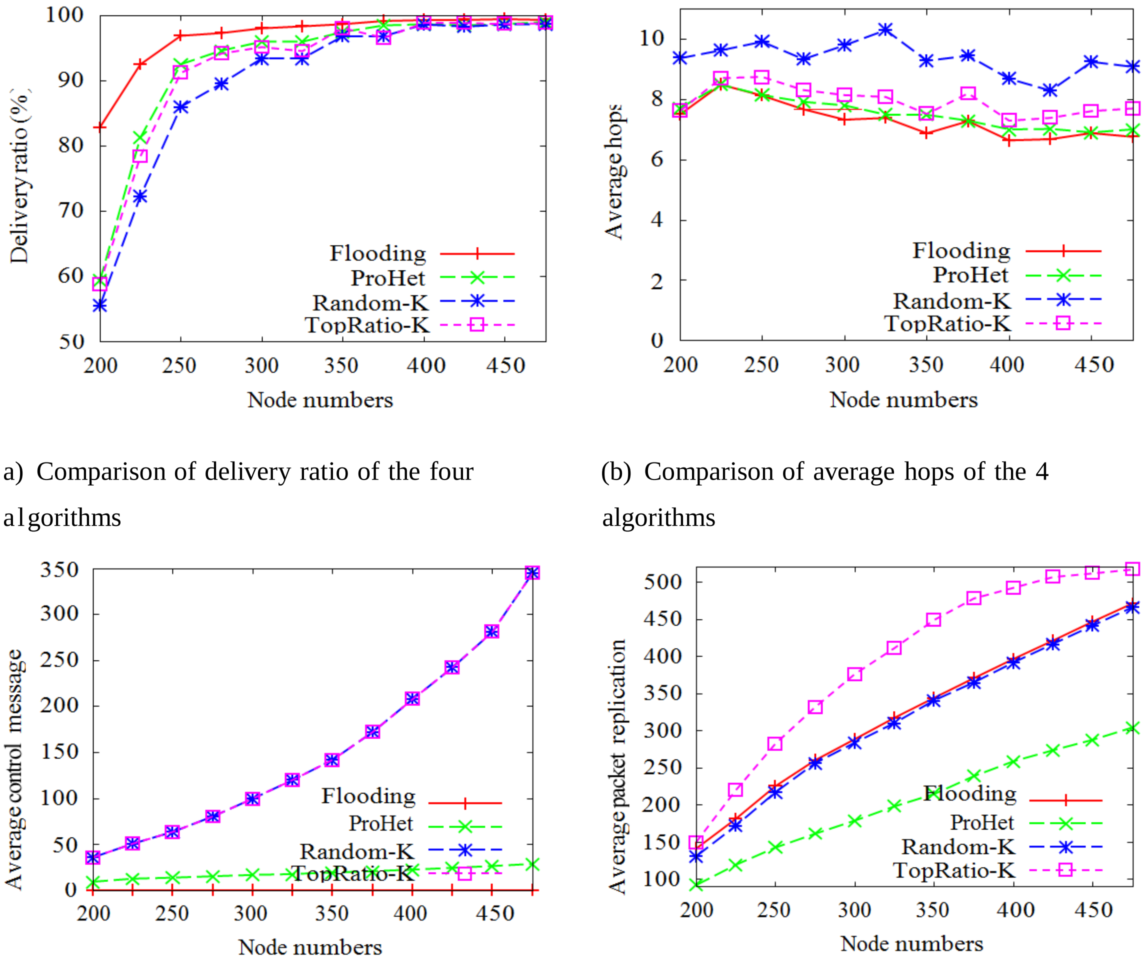


Fig. 18. Comparison of delivery ratio of ProHet and that of AODV.

