Presented at the DLSU Research Congress 2015
De La Salle University, Manila, Philippines
March 2-4, 2015

Queuing Schemes for Wireless Sensor Nodes Transmitting Prioritized Data

Christel Kimberly Cantillas, Phillip Enrico Ching, Jonathan James Esguerra, Aeron Luc Somera and Arlyn Verina Ong*
Center for Networking and Information Security, De La Salle University
Corresponding Author: arlyn.ong@delasalle.ph

Abstract: Wireless sensor networks (WSNs) are networks that use low powered devices interacting via radio signals. These networks are popular as platforms that provide monitoring and control services to an environment due to their portability and mobility. In order to acquire a comprehensive picture of a monitored environment, wireless nodes are deployed with mounted sensors to collect data from multiple heterogeneous sources, then transmitting these to receiving stations for processing. Sensors exhibit varying data characteristics in terms of transmission delay tolerance and reliable delivery requirements. If a wireless node is installed with multiple sensors, data from these sensors must be prioritized and queued accordingly for transmission. Critical data must be delivered as quickly and as reliably as possible. Low priority data, on the other hand, may be delayed or dropped during high traffic situations; but are still valuable to a monitoring application. As such, their delivery must not be completely omitted even if the wireless node has much data to transmit. This study provides a comparative study of data queuing techniques that may be used in WSNs that carry heterogeneous data traffic. To evaluate techniques, the study simulates a wireless node that acquires data with three priority levels: high, mid and low. These data are then run through test cases that vary the queuing technique used, as well as the ratio of high, mid, and low priority that the node must transmit. Based on results, Fair Preemptive and Aging-based Multilevel queuing provided the best performances in data delivery ratio. Both techniques allowed critical data to be quickly transmitted, while still providing opportunity for low priority data to be transmitted in high traffic scenarios.

Key Words: data queuing; data prioritization; wireless sensor nodes

1. INTRODUCTION

The Wireless Sensor Network (WSN) is an emerging technology that consists of small and smart sensor nodes that monitor physical and environmental phenomena over a given area. Sensor nodes may have multiple sensing units that observe different phenomena, and therefore generate heterogeneous traffic.

The nature of WSN traffic ranges from simple and periodic to unpredictable and bursty (Monowar et al, 2012). In addition, the importance of data gathered and transmitted by sensor node may also vary depending on the context of the application of the network. For example, some sensors may

generate infrequent but critical data; while others may regularly send data that do not often drastically change in value. Consequently, these sensors exhibit varying data stream characteristics in terms of delay tolerance, throughput and network reliability requirements (Xia, 2008).

In a sensor network, the wireless medium of transmission creates a shared environment where only a single node may transmit data at a time. With this constraint, periods of heavy wireless network traffic may impact the success of data transmission and the data travel time within the sensor network. In spite of this, critical data must be still delivered with as little delay as possible and with minimal loss. On the other hand, mundane data may be delayed or dropped during such situations; however, these must still be delivered to their destination, albeit sparingly, as these still provide valuable information regarding the monitored environment to the network application.

Wireless nodes that may be installed with multiple types of sensors must be designed with a method to cope with such requirements. When there is heavy network traffic and multiple nodes may be competing for network time, a node may not be able to immediately transmit across the network when it needs to. As such, it may benefit from a prioritization scheme which allows it to service the data transmission of its more critical sensors as soon as the network becomes available. Less critical ones may be treated with lower priority and will have their data transmitted when there are additional opportunities to use the network.

Among WSN communication protocols, one that is designed with provisions for data prioritization is Beacon Advertisement-based Time Division Multi-Access (Beacon-ATMA). Intended for deployment in a home automation application, the protocol is capable of differentiating traffic from heterogeneous sensors, which is essential to the data delivery requirements its target application (Ong and Cu, 2014). This protocol implements data prioritization in its queuing and scheduling algorithms. In this way, data delivery time and reliability is scaled according to priority; however, its simple queuing and prioritization strategy causes low priority data starvation in networks with a large number of high priority sensors during periods of heavy traffic. With this, it is possible that low priority data may be forced to wait for extended periods, and in extreme conditions, may never reach their destination at all. This results in starvation which must be addressed because it degrades the overall performance provided in a network (Wong et al, 2011; Warrier et al, 2006).

