www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 5 may, 2014 Page No. 5930-5936

AN EFFICIENT MULTILEVEL PRIORITY PACKET SCHEDULING FOR WIRELESS SENSOR NETWORK

C.Vijayakumaran, K. Janaky
Dept. of CSE,
Valliammai Engineering College, Chennai, India.
c_vijayakumaran@yahoo.com
Dept. of CSE,
Valliammai Engineering College, Chennai, India.
janaky.sarav16@gmail.com

Abstract— **Scheduling real-time and non-real time packets at the sensor nodes is significantly important to reduce processing overhead, energy consumptions, communications bandwidth, and end-to-end data transmission delay of Wireless Sensor Network (WSN). Most of the existing packet scheduling algorithms of WSN use assignments based on First-Come First-Served (FCFS), non-preemptive priority, and preemptive priority scheduling. However, these algorithms incur a large processing overhead and data transmission delay and are not dynamic to the data traffic changes. In this paper, we propose a Dynamic Multilevel Priority (DMP) packet scheduling scheme. In the proposed scheme, each node, except those at the last level of the virtual hierarchy in the zone based topology of WSN, has three levels of priority queues. Real-time packets are placed into the highest-priority queue and can preempt data packets in other queues. Non-real-time packets are placed into two other queues based on a certain threshold of their estimated processing time. Leaf nodes have two queues for real-time and non-real-time data packets since they do not receive data from other nodes and thus, reduce end-to-end delay. We evaluate the performance of the proposed DMP packet scheduling scheme through simulations for real-time and non-real-time data. Simulation results illustrate that the DMP packet scheduling scheme outperforms conventional schemes in terms of average data waiting time and end-to-end delay.**

***Keywords* – Wireless Sensor Network; Packet scheduling; FCFS; Multilevel Queue; Priority Scheduling; Real-time data.**

I. INTRODUCTION

A Wireless Sensor Network (WSN) consists of numerous unattended, resource- constrained, low power, and memory sensor nodes, which are of two types: sensor node, and sensor gateway or base station (BS). The sensor node has the basic capabilities of sensing, processing, and communicating. However, the sensor gateway or BS has more functionality besides these basic capabilities. It can collect, analyze, and process raw sensor's data and be connected to the Internet to share the data world- wide. Based on the two types of sensors, a WSN normally constitutes a wireless ad hoc sensor network. Figure 1 illustrates a typical WSN, in which a sensor node senses different environmental parameters such as temperature, pressure, humidity, and sends data directly to the BS if the node is within the communication range of BS or through other nodes using multi-hop routing. Finally, data from BS reach end users through different communication networks, such as Internet.

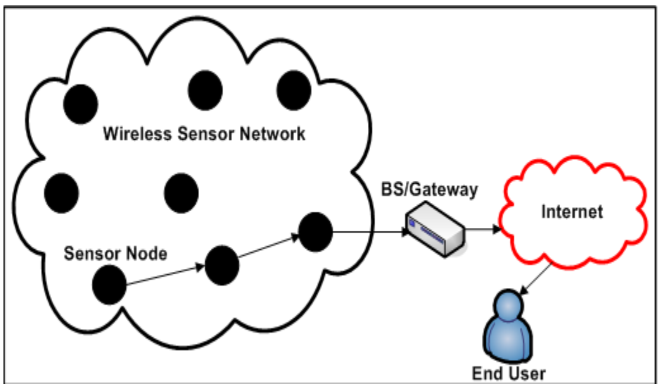

Fig 1. A Wireless Sensor Network

Though extensive research for scheduling the sleep-wake times of sensor nodes has been conducted [1]–[10], only a few studies exist in the literature on the packet scheduling of sensor nodes [11]–[14] that schedule the processing of data packets available at a sensor node and also reduces energy consumptions. Indeed, most existing Wireless Sensor Network (WSN) operating systems use First Come First Serve (FCFS) [15] schedulers that process data packets in the order of their arrival time and, thus, require a lot of time to be delivered to a relevant base station (BS). However, to be meaningful, sensed data have to reach the BS within a specific time period or before the expiration of a deadline. Additionally, real-time emergency data should be delivered to BS with the shortest possible end-to-end delay. Hence, intermediate nodes require changing the delivery order of data packets in their ready queue based on their importance (e.g., real or non-real time) and delivery deadline. Furthermore, most existing packet scheduling algorithms of WSN are neither dynamic nor suitable for large scale applications since these schedulers are predetermined and static, and cannot be changed in response to a change in the application requirements or environments [16]. For example, in many real time applications, a real-time priority scheduler is statically used and cannot be changed during the operation of WSN applications.

In this paper, we propose a dynamic multilevel priority (DMP) packet scheduling scheme that reduces end-to-end data transmission delay by allocating maximum three levels in the ready queue of each node. This approach schedules data based on their priority. For instance, real-time data have the highest priority and incur less end-to-end data transmission delay. This is one of the main objectives of using WSN in emergency applications. However, this scheduling approach achieves fairness by allowing the lowest priority non-real-time local data (i.e., data that are sensed at the current sensor node) to be transmitted after a certain number of timeslots if they cannot be transmitted due to the continuous arrival of the higher priority non-real-time remote data (i.e., non-real-time data, a node receives for the lower level nodes).

II. RELATED WORK

Scheduling data packets at sensor nodes, interchangeably used as task scheduling. It is significantly important since it determines the data transmissions order based on different factors such as data size, transmission deadline, and data priority. For instance, data sensed for real-time applications have higher priority than data sensed for non-real-time applications. Packet scheduling is very useful for heterogeneous Wireless Sensor Networks (WSNs) containing different types of sensors that sense different types of data.

