A QoS Based Routing Protocol for Wireless Sensor Networks

Mirela Fonoage, Mihaela Cardei, and Arny Ambrose
Department of Computer and Electrical Engineering and Computer Science
Florida Atlantic University
Boca Raton, FL 33431, USA
E-mail: {mmarta@, mihaela@cse., aambrose@}fau.edu

***Abstract*—In this paper we propose a QoS based routing protocol for wireless sensor network applications that support both periodic and event-based data reporting. A geographic routing mechanism combined with QoS support is used to forward packets in the network. Data is routed based on the packet type. To route packets with different priorities, multiple transmission queues are used. In choosing the next hop, the node that is closer to the sink, has high residual energy, high link quality, and low load is selected. Congestion control is achieved by using a ring or barrier mechanism that captures and aggregates messages that report the same event to the same sink. We present the main operations of the barrier mechanism, including barrier formation, repair, enlarge, shrink, and termination. Simulation results using JIST/SWANS simulator show the performance of our routing protocol compared with other related works.**

Keywords: wireless sensor networks, energy efficiency, routing, quality of service, congestion.

I. Introduction

A Wireless Sensor Network (WSN) is composed of a large number of small devices with limited power, processing and communication capabilities, that are densely deployed inside a phenomenon or very close to it [1]. Sensor nodes have two main functionalities: monitor the environment and send the sensed data to a special node, called the sink. Sensor nodes can send the monitored data periodically (periodic data reporting) or when an event occurs (event-based reporting). Some applications (e.g. forest fire monitoring) need a mixture of both periodic and event-based data reporting. In this case, each sensor node monitors the environment and besides sending periodical measurements to the sink, it also informs the sink when a specific event occurs.

There are multiple types of packets that flow through a WSN for a mixed data reporting application. Periodical data reporting and event-based packets are the two main packet types. The event-based packets usually alert the sink when a critical event occurs, and therefore these packets have to be transmitted as soon as possible, with higher priority than the periodic data reporting.

Many routing protocols for WSNs have been proposed in the literature, but there are some challenges that have not been resolved yet. One of them is integrating Quality of Service (QoS) requirements in the routing protocols for mixed data reporting applications. Due to the dynamic nature of the network, the existing QoS protocols for wired networks can not be applied directly to WSNs. Congestion control mechanisms are essential in WSNs.

Event occurrence in a monitored environment entails a high number of event messages that travel in the network toward the sink. Sensor nodes end up delaying or even dropping packets due to the limited queuing memory capacity. This is undesirable in an event-based data reporting, where timely data delivery is critical. Therefore, congestion control mechanism need to be integrated in the routing protocol to avoid packet loss.

In this paper we introduce a QoS routing protocol with a congestion control mechanism for mixed data reporting WSNs. The congestion control mechanism is targeting the event packets specifically, since the reporting interval is smaller compared to the periodic reporting. Allowing a large number of similar packets to flow in the network may trigger its congestion and a reduction in the network lifetime.

II. Related Work

Directed Diffusion [5] is one of the first routing protocols for WSNs, based on a data centric and application aware paradigm that tries to reduce the number of messages sent in the network using a data aggregation technique. The protocol finds paths from multiple sources to a destination (sink) and introduces data aggregation to reduce the number of messages flowing in the network. The sink sends interests in the network, representing queries. During the interest dissemination, gradients are being set-up by the sensor nodes. These gradients form multiple paths which are being used to transmit data from sources to the sink. Directed Diffusion reduces the energy consumption by choosing empirically good paths and by using data aggregation techniques (caching and processing data at each sensor node).

Stankovic et. al. [4] introduce SPEED, a real-time communication protocol. SPEED is a stateless localized algorithm that provides light real-time end-to-end guarantees. The end-to-end communications are achieved by maintaining a desired delivery speed across the sensor network using a feedback control and non-deterministic geographic forwarding mechanisms. SPEED contains a congestion control mechanism and provides real-time communication service. The end-to-end delay is proportional with the distance between the source and the destination because each sensor node estimates the required speed for a certain delay taking into account its distance to the sink. In routing, the next hop is chosen such that the speed requirement is met.