This research aims to improve the data queuing methods of the Beacon-ATMA for better handling of lower priority data even during periods of network congestion. In the succeeding sections, this paper shall discuss the queuing algorithm used in Beacon-ATMA and the cause of the data starvation issue. This will be followed by the alternative queuing strategies that were tested for potential improvement of the algorithm; as well as a comparison of test results among these methods. Finally a conclusion presents the research findings and future work for the project.

2. BEACON-ATMA

The topology of a WSN implementing the Beacon-Advertisement Based Time Division Multi-Access protocol (Beacon-ATMA), as represented in Fig. 1, is composed of a coordinator, a sink and sensor nodes organized in a peer-to-peer topology.

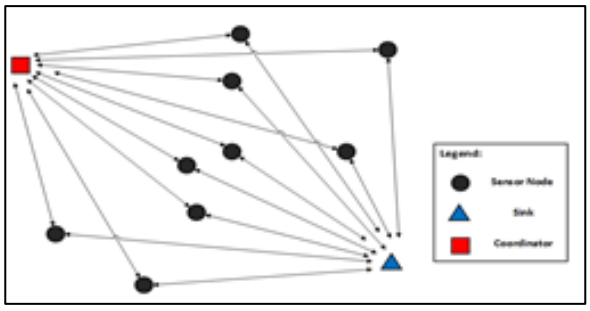

Fig. 1. Network Model

All sensor nodes have logical connections to both the coordinator and the network data collector node, known as a sink. Control messages are directly sent to the coordinator, and data directly to the sink from the sensor nodes. Three data priority levels are available for static assignment to sensors: high, mid and low.

The data queuing and transmission procedure using Beacon-ATMA is as follows: Once data is generated from a sensor, it is encapsulated by the wireless node housing the sensor into a data frame containing information such as node ID, time of generation, and priority level of the source sensor. The data frame is then inserted into the transmit queue of the node according to decreasing priority. Should there be any existing data on the queue that are of equal priority, then the data of the same level are ordered according to generation time. Any data frames that are of lower priority level are pushed further down the queue. The transmit queue is of fixed capacity; therefore in situations where the backlog of data for transmission exceeds the capacity of the queue, those at the tail end are discarded if

higher priority data is further generated (Ong and Cu, 2014).

In preparation for transmission, a node must wait for a beacon from the network coordinator which is broadcasted at fixed intervals to indicate the start of a period for nodes to request for permission to use the network. Nodes that are waiting to transmit data send requests to the coordinator during this period in the form of an advertisement control message that indicates the priority of the data that they intend to transmit.

The coordinator collects all requests then schedules the sending order among requesting nodes by allowing those with requests of higher priority to send first, followed by those with medium priority, then finally those with low priority. The transmit schedule is relayed back to the requesting nodes at which each will transmit the data frame waiting at the head of the transmit queue to the intended destination. An acknowledgment is expected from the receiving node for transmission to be considered successful. Otherwise, the sending node must attempt retransmission using the same procedure.

The method by which data frames are inserted into the transmit queue presents a potential issue of low priority data starvation. On a node that houses sensors of different priority levels, it is possible for data to continuously delay the transmission of those from lower priority sensors. This is due to three factors identified in its queuing strategy. First, the strategy allows high priority data to always pre-empt those of lower priority. Second, any data is not allowed to pre-empt those of higher priority even if it has been waiting in the queue for an exceptionally long time. Finally, priority levels of data remain fixed regardless of changing network conditions. In the event that a high priority sensor happens to frequently send data, lower priority data will be pushed back in queue, extending their waiting time.

The situation would be worsened during periods of high network activity. As more nodes contend for a chance to transmit data, the wireless medium stays occupied for longer periods of time. This directly affects all nodes as those that are not immediately granted permission to use the medium will have to store data in transmit queues even longer. If such network conditions persist over long periods, then node transmit queues will be filled to capacity causing the low priority data at the tail end to be continuously dropped especially if there is a significantly high ratio of high priority to low priority sensors. This can lead to cases where there is barely any low priority data moving through the network. As mentioned earlier, such a scenario must be avoided as much as possible because while delays and a certain level of loss are acceptable for low priority data, the application using the network must not be severely starved of low priority data.

3. QUEUING ALGORITHMS

To address the issue identified in the Beacon-ATMA protocol, this research investigates three alternative queuing algorithms that may be used to improve Beacon-ATMA. These are the Fair Preemptive Multilevel Queue, the Aging-based Multilevel Feedback Queue, and the Multilevel Priority Queuing with Round Robin Scheduling algorithms.

