EVENT BASED FAIRNESS IN WIRELESS SENSOR NETWORKS

By

Arden DERTAT

Submitted to the Department of Computer Engineering
in partial fulfillment of the requirements
for the degree of
Bachelor of Science
in
Computer Engineering

Boğaziçi University
June 2010

TABLE OF CONTENTS

ABSTRACT

Project Name : Event Based Fairness in Wireless Sensor Networks

Term : 2009/2010 Spring

Keywords : Wireless, Sensor, Network, Fairness

Summary : Wireless Sensor Networks (WSNs) are becoming more widespread and being used in numerous applications ranging from environmental to border surveillance. As the cost of the nodes and deployment drops within time, WSNs will become ubiquitous. Healthcare is one of the major fields that we can make use of WSNs. In this work, Wireless Multimedia Sensor Networks (WMSNs) will be used to carry event based data in a multihop manner.

In this work, two scheduling schemes, namely First Come First Served (FCFS) and Event Based Fairness – Least Attained Service (EBF-LAS) will be compared. An event to be detected with good quality is an important factor in WSNs. EBF-LAS is a fair queuing algorithm that maximizes overall visual information by increasing the minimum number of packets received from each event.

FCFS and EBF-LAS scheduling schemes will be compared by making experiments on real sensor nodes. An image will be sent over the multihop sensor network. The packets will be gathered at sink, and the quality of the images will be compared. Several experiments will be conducted to compare the schemes under different conditions. The results will be demonstrated visually.

LIST OF TERMS

TinyOS	:	Operating system designed for motes.
WSN	:	Wireless Sensor Network.
Sink Node	:	The node in a WSN which has the duty to collect the data coming from all the other nodes.
Sensor Node	:	The node in a WSN which has the duty of sensing the environment and sending this information to the sink node.
FCFS	:	First Come First Served
EBF-LAS	:	Event Based Fairness – Least Attained Service

1. INTRODUCTION AND RELATED WORK

Wireless Sensor Networks sense a certain environment with the deployment of the sensor nodes within the desired area. The sensing being whether periodic or event based depends on the needs of the application. This work is based on an event based healthcare monitoring application.

Wireless Sensor Networks are an active research field. Limited capacity of processing, low bandwidth, and energy constraints convert WSNs to a challenging area. Thus, much research has been conducted in this field ranging from MAC Protocols to Routing Algorithms and Topology Design. Hence, producing a novel approach is not so trivial.

The majority of the researches conducted in this area use computer simulations to evaluate the techniques they offer, instead of real world sensor deployments. The simulations performed on sensor nodes are more realistic because generally the results of the computer simulations can’t reflect the actual behavior of a real network accurately. This work compares two packet scheduling schemes on real sensors.

2. TOPOLOGY

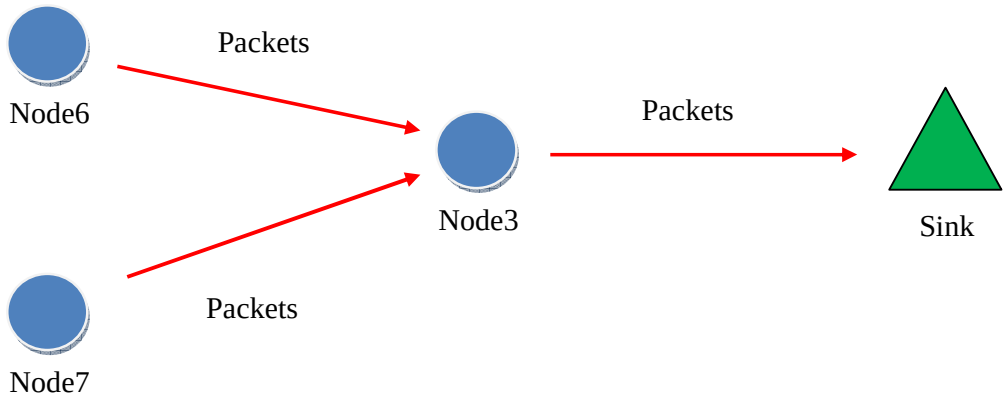


Figure 2.1. Topology of the Wireless Sensor Network (WSN)

This topology will be used in the majority of the experiments. Node6 is the node which sensed the event first (earlier starting sensor in the experiment), Node7 is the node which sensed the event next (later starting sensor in the experiment), and Node3 receives packets from Node6 and Node7. Sink is the place where all data packets are gathered.