Threshold Sensitive Energy Efficient Sensor Network Protocol (TEEN) [9] targets reactive networks. Sensor nodes are divided into clusters and each node monitors the environment and sends the data to the cluster head when certain conditions are met. The number of transmissions is reduced by introducing a range of interest, identified by a hard and a soft threshold, for a certain sensing attribute. A sensor node transmits its sensed data only when the reading is in this range. The range is disseminated by cluster heads. A drawback of TEEN is that a sensor node may never transmit its readings if the sensed information is not within the required range.

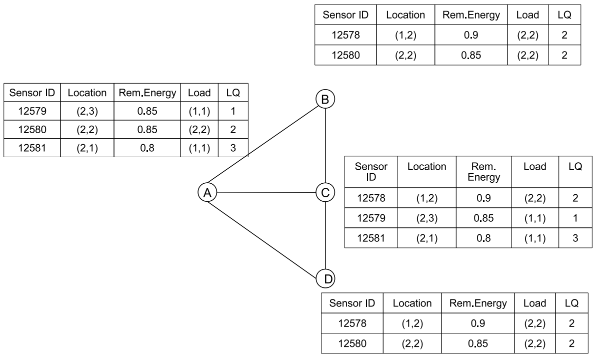
APTEEN [10] is a variant of TEEN which introduces periodic transmissions of sensed data to the sink. APTEEN is based on a query system which allows three types of queries: historical, on-time, and persistent which can be used in a hybrid network. QoS requirements are introduced for the on-time queries. Minimum delay is achieved by introducing a special time slot assignment to each node in a cluster using a TDMA schedule.

Location Aided Routing (LAR) [7] uses location information of sensor nodes to route packets in the network. The authors limit the search for a path to the sink by choosing a forwarding node from a *request zone* which represents a smaller part of the network. A node forwards a route request message only if it is part of the *request zone*. The challenge here is to determine the right *request zone*. The authors introduce a rectangular shape for the *request zone*, but other shapes are possible as well. The authors also discuss the possibility of adapting the *request zone* to further improve packet routing.

III. Problem Definition and Network Model

A. Problem Definition

In this paper we focus on applications that must support both periodic and event-based data reporting.The most challenging to achieve, with regard to delay, is the event-based data reporting since it must reach the users as soon as possible. More specifically, we are proposing a routing protocol that performs data forwarding according to the packet type. Periodic reporting mechanisms involve periodic transmissions of sensed data to a sink node. On the other hand, event-based mechanisms involve occasional data reporting when some critical condition is met. If a large number of sensors in the event area detect the event and start reporting to the sink, congestion may occur, therefore this issue has to be addressed.



Sensor ID	Location	Rem.Energy	Load	LQ
12578	(1,2)	0.9	(2,2)	2
12580	(2,2)	0.85	(2,2)	2

Sensor ID	Location	Rem.Energy	Load	LQ
12579	(2,3)	0.85	(1,1)	1
12580	(2,2)	0.85	(2,2)	2
12581	(2,1)	0.8	(1,1)	3

Sensor ID	Location	Rem. Energy	Load	LQ
12578	(1,2)	0.9	(2,2)	2
12579	(2,3)	0.85	(1,1)	1
12581	(2,1)	0.8	(1,1)	3

Sensor ID	Location	Rem.Energy	Load	LQ
12578	(1,2)	0.9	(2,2)	2
12580	(2,2)	0.85	(2,2)	2

Fig. 1. Routing example

We consider a WSN consisting of N sensor nodes which have the following characteristics: they monitor the environment, periodically send sensed data to a sink, and they alert the sink in case a certain event occurs. An example of such an application is forest fire monitoring. Periodically, sensed data such as temperature, humidity, and smoke level are sent to the sink. If the *composite event* fire is detected by a sensor, then an alert is sent to the sink. For example, the event fire can be defined as $fire = (temperature > th_1) \wedge (light > th_2) \wedge (smoke > th_3)$, where th_1, th_2, and th_3 are some predefined thresholds.

Event-based and periodic data reporting has different QoS requirements. Event-based reportings must be sent with a higher priority than the periodic data reporting messages. The problem that we address in this paper is:

Given a WSN consisting of N sensor nodes randomly deployed in an area, design a communication protocol for the sensor nodes such that the network lifetime is maximized and the communication complies with the QoS requirements.

Each sensor node must route the traffic according to the following QoS requirements:

1) Route traffic according to the priority level.
2) Minimize the delay to relay event-based packets.
3) Ensure the delivery of event alerts.
4) Deliver periodic data reporting packets using a best effort policy.