Comparison of ProHet, Flooding, Top Ratio-K and Random-K

Then, we did the comparison of the ProHet protocol with the other three strategies (see Fig. 19). The number of nodes was set from 200 to 475, the node transmission diversity was set to $20m$, and ProHet's assured delivery rate was 95%. Fig 19(a) shows the delivery ratios of these strategies. Because of the nature of Flooding, it has a higher delivery ratio than the other three. ProHet's delivery ratio is better than that of TopRatio-K and Random-K. This means a careful selection of forwarding nodes based on P_{th} is better than selecting the top K or selecting randomly. Also, the delivery ratios of all the strategies increase with the increase of node numbers, which means more connections among nodes can result in more successful deliveries. Fig. 19(b) confirms that Flooding has the lowest hops to deliver packets. The fact that ProHet is close to using the lowest hops reveals that the latency of ProHet is low. Fig. 19(c) shows that ProHet has the least average packet replication overhead, which proves that the probabilistic strategy to choose forwarding nodes in the two-hop neighborhood is effective to remove a lot of redundant packets in the delivery process. Fig. 19(d) reports that Flooding does not have any average control overhead. The reason for this is that it does not keep neighbor information to route packets. ProHet's control overhead is much lower than that of Random-K and TopRatio-K. This is because Random-K and TopRatio-K establish neighborhood information every one hop while ProHet establishes neighborhood information every two hops in the routing process. In summary, the ProHet protocol can achieve similar performance of delivery ratio and latency to those of Flooding, but with a much lower replication overhead and a low control overhead.



a) Comparison of delivery ratio of the four algorithms

(b) Comparison of average hops of the 4 algorithms

(c) Comparison of average packet replication overhead of the four algorithms

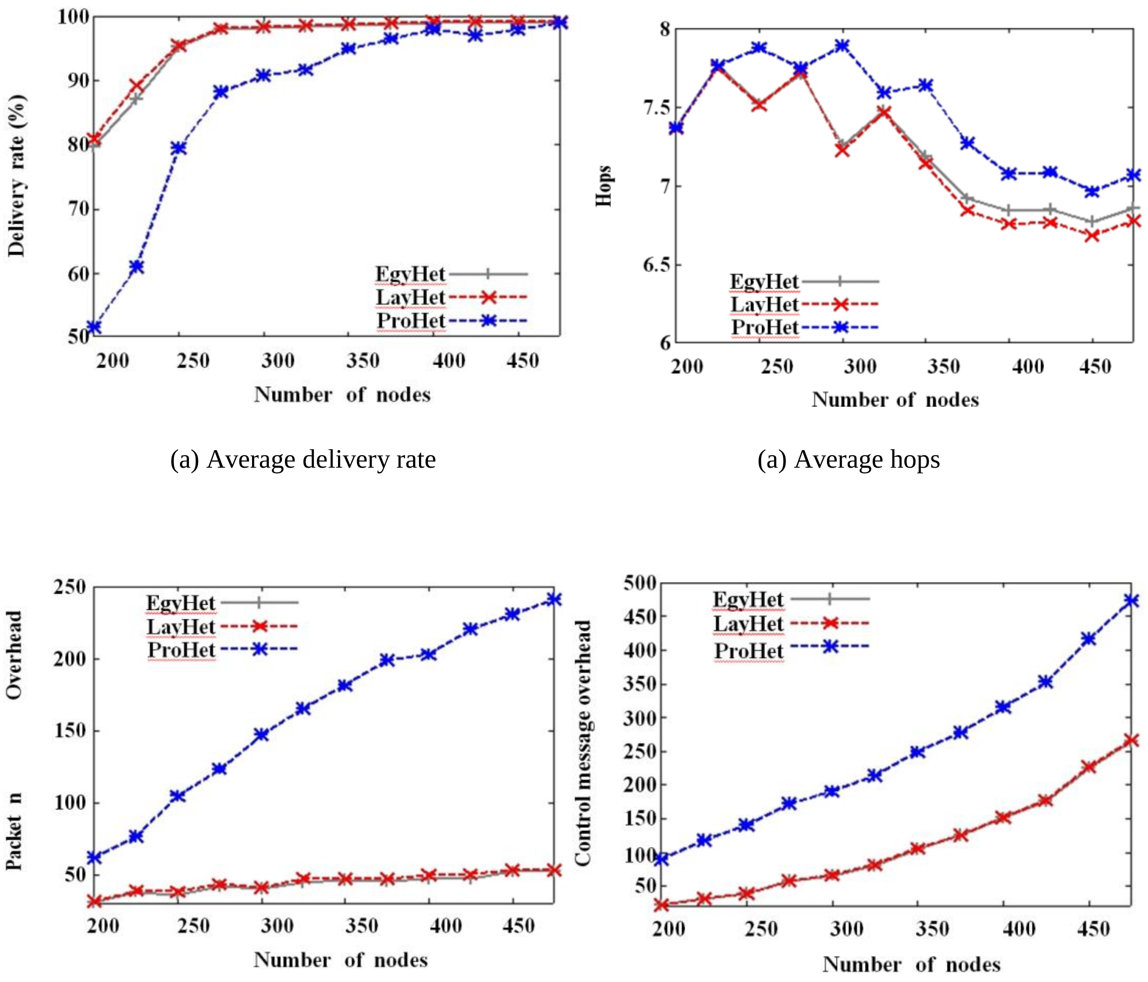
(d) Comparison of average control message overhead of the four algorithms