3. QUEUE SERVING SCHEMES

An image embedded to Node6 and Node7 is fragmented to multiple packets and sent to Node3. Node3 inserts received packets to its queue. Two different queue serving mechanisms are present at Node3: First Come First Served (FCFS) and Event Based Fairness – Least Attained Service (EBF-LAS). These queue serving mechanisms will be compared in the experiments.

3.1. First Come First Served (FCFS)

This is the simple classical queue serving scheme. Every packet arrived to Node3 is inserted to the end of the queue and the first packet in the queue is being sent.

Event id: 1 Packet id: 3	Event id: 2 Packet id: 8	Event id: 2 Packet id: 9

Figure 3.1. Queue before arrival of a packet

Event id: 1 Packet id: 3	Event id: 2 Packet id: 8	Event id: 2 Packet id: 9	Event id: 3 Packet id: 5

Figure 3.2. Queue after arrival of a packet

As it can be seen from Figure 3.1 and Figure 3.2, the new arriving packet is simply appended to the end of the queue.

3.2. Event Based Fairness – Least Attained Service (EBF-LAS)

The initial frames of an event deserve special care because they contain the earliest visual information. To give priority to the initial frames, an application level fairness scheme called Event Based Fairness (EBF) which is a scheduling scheme based on the Least Attained Service (LAS) will be used. For this purpose, I have implemented the EBF LAS scheduling scheme in C. It works as follows: Every packet that a sensor node generates has an Event ID, and Packet ID. Event ID's uniquely determine an event generated in the sensor network, and every event has a different Event ID. Since the events are assumed to be video streams, they consist of successively taken images. However, an individual image is larger than the transmission unit of the medium, so an image is divided into multiple packets and each packet is relayed with unique Packet ID's.

The nodes have a queue to which they insert the received packets, and from which they pop and send the packets to be forwarded. This queue is implemented as a sorted doubly linked list. The insertion of the packets to the queue takes the packet numbers into account, thus the queue is sorted on Packet ID's. Therefore, this scheme

gives priority to the events that have fewer packets sent and initial packets of an event get priority.

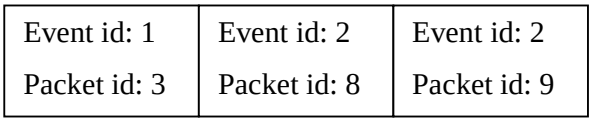


Figure 3.3. Queue before arrival of a packet

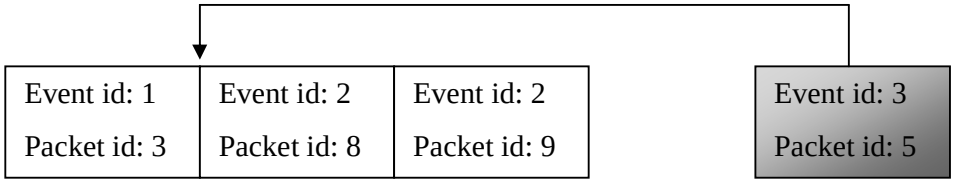


Figure 3.4. Arrival of a new packet with Packet id: 5

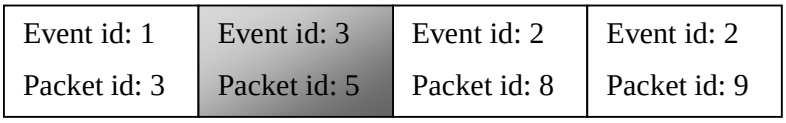


Figure 3.5. Queue after arrival of the packet

Figure 3.3 shows the initial state of the queue. There are three packets of two different events. Then, a packet with Packet ID 5 arrives to the queue (Figure 3.4). Since the queue is sorted according to the Packet ID's, it is inserted to the appropriate location to keep the queue sorted. Therefore, the new packet is inserted between packets with ID 3 and 8 (Figure 3.5).

4. EXPERIMENTS

The experiments are conducted as follows. A moving object is assumed to pass through an area of sensors. First a sensor senses the movement and starts to produce data. Then the object gradually moves away from the sensing area of first sensor and starts to enter the sensing area of the second sensor. This scenario is accomplished by starting the first sensor (Node6) to send the image, waiting a while, and starting the second sensor (Node7) to send the image. The images are embedded into the sensors. Each experiment is repeated 5 times to eliminate outlier results.