The routing protocol must prevent congestion and must be energy-efficient such that to prolong network lifetime.

B. Network Model and Network Formation

In this paper, we consider two types of nodes: sensor nodes and sink node(s). A sink has no limitations in terms of power, computation, and storage capacities, while the regular sensor nodes are resource constrained. For example, the Crossbow MICAz mote [3] operates on the 2.4GHz ISM band, uses two AA batteries, and has an ATmega 128L processor at 8 MHz, 4 Kbytes RAM, and a transmission range up to 30 m (indoor) /100 m (outdoor).

Sensor nodes are randomly deployed in the interest area and during the initialization phase discover their neighbors using *Hello* messages. We assume sensors are deployed densely enough such that the sensor network is connected. We consider each sink node is within communication range of at least one sensor. A sink broadcasts a request to the sensor network containing the following information: (i) sink location, (ii) area of interest, (iii) reporting type: periodic and/or event-based, (iv) periodic reporting attributes, reporting duration, and reporting interval, (v) event(s) description (e.g. using predicates), reporting duration, and reporting interval.

The first parameter in the request is the sink location, which will be used as a destination for data reporting. The area of interest can be described using two coordinates (e.g. low-left and up-right corners) for a rectangular area, or using the center and radius for a circular area. A sink can request periodic and/or event-based reporting. For periodic reporting, attributes such as temperature, humidity, etc. are specified, as well as the reporting duration, and the reporting interval. The *reporting duration* specifies the start and end time, between which sensed data will be reported. The *reporting interval* specifies how frequently data will be reported, that means every reporting interval sensed data will be sent toward the requesting sink. For event-based reporting, an event can be specified using predicates, e.g. $fire = (temperature > th_1) \wedge (light > th_2) \wedge (smoke > th_3)$. In addition, the sink has to specify the reporting interval.

One or more sinks can issue such requests. The requests are broadcasted in the monitoring area. Each sensor in the area of interest that receives the request starts sensing the requested attributes. Each time interval, the sensor node transmits its sensed data to the sink, according to the request: periodic reporting and/or event-based.

We assume that sensor nodes know their location. One way to accomplish this is to have some of the sensors equipped with GPS, while the other sensors compute their location using a triangulation mechanism [2].

In the next section, we present our routing protocol that delivers messages according to their priority (defined depending on the data report type) and which prevents congestion.

IV. QoS-based Routing Protocol

In this section, we propose a QoS-based routing protocol that routes traffic according to the type of data. We consider that each data reporting packet has a priority class: (i) class1: event-based reports, with higher priority, and (ii) class2: periodic reports, with lower priority. The mechanism can be extended to an arbitrary number of classes, but in our discussion we consider two classes.

It is critical that class1 packets reach the sink as soon as possible since they are alerts announcing critical events. One important issue that we need to address is congestion. Assume that a fire event takes place. Then the number of sensors in the fire area will detect the event and start sending events to the sink. Event-based reportings are expected to be more frequent than the periodic reportings. If all that traffic flow toward the same sink (see Figure 2a), then congestion occurs. As a result, nodes start dropping packets and they also consume energy at a higher rate. Our routing protocol proposes a mechanism to prevent congestion due to event-based reportings.

After a request has been broadcasted by a sink S^i, the following information is stored by each node in the area of interest, in its sink request table: (i) sink id S^i_{id}, (ii) sink location S^i_{loc}, and for each request the following information: (i) reporting type (periodic or event-based), (ii) for periodic reporting: attributes, and for event-based reporting: event description (e.g. using predicates), (iii) reporting duration, and (iv) reporting interval.

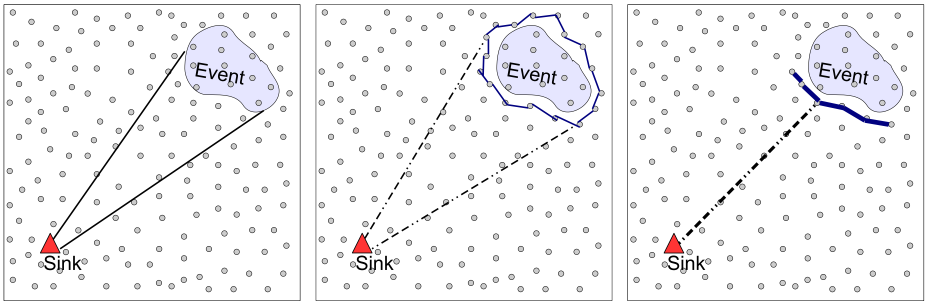
A. Routing of the sensed data

In order to perform the routing task, each sensor node maintains a neighbor table with information about its neighbors. Each sensor node sends periodically a *beacon* message, with the following information: sensor id, sensor location, remaining energy, and current load. An entry in the neighbor table contains all these information and in addition the link quality based on the received signal strength. Besides sensor nodes, a neighbor table may contain one or more sinks if they are in range.