3.1 Fair Preemptive Multilevel Queue

The Fair Preemptive Multilevel Queue (FPMQ) Algorithm allows mid and low priority frames to pre-empt high priority frames when a frame's counter reaches its assigned maximum value (Silberschatz at al, 2011). While no frame's counter has maxed out, high priority frames will pre-empt all lower priority frames. A frame's counter increases whenever it is pre-empted, guaranteeing that each frame will eventually be given a fair chance to transmit.

This algorithm first checks a frame's priority so that it will be able to insert frames into their designated priority queues. Frames are inserted according to ascending create time. Upon inserting a frame into its designated priority queue, the queue is checked if it is full. When it is full, the frame at the tail of the queue is discarded so that an incoming frame can be successfully inserted in the priority queue.

Once a frame is successfully inserted, the counters of the frames at the head of the mid and low queues are checked if they have been maxed out. If not, the counter is incremented. Otherwise, that particular frame pre-empts all other frames regardless of priority. If none of the counters are maxed out, all frames from the mid and low priority queues are pre-empted by the frames from the high priority queue. When both the counters of the frames at the head of the mid and low priority queue get maxed out at the same time, the frame with the lower create time pre-empts the other frames.

3.2 Aging-based Multilevel Feedback Queue

The Aging-based Multilevel Feedback-Queue (AMFQ) Algorithm uses the concept of aging—a technique used to gradually increase the priority of a

task while its waiting time increases—to ensure that lower priority frames will be transmitted. This algorithm makes use of a multilevel feedback-queue, which allows frames to move from one queue to another (Silberschatz et al, 2011).

This algorithm has a queue for each priority level. Similar with the FPMQ algorithm, frames enter the queues sorted according to the time they were created. When a frame is to be queued, its priority is checked first. Low priority data is placed at the lowest level queue; mid priority data at the middle level queue; and high priority data at the highest level queue. In this way, it is guaranteed that higher priority data will be transmitted before lower priority data. If the priority queue is full, as in the FMPQ algorithm, the frame at the tail is discarded.

The mid and low level priority queue make use of counters that increase every round. After a statically assigned number of rounds, the priorities of the frame at the head of the mid and low priority queues are increased. When its counter reaches the maximum, the frame's priority will be increased to the next higher priority; and it will be transferred into the appropriate queue of its new priority.

3.3 Multilevel Priority Queuing with Round Robin Scheduling

The Multilevel Priority Queuing with Round Robin Scheduling (MPQRR) Algorithm also uses a multilevel queue of three levels, one for each priority (Silberschatz et al, 2011). The algorithm uses a round robin scheduling scheme to dictate which priority queue is allowed to transmit. This scheduling scheme uses a time quantum or time slice for each queue. Each queue is assigned a different value for its time quantum; the larger the time quantum, the more frames are allowed to be transmitted from the queue. Typically, the high priority queue will be given the highest time quantum, and the low priority queue the lowest time quantum.

When a frame is to be inserted into the node's transmit queue, its priority is checked first, and then is inserted into the appropriate priority queue. A variable *prioVar* keeps track of which queue is allowed to transmit. The time quantum of the current queue allowed to transmit will then be checked. If the time quantum of the queue is not yet exhausted, it will be permitted to transmit further. Otherwise, *prioVar* will be set to the next priority queue. If all time quanta have been exhausted, all the quanta will be reset. Before the round ends, the time quantum of the current queue will be decremented.

4. RESULTS AND DISCUSSION

To evaluate the three proposed algorithms against the original data queuing algorithm of the Beacon ATMA protocol, each algorithm was simulated on the node level. The network itself and its communication processes were abstracted, focusing on the data queuing performance of the algorithms. The node to which the algorithms were implemented was assumed to have only three sensors—one assigned a high priority, one mid, and one low. The length of the queues was set to twenty (20) frames. The counters for the FPMQ and the AMFQ algorithms were set to five and seven for the mid and low priority queues, respectively; the counters for the MPQRR algorithm were set to eight (8), five (5), and three (3) for the high, mid, and low priority queues, respectively. A summary of the parameter matrix can be seen in Table 1.

Table 1. Simulation parameters

Parameter	Value
Number of nodes	1
Number of sensors	3 (1 high, 1 mid, 1 low)
Size of queue	20
FPMQ and AMFQ counters	Mid - 5 Low - 7
MPQRR counters	High - 8 Mid - 5 Low - 3

Test cases were prepared using a pseudo-random event generator to ensure that all four algorithms would be subjected to the same network behavior. Each of the algorithms were subjected to ninety (90) test cases, each with the possibility for data starvation to occur, varying in terms of the total number of data frames produced by the sensors, the ratio of high, mid and low priority data frames produced, and the frequency with which the node can transmit. In all test cases, each sensor generates an average of 533 frames for the duration of the test.