Fig. 19. Comparison of ProHet with Flooding, Random-K and TopRatio-K.

Comparison of ProHet, LayHet and EgyHet

The simulation results are shown in Figs. 20 and 21. From the results, we can see that all of the three algorithms can guarantee the desired delivery rate after the network density reaches a certain level. This is because with the increase of network density, the connection between nodes increases, so a message can get more chances to be delivered to the sink. Also LayHet and EgyHet can reach the desired delivery rate earlier than ProHet.

After the network becomes dense enough, for example, more than 250 nodes in the 80% desired delivery rate setting, ProHet's delivery rate will continue increasing but LayHet and EgyHet's delivery rates will keep at the 80% delivery rate level. This is because LayHet and EgyHet are designed to satisfy the desired delivery rate at each layer but do not go over that for the purpose of saving energy. In ProHet, on the other hand, more nodes will receive the message in denser topologies by the nature of opportunistic routing, so its delivery rate will eventually reach 100%. Comparing the delivery rates of LayHet and EgyHet, LayHet is better due to the fact that EgyHet uses a subset of LayHet's forwarders in each forwarding. For the number of hops to send a message from a source to the sink, LayHet and EgyHet are better than ProHet because the layer numbers in them embed the shortest path information. But the average hops of ProHet are also close to the ideal results. EgyHet has a little higher hop number than LayHet again because of its using a subset of LayHet's forwarders in each forwarding. The major improvement of LayHet and EgyHet over ProHet lies in the packet replication overhead and the control message overhead. The packet replication overhead of LayHet and EgyHet is substantially less than that of ProHet and the reduction in control message overhead in LayHet and EgyHet is also large even with their initial overhead to set up layer numbers counted. This indicates that the proactive protocols LayHet and EgyHet can save a lot of overhead in each hop by identifying node layers at the beginning whereas the reactive protocol ProHet has more overhead in each hop trying to discover the route. The packet replication overhead and control message overhead of EgyHet are smaller than those of LayHet (though not very obvious in the figures for the same subset reason and thus proves the improvement in EgyHet.