Each data reporting packet contains the following fields: (i) source sensor id, (ii) source sensor location, (iii) sink id, (iv) sink location, (v) next hop, (vi) reporting type, (vii) reporting attributes/event, and (viii) reporting time. The last attribute contains the time when data has been recorded by the source sensor.

To perform packet routing, we use a geographic routing mechanism combined with the QoS support. Geographic routing has been used before in [7], [4]. When selecting the next hop, the goal is to find a sensor node that is close to the sink, has high residual energy, high link quality, and a low load.

The decision of the next hop is based on the information in the neighbor table. Such an approach is scalable,



(a) Trajectory of event reporting messages. (b) Using a ring of sensors to capture and aggregate event messages (c) Using a barrier of sensors to capture and aggregate event messages

Fig. 2. Congestion prevention mechanisms.

since it does not require storing paths to the sinks.

To decide the next hop of a data reporting packet, we define a weighted function as follows:

Definition *The weighted function $\mathcal{F}$ is defined as $\mathcal{F}$ = $w_1 \times \frac{1}{dist} + w_2 \times LQ + w_3 \times E_{rem} + w_4 \times \frac{1}{Load}$, where w_i are weights that indicate the importance of each parameter in selecting the next hop, and $w_1 + w_2 + w_3 + w_4 = 1$.*

To decide the next hop, function $\mathcal{F}$ is computed for each neighbor in the neighbor table, and the one with the maximum $\mathcal{F}$ value is selected as the next hop. In computing $\mathcal{F}$, the following parameters of the candidate node are used: *dist* represents the distance between the candidate neighbor and the sink, *LQ* is the link quality between the current node and the candidate neighbor, E_{rem} is the candidate neighbor's remaining energy, and *Load* is the candidate neighbor's load.

The weight of each characteristic depends on the type of application and can be set-up according to the type of packet (periodic reporting or event-based reporting). For example, for a fire detection application, event-based reporting, we could set-up $w_1 = 0.3$, $w_2 = 0.2$, $w_3 = 0.2$, and $w_4 = 0.3$, since the message has to be forwarded to the sink as fast as possible.

Let us consider the scenario in Figure 1, where sensor A has a message to relay to sink. As illustrated in the figure, each node has a neighbor table containing the following fields for each neighbor: sensor id, sensor location, remaining energy, current load, and link quality. Node A computes function $\mathcal{F}$ for each neighbor in its routing table and decides which one will be the next hop on the way to the sink.

Assuming $w_1 = 0.3$, $w_2 = 0.2$, $w_3 = 0.2$, $w_4 = 0.3$ and sink location $(3, 4)$, node A computes function $\mathcal{F}$ for each of its neighbors: $\mathcal{F_B}$ = 0.68, $\mathcal{F_C}$ = 0.73, and $\mathcal{F_D}$ = 1.07. Node A then chooses node D to relay the message to the sink, as it has the highest value for $\mathcal{F}$.

To route packets of different priority classes, we use multiple transmission queues for different priority levels. Assuming two packet classes, we can use two priority queues. Packets are being forwarded according to an input parameter, the queue serving rate. For example, $\alpha : 1$ rate means that the node will send α packets from the first queue (class1 packets) and 1 packet from the second queue (class2 packets), then it repeats in a round-robin fashion.

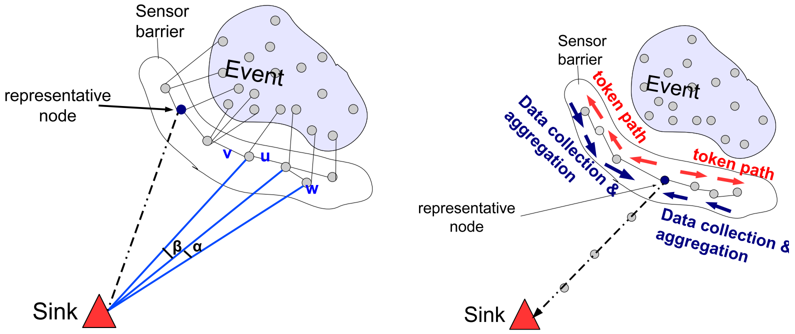
Algorithm *Routing* presented next is executed by all the sensor nodes in the network.