The original image being sent over the sensor network:

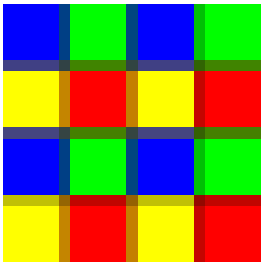


Figure 4.1. Original Image

4.1. Experiment 1

Node6 and Node7 send packets to Node3 every 100 milliseconds. Node3 forwards a packet to sink every 100 milliseconds. This means, Node6 and Node7 individually send packets that Node3 can handle, but combined they fill up Node3's queue.

4.1.1. FCFS Results

Node3 serves its queue in FCFS fashion.

This is the image of Node6 received by the sink:

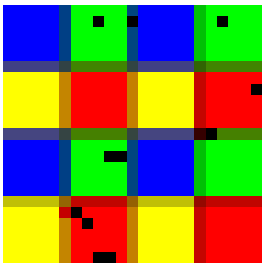


Figure 4.2. Image of Node6

The image of Node7 received by the sink:

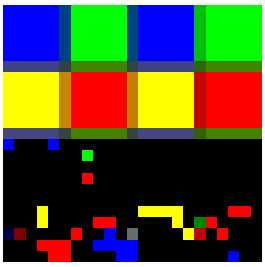


Figure 4.3. Image of Node7

We can get little information from the image of Node7. The reason is, when Node7 started to send the image, the queue of Node3 is nearly full with the packets of Node6 already. After a while, the queue of Node3 is completely full and some of the received packets are being dropped. Since Node3 serves its queue in a FCFS basis, the packets of Node7 are mostly dropped.

4.1.2. EBF-LAS Results

Node3 serves its queue in EBF-LAS fashion.

The image of Node6 received by the sink:

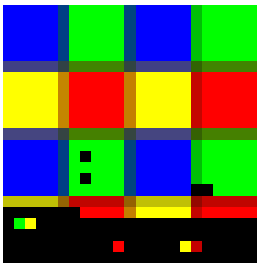


Figure 4.4. Image of Node6

The image of Node7:

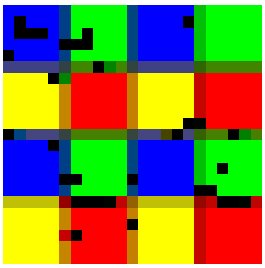


Figure 4.5. Image of Node7

Now the condition of Node7's image is much better than the previous case. Here when Node7 started to send, even Node 3's queue was full of Node6's packets, Node7's packets managed to reach the sink because of the queuing mechanism being fair. It gives priority to the initial frames of an event. Node7 didn't send any packets previously, therefore its packets have priority over Node6's packets. This is also the reason of the distortion at the end of Node6's image.

4.2. Experiment 2

Node6 and Node7 send packets to Node3 every 50 milliseconds (twice as fast as the previous case). Node3 forwards a packet to sink every 100 milliseconds (same as previous case). Both Node6 and Node7 individually send twice as many packets as Node3 can handle.

4.2.1. FCFS Results

Node3 serves its queue in FCFS fashion.

This is the image of Node6 received by the sink:

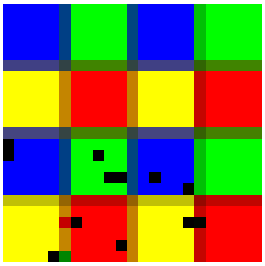


Figure 4.6. Image of Node6

The image of Node7 received by the sink:

Figure 4.7. Image of Node7

Now, Node7 can't even handle only one sensor's data. Therefore, drop rate increased dramatically. Since FCFS gives priority to packets which arrived earlier, Node6's packets arrived to sink with good quality. However, most of

Node7's packets are being dropped since the buffer is full and Node7 transmits packets faster than Node3 can handle. Now we can get even less information from Node7's image compared to the previous case because of increased packet sending frequency. The important result to mention is that we get very low information from the event sensed by Node7. Only Node6's event is detected with good quality, the image of Node7 is nearly useless.

4.2.2. EBF LAS Results

Node3 serves its queue in EBF-LAS fashion.