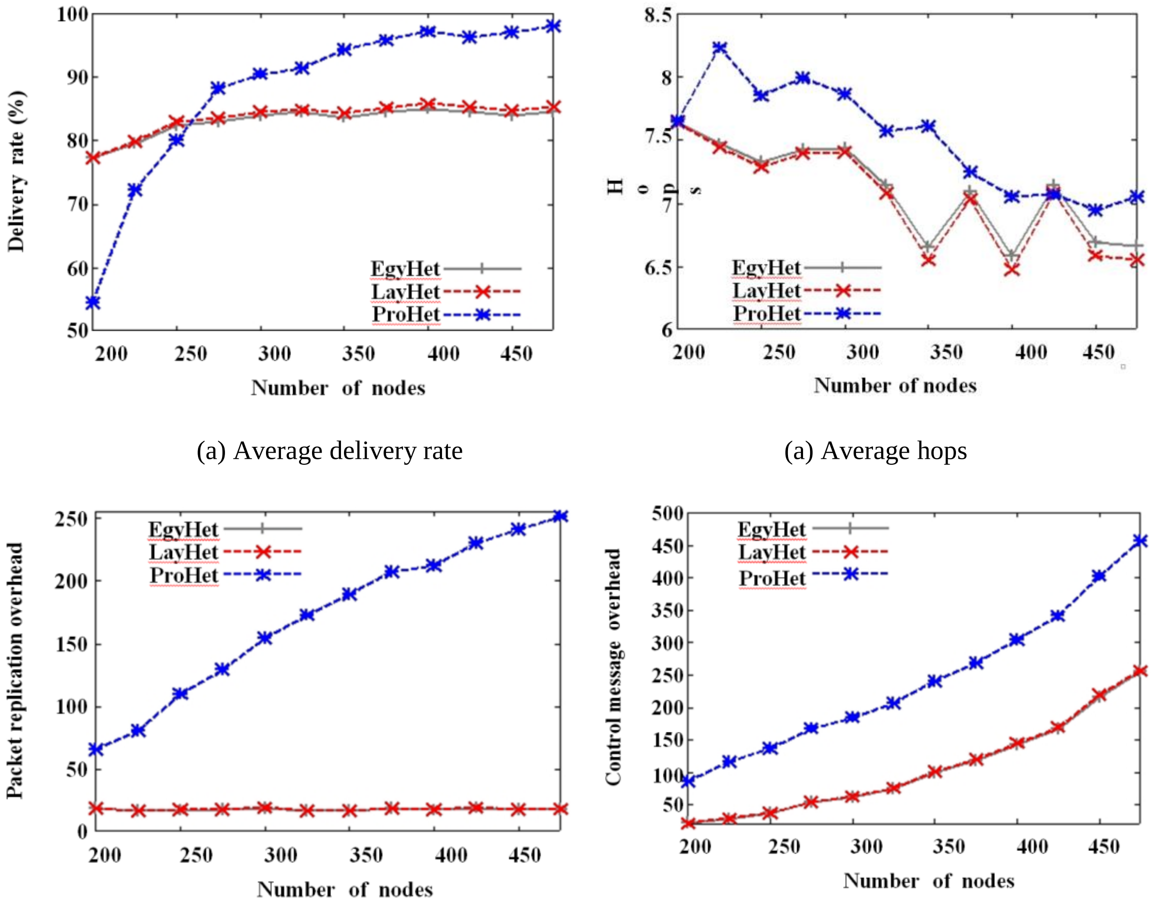
(a) Average delivery rate

(a) Average hops

(c) Average packet replication overhead

(d) Average control message overhead

Fig. 20. Comparison of ProHet, LayHet and EgyHet with assurable delivery rate 99%.



(a) Average delivery rate

(a) Average hops

(c) Average packet replication overhead

(d) Average control message overhead

Fig. 21. Comparison of ProHet, LayHet and EgyHet with assurable delivery rate 80%.

Effect on delivery rate when sensor energy is reduced

In this simulation, we want to see how the delivery rates of protocols are affected if sensor energy reduces.