Routing()

```
while (1) do
  if at least one queue $\neq$ empty then
    dequeue the next message based on the serving rate
    find the next hop based on $\mathcal{F}$ and forward the message.
  end if
  if packet received then
    if packet of type beacon then
      update the neighbor table
    else if packet of type class1/class2 data reporting then
      enqueue the packet to the corresponding queue
    end if
  end if
end while
```

B. Congestion prevention mechanism

One of the major communication concerns that we have to address is congestion. Referring back to our forest fire application, if a fire starts, then all the sensors in the affected area will start sending fire events to the sink. Thus, a large flow of messages will follow a trajectory toward the sink, see Figure 2a. As a result, some of the nodes, especially those closest to the sink, will start depleting their energy at a faster rate, and



(a) Barrier formation.

(b) Data collection.

Fig. 3. Congestion prevention using a barrier mechanisms.

over time will die, partitioning the network. Another effect due to congestion is that queues of the forwarding sensors become full, thus nodes start dropping packets.

To deal with congestion, we propose a mechanism to capture and aggregate messages that report the same event to the same sink. Our idea is to use a ring or barrier of sensors, see Figure 2, that captures event reporting messages, aggregates, and forwards them to the sink. The ring/barrier must be adaptive, that means it should enlarge/shrink as the event area becomes larger or smaller.

The basic functional concepts are as follows. The nodes in the ring/barrier should be able to communicate with each other, and will elect representative(s) in charge with sending the aggregate data to the sink. The representative changes over time to balance energy consumption. To increase reliability, aggregate data can be sent over multiple paths (e.g. 2-3 paths) to the sink. The number of paths chosen is an input parameter in the protocol configuration. The main operations supported are:

- ring/barrier formation
- choose representative(s)
- data aggregation and forwarding mechanism
- ring/barrier repair
- ring/barrier enlarge/shrink
- ring/barrier termination

Next, we will detail the barrier mechanism with one representative. Other variations can be proposed as well starting from this idea.

1) Barrier formation mechanism: The barrier is formed by the "border" sensors, that delimit the event area, see Figure 2c. Each barrier node is a node that does not detect the event itself, but has at least one event-detecting neighbor. In addition, the barrier nodes must form a connected topology.

A node detecting an event E, sends periodically an *event detection* message, containing the event id and sensor node id (and/or sensor location). For energy-efficiency purpose, *event detection* messages could be combined with *beacon* messages, which are being transmitted periodically anyways. A sensor node receiving *event detection*, (i) does nothing if it detects the event E as well, and (ii) becomes a barrier sensor if it does not detect the event and if its distance to the sink is smaller than that of the sending node.

Next step is to ensure that the barrier nodes are connected and that each barrier node selects two neighbors, one clockwise and another counter clockwise. The ending nodes will have only one neighbor. Each barrier node broadcasts a *neighbor discovery* message with $TTL = 1$. Let us take a barrier sensor node u, see Figure 3a. Based on the messages received, it tries to establish its neighbors. In this case, they are nodes w (clockwise direction) and node v (counter clockwise direction), selected such that the angles α and β formed with the sink are minimized.

Since the network is densely deployed, we expect that neighbors are established at this step. If two barrier nodes are not connected through a path of barrier nodes, then the drawback is that multiple smaller barriers are formed. On way to get around this issue, is as follows. Assume a node u could not identify a clockwise neighbor. Then u could resend the *neighbor discovery* message in the clockwise direction, with an increased TTL, let's say $TTL = 3$. If at least one barrier neighbor is discovered within the 3-hop neighborhood, then the "closest one" in the clockwise direction is selected and the intermediate nodes become "relay nodes" in the barrier.

2) Electing the representative: Once the barrier is formed, and each node establishes its neighbors, a representative is selected. Initially, the representative can be the barrier node with the smallest id. This can be done using a distributed leader election algorithm similar to

LCR [8]. Basically, each barrier node sends a leader message with its id to both its neighbors. A node u receiving a leader message from one neighbor, will forward the message to the other neighbor if and only if the id in the message is smaller than u's id. If a non-end barrier node receives its id back from both right and left neighbors, then it becomes the leader (or representative). An ending barrier node is sufficient to receive its id back from one side only. Once a representative is chosen, it will broadcast in both directions a message, announcing its status.