The image of Node6 received by the sink:

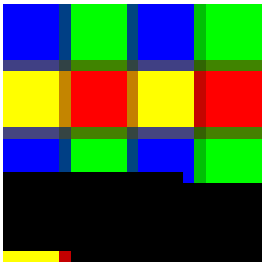


Figure 4.8. Image of Node6

The image of Node7 received by the sink:

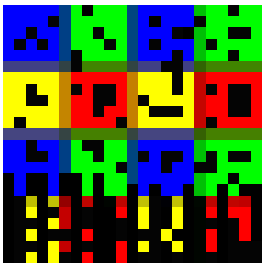


Figure 4.9. Image of Node7

With LAS, we can detect both Node6 and Node7's event. Node6's packets have priority over Node7's since Node6 didn't send any packets previously. Since Node3 gives priority to the initial frames of an event, Node7's packets have priority over Node 6's packets. Therefore, we can detect Node7's event fairly well compared to the previous case. In FCFS case we

were able to detect only the event of Node6. Node7's event was nearly undetectable. However, in LAS case both Node6 and Node7's event are detectable. This is the most important feature of LAS.

4.3. Experiment 3

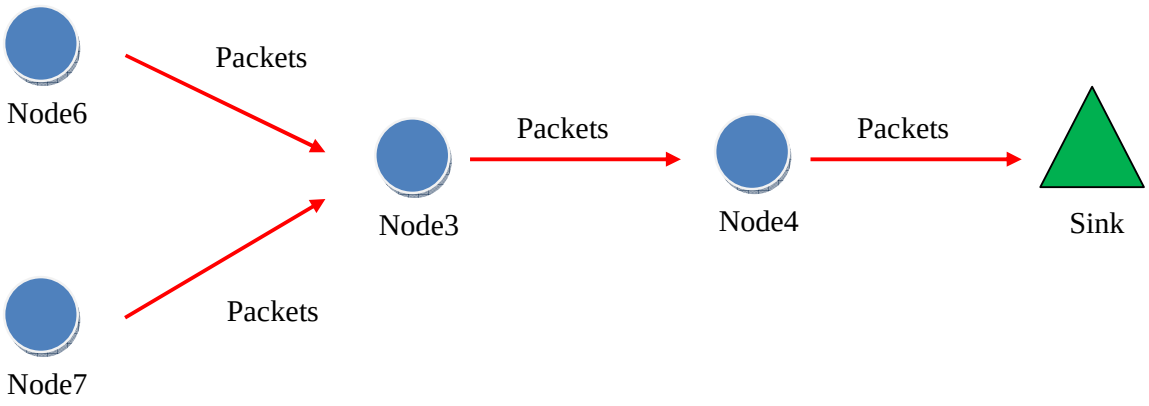


Figure 4.10. Modified Topology

When the topology is modified like Figure 4.10, a difference is not detected on the quality of detected images. So, we can conclude that the quality of the images mostly depend on transmission frequency of nodes and queue serving policy but not much on MAC layer and Exposed Terminal issue.

4.4. Experiment 4

In this experiment, the number of packets received from each event is recorded and compared for FCFS and EBF-LAS scheduling schemes. There are 3 event generating nodes (Node6, Node7, and Node8) sending their packets to the middle node (Node3) which forwards them to the sink. The topology is very similar to the one used in the previous experiments, only one additional node added to generate more traffic.