1) Energy model: We assume that each node u has a finite and unreplenishable initial energy e_u, which is a non-negative integer value. For the energy consumption of sending and receiving a message by a node, we adopt the first order radio model [14] where for k-bit data over distance l, the transmission energy E_T (k, l) and the receiving energy E_R(k) are calculated as follows:

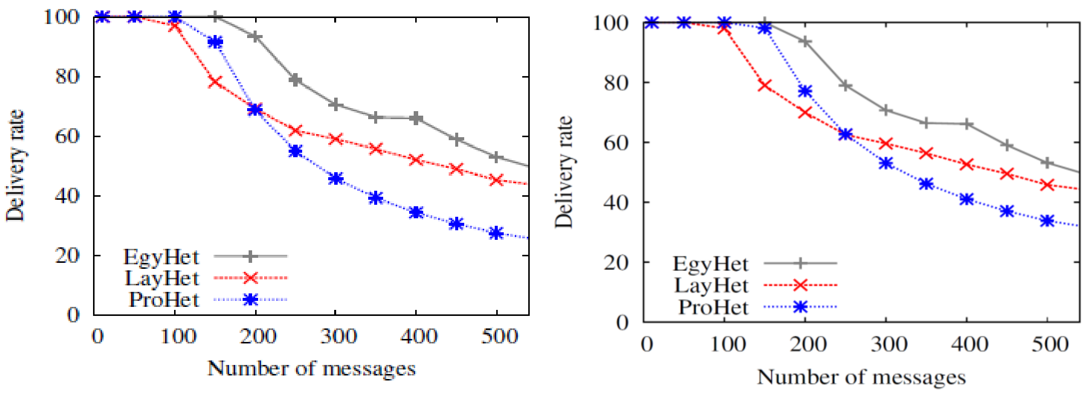
$$E_T\,(k, l) = E_{elec} \times k + \epsilon_{amp} \times k \times l^2 \qquad (6)$$

$$E_R(k) = E_{elec} \times k \qquad (7)$$

where E_{elec} = 50nJ/bit and ϵ_{amp} = 100pJ/bit/m^2. When the distances among nodes are in the order of one hundred meters, the term with ϵ_{amp} is much larger than the term with E_{elec}. Therefore, we assume that for each node, sending one unit-sized message costs one unit of energy while receiving one message costs zero energy.

2) Parameter setting: We again assume that, in our simulations, a sensor has one of the three transmission ranges 40m, 50m or 60m and a link loss rate between 0% and 20% initially. We randomly generated 30 different deployments of ASNs, set the desired delivery rate to be 99% and randomly generated the sink and the sources. We tried from 10 to 550 messages with a step of 50. In the routing process, we set the ratios of the length of a control message and that of a regular message to be 1: 25 and 1: 50. We set the initial energy level for each node to be 2000 when the ratio is 1: 25 and 4000 for each node when the ratio is 1: 50. Whenever a sensor sent a packet, some energy would be deducted from its energy level using the above energy model. The simulation results are shown in Fig. 22.

From the figures, we can see that, with the sending of the messages, the remaining energy of sensors decreases and after a certain point, the network cannot satisfy the desired delivery rate any more. Regardless of message length ratios and nodes' remaining energy levels, the delivery rate of EgyHet is better than those of LayHet and ProHet, which justifies the energy consideration in EgyHet. In Fig. 22(a), when the number of messages is between 90 and 200 and in Fig. 22(b), when the number of messages is between 90 and 250, ProHet's delivery rate is better than that of LayHet. This is because LayHet tries to use the shortest path between a source and the sink. The failure of the sensors on or close to the shortest path will make the routing more difficult. On the other hand, ProHet can use detours so that its delivery rate during this section does not decrease as much as LayHet’s. But ProHet depletes nodes’ energy faster. So eventually its delivery rate falls faster.



(a) Delivery rate with control message and regular message length ratio 1 : 25

(b) Delivery rate with control message and regular message length ratio 1 : 50

Fig. 22. Comparison of ProHet, LayHet and EgyHet’s delivery rates as sensors run out of energy.