Table 2 and Figure 2 show the average number of dropped frames of each algorithm for all test cases. The total number of dropped frames in all test cases was simply divided by the number of test cases to get the average.

Table 2. Average number of dropped frames

Algorithm	High	Mid	Low	Total
B-ATMA	31.39	99.08	154.30	284.78
MPQRR	73.97	106.79	127.00	307.76
FPMQ	40.13	90.93	130.96	262.02
AMFQ	31.72	108.87	121.84	262.43

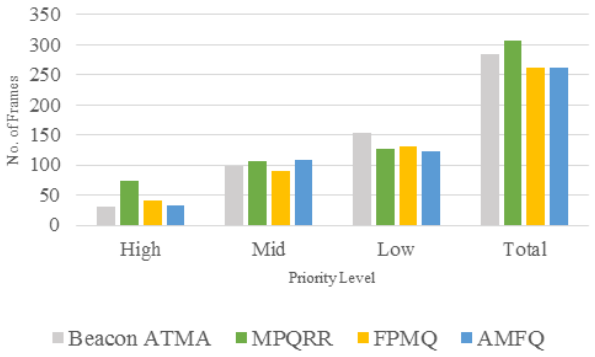

Fig. 2. Average number of dropped frames

The figure exhibits the susceptibility of the Beacon ATMA's current data queuing algorithm to low priority data starvation. Although it discards the least and transmits the most number of high priority frames on average, it also discards the most number of low priority frames. There is also a significant difference between Beacon ATMA's treatment of low priority data in comparison with the three proposed algorithms.

The MPQRR algorithm discarded the most number of frames in total. Since the maximum time quanta in the current design of the algorithm is static and there is no further processing after checking the time quantum of each queue, there are instances wherein the node, at least in the simulation, fails to transmit anything when the current queue that is under active transmission is empty. This places a hold on all other queues until it exhausts its quantum when further data is generated. It is also of note that the ratio between the numbers of transmitted frames is in proportion with the assigned time quantum for each priority queue.

The FPMQ and AMFQ vary slightly with each other in that the AMFQ appears to handle high priority data better than the FPMQ; however, the FPMQ exhibits more fairness, i.e. the number of dropped and transmitted packets are more distributed among the three priorities. This slight difference is due to the way the algorithms implement fairness: the FPMQ is more straightforward in that it allows lower priority frames to pre-empt higher priority frames; whereas the AMFQ only allows frames to move to a higher priority queue, which may still require waiting time, depending on the contents of the queues.

Succeeding test cases illustrate results of algorithm performance when the ratio of high, mid and low priority data to be sent by a node is unequal. In extreme conditions, the differences among the algorithms are more prominent. The following test cases show the average number of dropped and frames only for cases where the ratio of generated high, mid and low priority frames is 70:15:15 (H:M:L), and where the ratio of generated frames is 15:15:70 (H:M:L).

Table 3. Average number of dropped frames (70:15:15)

Algorithm	High	Mid	Low	Total
B-ATMA	133.89	71.44	79.56	284.89
MPQRR	256.00	12.33	29.89	298.22
FPMQ	162.22	58.89	32.22	253.33
AMFQ	134.56	68.44	50.33	253.33

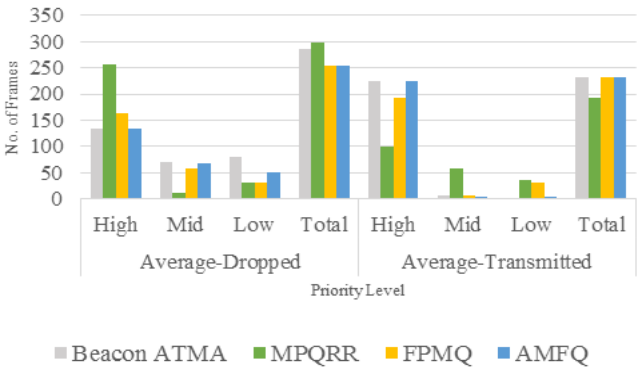

Fig. 3. Average number of dropped and transmitted frames for a ratio of 70:15:15: (H:M:L)

In cases where there is a significantly huge stream of incoming high priority data, as illustrated in Fig. 3, the FPMQ exhibits the best results, being able to transmit a large amount of high priority frames without necessarily sacrificing the transmission of lower priority frames. While the Beacon ATMA and AMFQ show similarity in their results, in such cases, the Beacon ATMA may not be able to transmit low priority data at all, while the AMFQ only does so minimally.