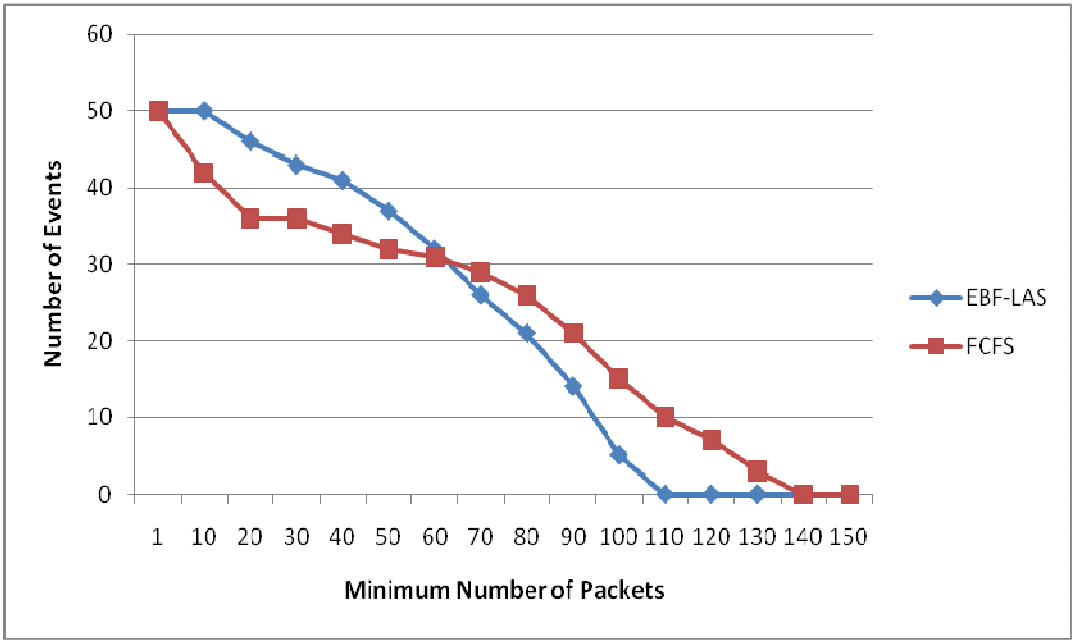


Figure 4.11. Experiment 4

A total of 50 events are generated. Each event has a unique ID. Events start and end at random times, and on the average 200 packets per event are generated. The horizontal axis of the graph is minimum number of packets per event received at the sink, and the vertical axis is the number of events. A point (x,y) in the graph represents the number of events (y) that have reported more than x number of packets to the sink. For example, for FCFS case there are 42 events that have reported more than 10 packets to the sink, and for EBF-LAS case there are 50 events that have reported more than 10 packets to the sink.

We can numerically verify that EBF-LAS gives priority to the initial packets of an event. For example, in FCFS case there are 36 events that have reported more than 20 packets to the sink, whereas in EBF-LAS case there are 46 events. So there are 14 events in FCFS case that couldn't report more than 20 packets to the sink, but in EBF-LAS case there are only 4. Another important result to note is that EBF-LAS approaches to zero before FCFS. That is EBF-LAS sends less number of packets per event that FCFS, but it equally distributes packets between events. There are 7 events in FCFS case that report more than 120 packets, but in EBF-LAS there are 0. The reason is because of the priority scheme ensured by EBF-LAS, the more packets that an event send the less priority those packets get.

5. DETAILED ANALYSIS

More detailed information about testing, programming of nodes, and data received at the sink will be presented.