VIII. CONCLUSION

In this thesis, we designed performance guaranteed routing protocols in asymmetric sensor networks where two end nodes may not use the same path to communicate with each other. To address the difficulty caused by the asymmetric links, we first proposed a general framework protocol RP that finds reverse paths for asymmetric links. Then we presented three efficient routing algorithms ProHet, LayHet, and EgyHet built on RP to satisfy performance requirements. Simulation results show that ProHet, LayHet, and EgyHet can reach the desired delivery rate earlier than the existing protocol and outperform it in terms of average hops, average packet replication overhead, and average control message overhead. Furthermore, LayHet and EgyHet's performance degrades more slowly than the existing one as sensors run out of their energy. In this thesis, we focused on designing efficient routing protocols on the top of the reverse path protocol RP. The study of the reverse path protocol itself and the comparison with the one proposed by Ramasubramanian and Mosse [25] can be an independent topic, which we will leave for the future work. We believe asymmetric links are very common in many wireless networks. They can be the result of the time dependency of nodes' connections such as in the case of delay tolerant networks, vehicular networks, and mobile social networks. In the future, we will study efficient routing algorithms in these wireless asymmetric networks.

REFERENCES

[1] S. Ben Alla, A. Ezzati, A. Beni Hssane and M. L. Hasnaoui, “Hierarchical Adaptive Balanced Energy Efficient Routing Protocol (HABRP) for Heterogeneous Wireless Sensor Networks”, in *International Conference on Multimedia Computing and Systems (ICMCS)*, 2011.

[2] C. Adjih, P. Jacquet and L. Viennot, “Computing Connected Dominated Sets with Multipoint Relays”, Technical Report, INRIA, Oct. 2002.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayicrci, “A Survey on Sensor Networks”, IEEE CommunicationsMagazine, Vol. 40, No. 8, 2002, pp. 102-116.

[4] C. L. Barrett, S. Eidenbenz, L. Kroc, M. V. Marathe and J. P. Smith, “Parametric Probabilistic Sensor Network Routing”, in *Proceedings of the Second ACM International Conference on Wireless Sensor Networks and Applications(WSNA)*, San Diego, California, USA, Sept. 2003, pp. 122-131.

[5] A. Behzadan, A. Anpalagan and B. Ma, “Prolong Network Lifetime via Nodal Energy Balancing in Heterogeneous Wireless Sensor Networks”, in *Proceedings of IEEE ICC*, 2011, pp. 1-5.

[6] X. Chen and J. Shen, “Improved Schemes for Power-Efficient Broadcast in Ad Hoc Networks”, International Journal of High Performance Computing and Networking, Inderscience, Vol. 4, No. 3/4, 2006, p. 198-206.

[7] X. Chen, W. Y. Qu, H. L. Ma and K. Q. Li, “A Geography-based Heterogeneous Hierarchy Routing Protocol for Wireless Sensor Networks”, in *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications(HPCC)*, Dalian, China, Sept. 2008, pp. 767-774.

[8] B. Deb, S. Bhatnagar and B. Nath, “Information Assurance in Sensor Networks”, in *Proceedings of the Second ACM International Conference on Wireless Sensor Networks and Applications(WSNA)*, San Diego, California, USA, Sept. 2003, pp. 160-168.

[9] X. Du and F. Lin, “Designing Efficient Routing Protocol for Heterogeneous Sensor Networks”, in *Proceedings of the 24th IEEE International Performance Computing and Communications Conference(IPCCC)*, Phoenix, Arizona, USA, Apr. 2005, pp. 51-58.

[10] O. Egecioglu and T. Gonzales, “Minimum-energy Broadcast in Simple Graphs with Limited Node Power,” in *Proceedings of IASTED international conference on Parallel and Distributed Computing and Systems*, 2001, pp. 334-338.