As illustrated in Table 4 and Fig. 4, where the generation of low priority data is significantly

larger than higher priority data, there is very minimal discarding of high priority frames. A large amount of low priority frames is still discarded, and only a low amount is transmitted. In such cases, the MPQRR has the least effective performance because the ratio of its transmitted frames is proportional to the time quanta assigned to each priority queue, and the time quanta for the low priority queue is only three (3). This means that low priority data is only allowed to be transmitted three (3) out of every sixteen (16) transmission rounds whether or not higher priority queues remain empty.

Table 4. Average number of dropped frames (15:15:70)

Algorithm	High	Mid	Low	Total
B-ATMA	0.00	13.00	271.89	284.89
MPQRR	4.33	21.44	328.00	353.78
FPMQ	0.00	12.67	262.78	275.44
AMFQ	0.00	16.11	258.67	274.78

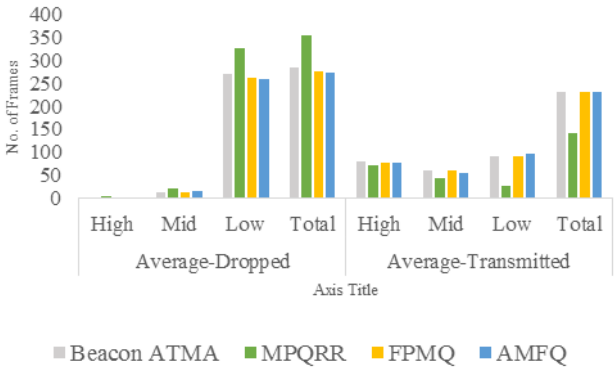

Fig. 4. Average number of dropped and transmitted frames for a ratio of 15:15:70 (H:M:L)

5. CONCLUSIONS

The three proposed data queuing algorithms exhibited notable differences from the Beacon ATMA's original queuing algorithm. All three feature multiple queues rather than a single data queue that the Beacon-ATMA protocol implements; and allow dynamic adjustment of data priority depending on network conditions. Results from MPQRR deviated most from those of the other three algorithms. While it transmits data proportionally among the three priority levels based on the assigned time quanta, it discards a significantly larger average number of frames and transmits significantly less, overall, than the rest of the algorithms. The FPMQ and AMFQ exhibited similar trends with the Beacon ATMA. In cases where there are large streams of incoming high priority data, the FMPQ strategy is able to transmit more low priority data than the AMFQ strategy and Beacon ATMA. The AMFQ, on the other hand, shows only a slight improvement of the Beacon ATMA, based on the test results. It is also notable that mid priority frames become a significant tradeoff for the Beacon ATMA, FMPQ, and AMFQ algorithms.

As further work for the study, the algorithms will be tested in network scenarios involving multiple nodes. The values of the parameters for test cases may be scaled up to further distinguish the difference in performance of the algorithms. The counters may also be fine-tuned to assess which values would be ideal for the algorithms in different network behaviors.

6. REFERENCES

Monowar, M., Rahman, O., & Pathan, A. K. (2012). Prioritized heterogeneous traffic-oriented congestion control protocol for WSNs [Electronic version]. The International Arab Journal of Information Technology, 2012, 39-48.

Ong, A. V. & Cu, G. (2014). Data Collection with Prioritization for Wireless Sensor Networks. Proceedings of Workshop on Computation: Theory and Practice (WCTP2013).

Silberschatz, A., Gagne, G. & Galvin, B. P. (2011). Operating system concepts, 7th ed. USA: John Wiley & Sons, Inc.

Warrier, A., Min, J. & Rhee, I. (2006). Mitigating starvation in wireless sensor networks. MILCOM '06 Proceedings of the 2006 IEEE Military Communications Conference, 1937-1941

Wong, Y., Deng, D. & Chen, Y. (2011). On alleviating starvation in wireless sensor networks [Electronic version]. 2011 IEEE International Conference on Communications (ICC), 1-5.

Xia, F. (2008). QoS challenges and opportunities in wireless sensor/actuator networks. Sensors 8, 1099-1110.