5.1. Testing Environment

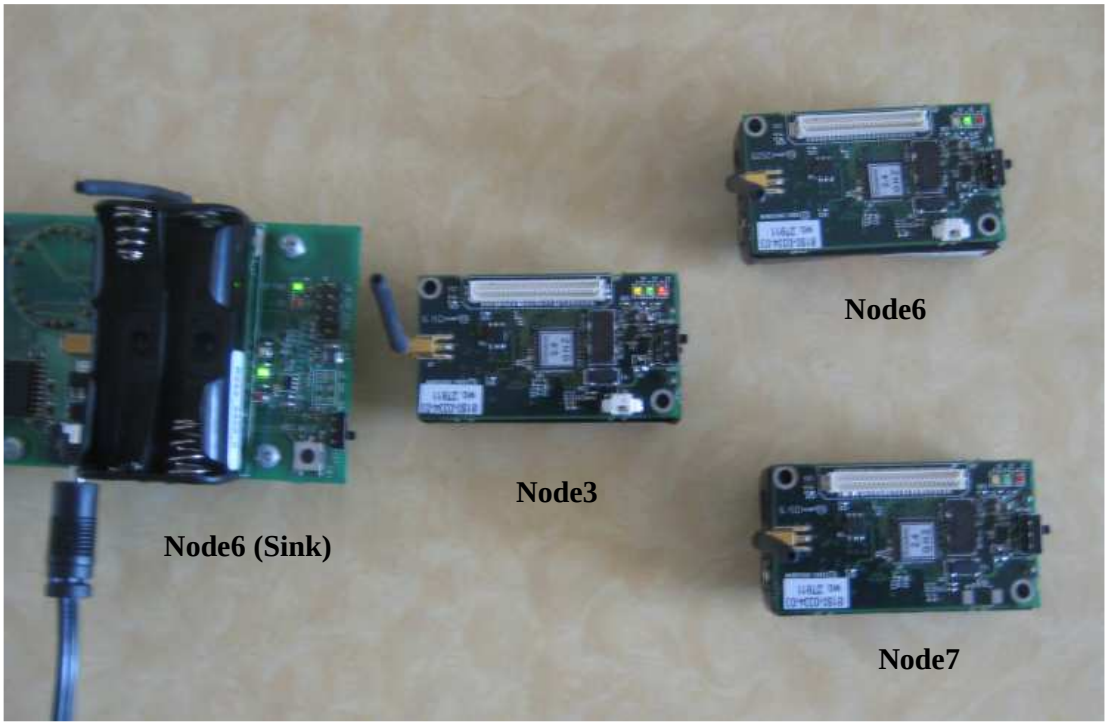


Figure 5.1. Testing Environment

The nodes (except sink) are programmed such that they toggle certain LED's during specific actions. The sink only receives packets and only toggles green LED.

Green LED: When a node sends a packet, green LED is toggling.
Yellow LED: When a node receives a packet, yellow LED is toggling.
Red LED: When a node drops a packet, red LED is toggling.

Here both Node6 and Node7 are sending packets (in fact Node7's LED was toggling green too but I couldn't capture the moment where all LED's were blinking).

Node3 is seen as sending, receiving and dropping packets. All these are not happening at the same time. In fact Node3 receives 20 packets per second, sends 10 packets per second, and drops approximately 10 packets per second. But the LED's can't toggle so fast, instead they toggle 2 times per second. That's why it is seen that Node3 is doing all these actions at the same time.

5.2. Programming Environment

There are 3 types of applications running on sensors. Namely, Radio, Multihop, and BaseStation. BaseStation is a premade application available at TinyOS applications. Radio and Multihop applications are written by me. Node6 and Node7 run the Radio application and they continuously send packets. Node3 runs the Multihop application. It receives packets from Node6 and Node7, inserts them into its queue, and sends packets from this queue to sink according to a queue serving mechanism (either FCFS or LAS). The sink runs the BaseStation application. It acts as a bridge between the air interface and the serial cable, and directly forwards packets received from radio to serial.

The applications are installed to the sensor like the following:

make micaz install,6 mib510,/dev/ttyS0

This command compiles the program contained in the current directory and installs in to the sensor connected to the serial port /dev/ttyS0. Micaz is the target platform. The number 6 after the install command is to specify the ID of the node. MIB510 is an installation option to specify which serial port to use.

These commands are issued to install the applications to nodes (every time a new node ID is specified after the install command). Then the node running BaseStation is put on the sensor board connected to the serial port of the PC with a serial cable, and the following command is issued at the terminal of the PC:

java net.tinyos.tools.Listen -comm serial@COM1:micaz

This command specifies that Java application Listen to listen the COM1 serial port at the correct speed for a micaz mode. After executing this command the received packet data is flowing on the terminal of the PC.

5.3. Data Received at Sink

The packet format is as follows:

typedef struct packet{

uint16_t nodeID;

uint16_t sequenceNumber;