[11] B. Elbhiri, R. Saadane and D. Aboutajdine, “Stochastic Distributed Energy-Efficient Clustering (SDEEC) for Heterogeneous Wireless Sensor Networks”, ICGST International Journal on Computer Network and Internet Research, Vol. 09, Issue II, 2009, pp. 11-17.

[12] I. A. Essa, “Ubiquitous Sensing for Smart and Aware Environments”, IEEE Personal Communications, Vol. 7, 2000, pp. 47-49.

[13] D. L. Guidoni, A. Boukerche, L. A. Villas, F. S. H. Souza, R. A. F. Mini and A. A. F. Loureiro, “A Framework based on Small World Features to Design HSNs Topologies with QoS”, in *Proceedings of IEEE Symposium on Computers and Communications (ISCC)*, 2012.

[14] W. R. Heinzelman and A. Chandrakasan and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks*”,* in *Proceedings of the 33rd Hawaii International Conference on System Sciences(HICSS)*, Jan. 2008.

[15] Y. F. Hu, W. Z. Li, X. Chen, X. Chen, S. L. Lu, and J. Wu, “A Probabilistic Routing Protocol for Heterogeneous Sensor Networks”, in *Proceedings of IEEE International Conference on Networking, Architecture, and Storage (IEEE NAS),* July 2010.

[16] C. Intanagonwiwat, R. Govindan and D. Estrin, “Directed Diffusion: a Scalable and Robust Communication Paradigm for Sensor Networks”, in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking(MobiCom),* Boston, MA, USA, Aug. 2000, pp. 56-67.

[17] B. Karp and H. T. Kung, “Gpsr: Greedy Perimeter Stateless Routing for Wireless Networks”, in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom),* Boston, MA, Aug. 2000, pp. 243-254.

[18] L. Lazos, R. Poovendran, and J. A. Ritcey, “Probabilistic Detection of Mobile Targets in Heterogeneous Sensor Networks”, in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN),* Cambridge, MA, USA, Apr. 2007, pp. 519-528.

[19] A. M. Mainwaring, D. E. Culler, J. Polastre, R. Szewczyk and J. Anderson, “Wireless Sensor Networks for Habitat Monitoring”, in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA),* Atlanta, GA, USA, 2002.

[20] A. Manjeshware and D. P. Agrawal, “Teen: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks”, in *Proceedings of International Parallel and Distributed Processing Symposium*, Los Alamitos, CA, USA, Apr. 2001, pp. 2009-2015.

[21] J. Newsome and D. X. Song, “Gem: Graph Embedding for Routing and Data-centric Storage in Sensor Networks without Geographic Information”, in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys)*, Los Angeles, CA, USA, Nov. 2003, pp. 76-88.

[22] M. Perillo, Z. Cheng and W. Heinzelman, “An Analysis of Strategies for Mitigating the Sensor Network Hot Spot Problem”, in *Proceedings of IEEE MobiQuitous*, 2005, pp. 474-478.

[23] D. A. Patterson, “Rescuing Our Families, Our Neighbors, and Ourselves”, Commun. ACM, Vol. 48, No. 11, 2005, pp. 29-31.

[24] J. M. Rabaey, M. J. Ammer, J. L. da Silva Jr., D. Patel and S. Roundy, “Picoradio Supports Ad Hoc Ultra-low Power Wireless Networking”, IEEE Computer, Vol. 33, No. 7, 2000, pp. 42-48.

[25] V. Ramasubramanian and D. Mosse´, “BRA: a Bidirectional Routing Abstraction for Asymmetric Mobile Ad Hoc Networks”, IEEE/ACM Transactions on Networking(TON), Vol. 16, No. 1, 2008, pp. 116-129.

[26] A. Rao, C. H. Papadimitrious, S. Shenker and I. Stoica, “Geographic Routing Without Location Information”, Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom), San Diego, CA, USA, 2003.