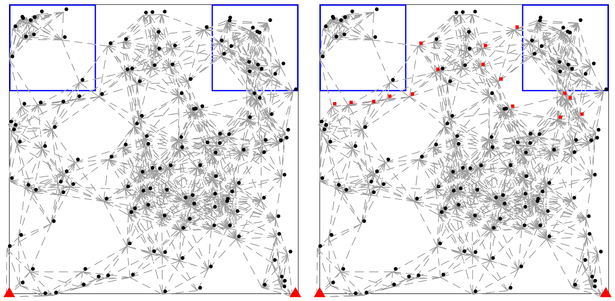
The representative is in charge with collecting data and sending it to the sink, so it is expected to consume more energy. Therefore we propose that the representative role to be changed from time to time to balance the energy consumption.

One way to do this, is that after a number of transmissions, the representative gives this role to one of its neighbors. Let's say node u is the first representative. After a number of transmissions, it gives this role to its clockwise neighbor. Therefore, the representative role moves clockwise until reaches the ending node in the barrier, then propagates counter clockwise, and so on.

3) Data aggregation and forwarding mechanism: Nodes in the barrier receiving event reporting messages, store them locally. At the end of each reporting interval, the representative sends two tokens in each direction, clockwise and counterclockwise. Note that if the representative is an ending node in the barrier, then it sends only one token in one direction. When a token reaches an end, then it propagates back to the representative. While it propagates back, it performs data collection and aggregation. Each node on the path aggregate its own data with the received message. To be able to perform the aggregation, the same event must have been reported in the same reporting interval. Note from the section IV-A that the first two fields in a data reporting message are the source sensor id and its location.

The list of sensors is appended in the message traveling toward the representative. The representative then aggregates this list of sensors into an area enclosing those sensors. Then only that area is specified when data is reported to the sink. One method is to take the two farthest away points and to report the circle with diameter defined by those points. A circle would require only the center and radius to be transmitted. Another solution is to transmit several important points on the polygon surrounding the reporting sensors; for example the east, west, north, and south most locations.

Once the aggregation is performed by the representative, a report is sent to the sink. Again, the first two fields in the report will contain in this case an area specification.



(a) SPEED Basic (b) QoS Routing

Fig. 4. Network organization for 150 nodes

4) Barrier repair mechanism: This case can occur due to node failures or when sensors run out of energy. Every reporting interval, a barrier node should receive a token from the representative. If no token is received, then it may be the case that one or more of the sensors are dead. In that case, the node checks its neighbor and if it is dead then sends a *neighbor discovery* message with a small TTL (e.g. $TTL = 3$) and elects a new neighbor as described in section IV-B1.

5) Barrier enlarge/shrink mechanism: Referring back to the fire detection application, assume that the fire area become larger or smaller. Then the sensor barrier should become larger/smaller accordingly.

Enlarging the barrier: assume that a node that is currently in the barrier, starts detecting the event itself. Then, similar to section IV-B1, it starts sending *event detection* messages. New nodes become barrier nodes, and they establish connectivity with the other barrier nodes using *neighbor discovery messages*.

Shrinking the barrier: consider now the case that a fire starts diminishing. Some of the nodes will not detect the event anymore, so they will stop sending *event detection* messages. A barrier node that does not receive any *event detection* messages for a number of rounds (input parameter) leaves the barrier and announces this information to its neighbors using a *leave barrier* message. When one or more nodes leave the barrier, new nodes may join the barrier based on the same criteria as in IV-B1. Connectivity is achieved using *neighbor discovery* messages, as discussed before.

As part of the barrier enlarging/shrinking, it may happen that the barrier is partitioned into two or more barriers. If no token is received during few reporting intervals, then a new representative has to be chosen, using the mechanism in IV-B2.

6) Barrier termination mechanism: This refers to case when the event is not detected anymore, for example the fire is extinguished. Barrier termination follows the same idea as in the barrier shrinking: nodes stop sending *event detection* messages, and as a result no node will be part of the barrier anymore.