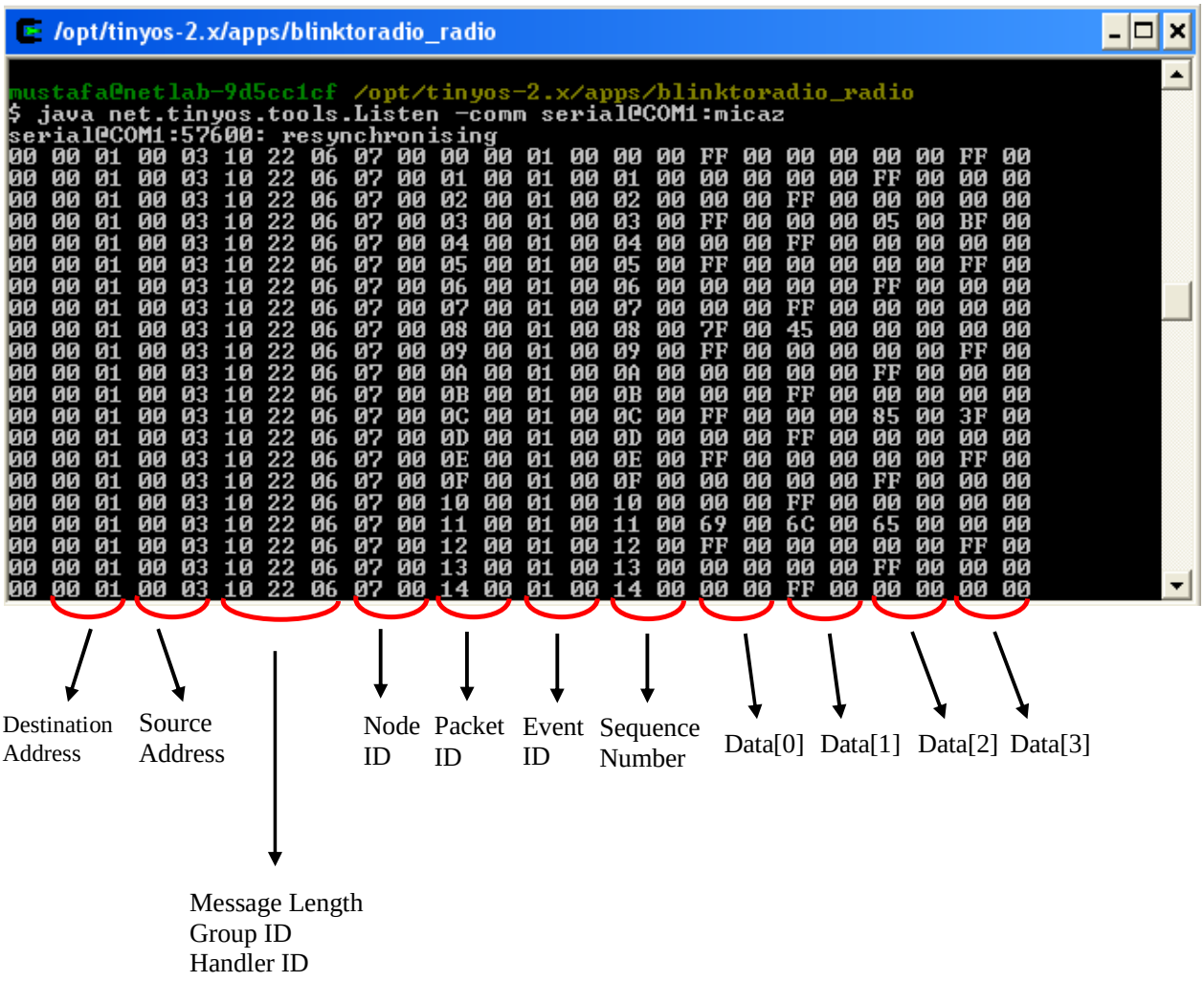
uint16_t eventID;

uint16_t packetID;

uint16_t data[4];

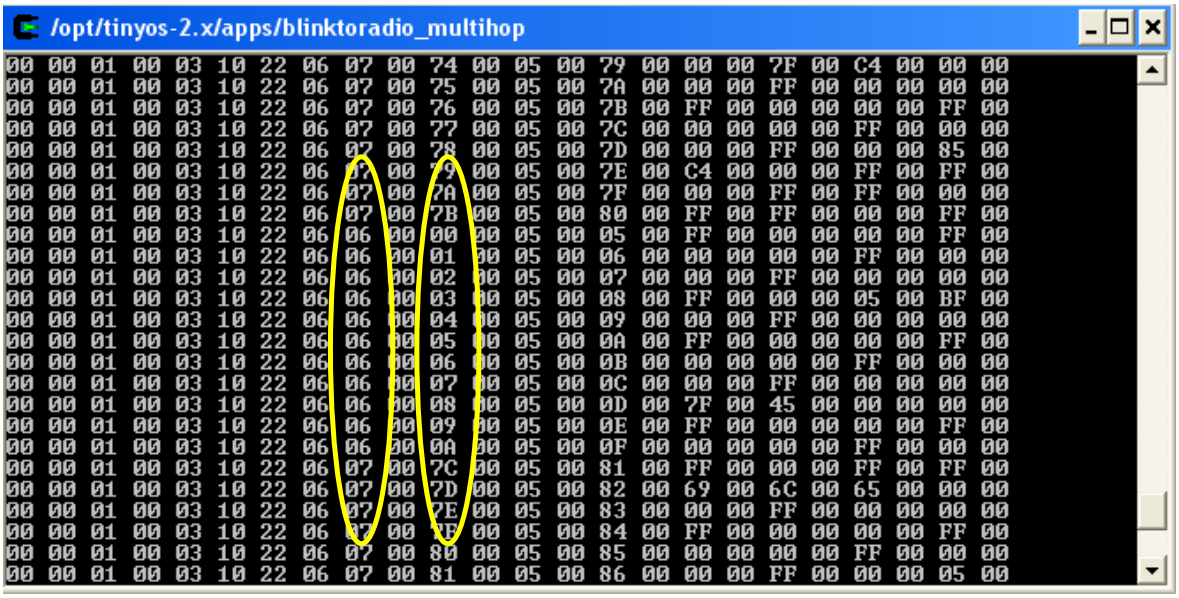
}packetStruct;