[27] H. Rivas, T. Voigt and A. Dunkels, “A Simple and Efficient Method to Mitigate the Hot Spot Problem in Wireless Sensor Networks”, In Workshop on Performance Control in Wireless Sensor Networks, Coimbra, Portugal, May 2006.

[28] R. R. Rout, S. K. Ghosh and S. Chakrabarti, “A Network Coding based Probabilistic Routing Scheme for Wireless Sensor Network”, in *Proceedings of the Sixth Intertional Conference on Wireless Communication and Sensor Networks,* 2010, pp. 1-6.

[29] A. Sixsmith and N. Johnson, “A Smart Sensor to Detect the Falls of the Elderly”, IEEE Pervasive Computing, Vol. 3, 2004, pp. 42-47.

[30] D. Tian and N. D. Georganas, “Energy Efficient Routing with Guaranteed Delivery in Wireless Sensor Networks”, in *Proceedings of IEEE Wireless Communications and Networking Conference(WCNC)*, New Orleans, LA, USA, Mar. 2003, pp. 1923-1929.

[31] G. Q. Wang, Y. C. Ji, D. C. Marinescu and D. Turgut, “A Routing Protocol for Power Constrained Networks with Asymmetric Links”, in *Proceedings of the 1st ACM International Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks(PE-WASUN)*, Venezia, Italy, Oct. 2004, pp. 69-76.

[32] N. Wang and C. H. Chang, “Performance Evaluation of Geographic Probabilistic Flow-based Spreading Routing in Wireless Sensor Networks”, in *Proceedings of the 4th ACM International Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks(PE-WASUN)*, Chania, Crete Island, Greece, Oct. 2007, pp. 32-38.

[33] J. Wu, “An Enhanced Approach to Determine a Small Forward Node Set based on Multipoint Relays”, in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, Vol. 4, Oct. 2003, pp. 2774-2777.

[34] Networks”, 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Miami, FL, USA, Mar. 2005, pp. 878-890.

[35] F. Ye, H. Y. Luo, J. Cheng, S. W. Lu and L. X. Zhang, “A Two-tier Data Dissemination Model for Large-Scale Wireless Sensor Networks”, in *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking (MOBICOM)*, Atlanta, GA, USA, Sept. 2002, pp. 148-159.

[36] Q. Zhang and W. G. Chang, “A Power Efficiency Routing Protocol for Heterogeneous Sensor Networks”, 4th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM), Dalian, China, Oct. 2008, pp. 1-4.

[37] W. Y. Zhang, X. J. Du, J. Wu, S. D. Soysa, Y. Liu, “Near-Minimum-Energy Routing in Heterogeneous Wireless Sensor Networks”, in *Proceedings of IEEE GLOBECOM*, 2010, pp. 1-5.

[38] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. An empirical study of epidemic algorithms in large scale multihop wireless networks. Technical Report IRB-TR-02-003, Intel Research, 2002.

[39] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. In Proceedings of ACMASPLOS, 2002.

[40] M. K. Marina and S. R. Das. Routing performance in the presence of unidirectional links in multi-hop wireless networks. In *Proceedings of Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, June 2002.

[41] V. Ramasubramanian and D. Mosse. Bra: a bidirectional routing abstraction for asymmetric mobile adhoc networks. IEEE/ACM Transaction on Networking(TON), 16(1):116-129, 2008.

[42] Notification system. http://en.wikipedia.org/wiki/Notification system.

VITA

Zanxun Dai is a Master candidate in Computer Science at Texas State University-San Marcos, Texas. He is working under the supervision of Dr. Xiao Chen and Dr. Hongchi Shi. His research area is wireless sensor networks. He entered the Graduate College of Texas State in August, 2010. Before that, he received his Bachelor’s degree of Software Engineering from Harbin Institute of Technology, China in 2006.

Permanent Address: 1570 West Pond Drive
Apartment #20
Okemos, MI 48864

This thesis was typed by Zanxun Dai.