V. Simulation

In this section, we analyze and compare the performance of the QoS routing protocol proposed versus SPEED [4]. We performed simulations for the two algorithms using an application built on top of JistSwans platform [6]. The sensors are randomly deployed in an area of 2000×2000 meters. Sensor communication range is 110 meters. All sensor nodes have the same capabilities, with initial energy of 1 Joule. Sinks have unlimited energy.

Node positions are maintained the same for both algorithms per run, but they are randomly deployed in the next run.

The performance metrics examined in our simulations are the following:

- *Received Packets Ratio*: the number of successfully received packets over the total number of packets sent in the network.
- *Consumed Energy*: computed as the total amount of energy consumed in the network over the number of nodes.
- *Total Number of Packets*: represents the total number of packets sent and received in the network including control packets.
- *Total Number of Event Packets*: represents the total number of received and sent event packets in the network.
- *Mean Packet Delivery Delay*: the delay of a successfully received packet.
- *Dropped Packets*: represents the number of packets dropped.

The results are averaged over 10 runs, and one run has a duration of 3600 Jist/Swans seconds. The scenario used in the simulations is described next. Two sinks, each with unlimited capabilities, are positioned at the lower corners of the area, see Figure 4a. Two events take place in the upper corners of the area (blue rectangles in Figure 4). The event in the upper left corner is reported to the sink in the bottom right corner and the event in the upper right corner is reported to the sink in the bottom left corner. Figure 4a illustrates the sensor network for 150 nodes. Figure 4b shows the same network after our QoS Routing algorithm is run. Barrier nodes are represented by red squares.

All sensor nodes send periodic reporting messages at 90 second intervals and event messages at 60 second intervals. A beacon message is sent every 30 seconds. Barrier related control messages are sent when the barrier is built, and every 120 seconds after that. An event is set to occur 60 seconds after the start of the simulation and will be on until the end of the simulation.

The simulation is organized in rounds. Each round has 30 seconds. The nodes send at least one beacon message per round. Not every round a reading or an event is sent. The event messages are sent only when an event occurs. Beacon, sensor readings, and event packets have 24 bytes, while barrier control messages have 12 bytes. For our protocol, the barrier leader sends an aggregated message event to the sink containing the area where the event has occurred. We consider that a disk area will be reported, therefore the center and radius will be included in the message, such that a size of 24 bytes can be achieved.

The barrier is re-constructed every other event interval such that the changes in the network can be taken into account for the barrier. We are sending beacon messages every period, but the barrier messages and the beacon messages can be combined into one message, and the network lifetime will be further improved.

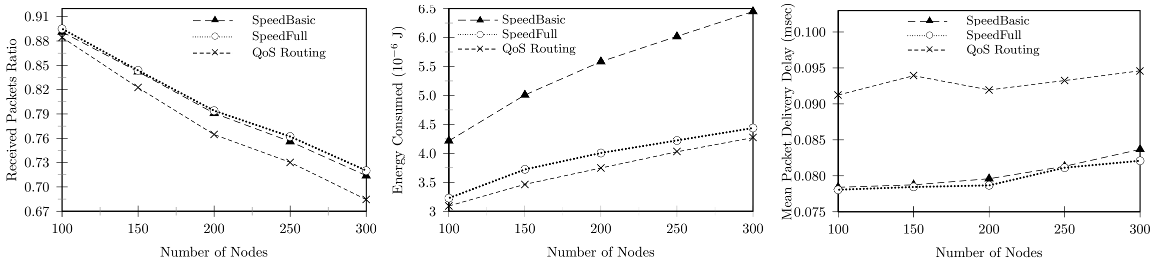
The proposed protocol is compared with SpeedBasic and SpeedFull. SpeedBasic is a variation of the SPEED protocol without the congestion control mechanism. SpeedFull is the actual SPEED protocol, where the congestion mechanism is used.

The performance results are illustrated in Figures 5 and 6. In Figure 5a, the received packet ratio of our QoS routing algorithm is lower than the two other algorithms because we have introduced a small delay due to the barrier configuration which will determine some of the packets to be dropped since the speed condition is not achieved. Overall, the received packet ratio decreases with the increase in the number of nodes because for a higher number of packets that need to be sent there will be more contentions for the medium and more collisions occurring.

Figure 5b illustrates the energy consumed per node. Our QoS routing algorithm consumes the least power among the three algorithms compared. This is because our algorithm aggregates similar packets and sends less reporting messages to the sinks.