This is the content of the packets received at sink:



Destination Address is 1 since sink's Node ID is 1. Source Address is 3 since both Node6 and Node7 send their packets to Node3, and Node3 forwards these packets to the sink. Node ID is 7 since Node7 sends these packets. Packet ID's and Sequence Number's start from 00 and increment with each packet. Packet ID's are used to sort the packets in the queue of Node3. Sequence numbers are used to sort the packets at sink in case of out-of-order delivery. Event ID is 1 since this is the first event. Data is bytes of the bitmap image. Message Length, Group ID, and Handler ID are not programmed by me. They are put into the packet by TinyOS running at the sensor.

This sample demonstrates how LAS works:



In this experiment first Node6 starts to send its packets. After a while, Node7 is turned on for a very short time and then turned off.

The numbers within the yellow circle on the left are node ID's of the sensors. First, packets with node ID 7 flow, and then some packets with node ID 6 intervene, after that packets with node ID 7 continue. This shows that Node7's initial packets are given priority at Node3's queue as expected.

The numbers inside the yellow circle on the right are packet ID's of the packets. Packet ID's start from 00 and get incremented with every packet. At the moment, packets ID 78,79,7A,7B are arriving at the sink. These are the packets of previously started Node6. Then we see packets with ID's 00,01,02 ... 09,0A. These are packets of Node7 (we can also verify this from the node ID's in the left circle). Since these packet ID's are smaller than the previous ones, they get priority at Node3's queue.

6. CONCLUSIONS AND FUTURE WORK

After the tests, we have visually seen that fairness is an important issue in Wireless Sensor Networks, especially if the data contains visual information. After conducting several tests we have verified that fairness is achieved using EBF-LAS scheduling scheme. Since this scheme gives priority to the initial packets of an event, every event is detected with fairly well quality comparing to FCFS scheme where some events may remain undetected. Therefore, EBF-LAS scheme will be convenient for healthcare monitoring applications using Wireless Video Sensor Networks.

As a future work, these queuing schemes may be tested using real video streams. Since the cameras of the sensors are not available, predetermined images are sent continuously through the network. However, to be completely accurate, the tests must also be done using cameras. But current situation is still realistic because we are still sending images as if they were taken from a camera.

As well as the image quality, the delay that the packets of an event encounter is also an important metric in WSNs, because an out-of-date data would contradict with the Quality-of-Service (QoS) requirements of the application. In the field of healthcare monitoring, the QoS requirements are higher than general purpose WSNs and the application is more delay sensitive due to its nature. So, the reporting delay of the algorithms must be tested too. The initial packets are still important because they contain the earliest visual information and also the delay experienced by them directly affects the reporting delay. So, fair queuing schemes are needed in this case either.

REFERENCES

[1] Durmuş, Y., “Performance Evaluation and Enhancements in Video Sensor Networks”, Master Thesis, Boğaziçi University, 2009.

[2] TinyOS Execution Model
http://docs.tinyos.net/index.php/Modules_and_the_TinyOS_Execution_Model

[3] Radio Communication Between Nodes
http://docs.tinyos.net/index.php/Mote-mote_radio_communication

[4] Serial Communication Between Nodes and Sink
http://docs.tinyos.net/index.php/Mote-PC_serial_communication_and_SerialForwarder

[5] Levis, P., TinyOS Programming, October 2006.
http://www.tinyos.net/tinyos-2.x/doc/pdf/tinyos-programming.pdf