The results for mean packet delivery delay (see Figure 5c) show that even in the worst case our protocol introduces a delay of only 11 nanoseconds, which represents the tradeoff for reducing the number of event messages sent in the network. QoS routing protocol incurs additional delay when the barrier captures event messages, aggregates them, and sends them to the sink.

Figure 6a shows the total number of packets that traveled in the network. Our algorithm introduces additional control packets when dealing with the barrier (which are counted toward the total), so it is expected to have a higher number of packets versus the SPEED protocol. The total number of messages sent can be further reduced if we combine the beacon messages with the barrier related messages.

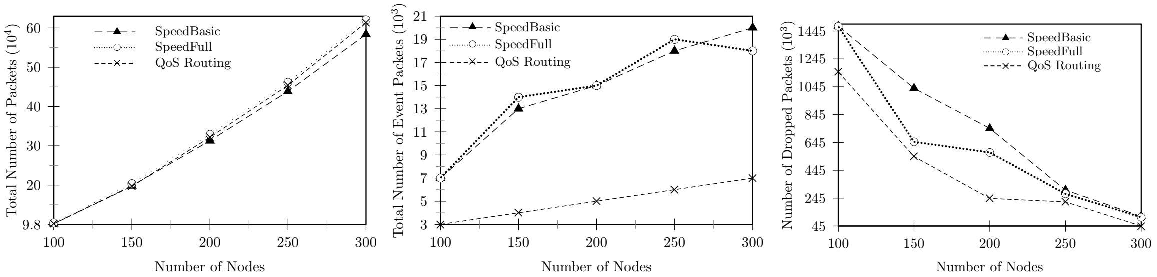


(a) Received packets ratio

(b) Average energy consumed per node

(c) Mean packet delivery delay

Fig. 5. Simulation results



(a) Total packets

(b) Total event packets

(c) Total dropped packets

Fig. 6. Simulation results

Figure 6b shows the total number of event packets sent and received in the network. Our protocol has a considerable lower number of event messages traveling in the network. The total number of event packets increases with the number of nodes because more nodes will be in the event area, transmitting events. Increasing the number of nodes triggers a more linear increase in the number of event packets sent for QoS Routing compared with SpeedBasic and SpeedFull because of the aggregation done by the barrier nodes.

The total number of dropped packets is examined next. Figure 6c shows that our QoS routing protocol has the least number of dropped packets. One reason is the lower number of packets traveling in the network. The number of dropped packets decreases with an increase in the number of nodes because the number of neighbors per node increases, making it easier to find a next hop that satisfies the speed condition.

VI. Conclusions

In this paper we proposed a QoS, location-based routing protocol for WSN applications that supports both periodic and event-based reportings. The protocol uses a barrier-based congestion control mechanism that allows efficient data aggregation and avoids energy depletion due to heavy reportings. Simulation results using JIST/SWANS simulator show the performance of our protocol compared to other related works.

Acknowledgments

This work was supported in part by NSF grant CCF 0545488.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, Wireless Sensor Networks: a Survey, *Computer Networks*, Vol. 38, No. 4, pp. 393-422, 2002.

[2] X. Cheng, A. Thaeler, G. Xue, and D. Chen, TPS: A Time-Based Positioning Scheme for Outdoor Wireless Sensor Networks, *IEEE INFOCOM*, 2004.

[3] Crossbow Technology, http://www.xbow.com, last accessed on May 28, 2010.

[4] T. He, J. A. Stankovic, C. Lu, T. Abdelzaher, SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks, *ICDCS*, 2003.

[5] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, Directed Diffusion for Wireless Sensor Networking, *Proceedings of MobiCom'00*, August 2000.

[6] JIST/SWANS Scalable Wireless Ad hoc Network Simulator, http://jist.ece.cornell.edu/

[7] Y.-B. Ko and N. H. Vaidya, Location-Aided Routing (LAR) in Mobile Ad Hoc Networks, *Wireless Networks*, Vol. 6, pp. 307-321, 2000.

[8] N. A. Lynch, *Distributed Algorithms*, Morgan Kaufmann Publishers Inc., 1996.

[9] A. Manjeshwar and D. P. Agrawal, TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks, *1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, April 2001.

[10] A. Manjeshwar and D. P. Agrawal, APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks, *International Proceedings of Parallel and Distributed Processing Symposium (IPDPS)*, 2002.