In the following subsections, we classify packet scheduling schemes of WSN based on different factors (Figure 2), and present existing scheduling schemes of these types.

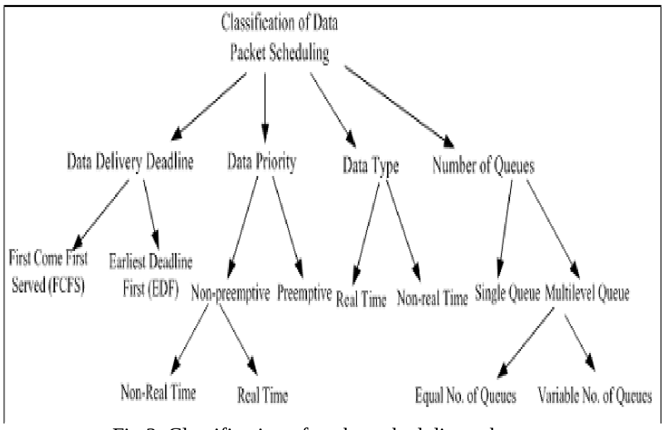

Fig 2. Classification of packet scheduling schemes

A. Classification Factor: Deadline

Packet scheduling schemes can be classified based on the deadline of arrival of data packets to the base station (BS), which are as follows.

First Come First Served (FCFS)
Most existing WSN applications use First Come First Served (FCFS) schedulers that process data in the order of their arrival times at the ready queue. In FCFS, data that arrive late at the intermediate nodes of the network from the distant leaf nodes require a lot of time to be delivered to base station (BS) but data from nearby neighboring nodes take less time to be processed at the intermediate nodes. In FCFS, many data packets arrive late and thus, experience long waiting times.

Earliest Deadline First (EDF)
Whenever a number of data packets are available at the ready queue and each packet has a deadline within which it should be sent to BS, the data packet which has the earliest deadline is sent first. This algorithm is considered to be efficient in terms of average packet waiting time and end-to-end delay.

B. Classification Factor: Priority

Packet scheduling schemes can be classified based on the priority of data packets that are sensed at different sensor nodes.

Non-preemptive
In non-preemptive priority packet scheduling, when a packet *t*1 starts execution, task *t*1 carries on even if a higher priority packet *t*2 than the currently running packet *t*1 arrives at the ready queue. Thus *t*2 has to wait in the ready queue until the execution of *t*1 is complete.

Preemptive
In preemptive priority packet scheduling, higher priority packets are processed first and can preempt lower priority packets by saving the context of lower priority packets if they are already running.

C. Classification Factor: Packet Type

Packet scheduling schemes can be classified based on the types of data packets, which are as follows.

Real-time packet scheduling
Packets at sensor nodes should be scheduled based on their types and priorities. Real-time data packets are considered as the highest priority packets among all data packets in the ready queue. Hence, they are processed with the highest priority and delivered to the BS with a minimum possible end-to-end delay.

Non-real-time packet scheduling
Non-real time packets have lower priority than real-time tasks. They are hence delivered to BS either using first come first serve or shortest job first basis when no real-time packet exists at the ready queue of a sensor node. These packets can be intuitively preempted by real-time packets.

D. Classification Factor: Number of Queue

Packet scheduling schemes can also be classified based on the number of levels in the ready queue of a sensor node. These are as follows.

Single Queue
Each sensor node has a single ready queue. All types of data packets enter the ready queue and are scheduled based on different criteria: type, priority, size, etc. Single queue scheduling has a high starvation rate.

Multi-level Queue
Each node has two or more queues. Data packets are placed into the different queues according to their priorities and types. Thus, scheduling has two phases: (i) allocating tasks among different queues, (ii) scheduling packets in each queue. The number of queues at a node depends on the level of the node in the network. For instance, a node at the lowest level or a leaf node has a minimum number of queues whilst a node at the upper levels has more queues to reduce end-to-end data transmission delay and balance network energy consumptions.

To eliminate problems in [16] Lee et al. [17] propose a multilevel queue scheduler scheme that uses a different number of queues according to the location of sensor nodes in the network. This approach uses two kinds of scheduling: simple priority-based and multi-FIFO queue-based. In the former, data enter the ready queue according to priority but this scheduling also has a high starvation rate. The multi-FIFO queue is divided into a maximum of three queues, depending on the location of the node in the network. If the lowest level nodes that are located at level have only one queue but there are two queues for nodes at level . Each queue has its priority set to high, mid, or low. When a node receives a packet, the node decides the packet's priority according to the hop count of the packet and accordingly sends it to the relevant queue. The work done by Karimi E. and Akbari B. [18] also proposes a priority queue scheduling algorithm for WMSN. In this scheduling scheme, buffer space of intermediate nodes is divided into four queues to hold three different types of video frames and one regular data frames. Data in the first three queues have the highest priority and are scheduled in round robin fashion. Data in the fourth queue is transmitted when the first three queues are empty. However, these scheduling schemes do not consider variable number of queues based on the position of sensor nodes to reduce the overall end-to-end delay.

III. PROPOSED PACKET SCHEDULING SCHEME

In this section, we present the proposed dynamic multilevel priority (DMP) packet scheduling mechanism for WSN.

A. Working Principle

Scheduling data packets among several queues of a sensor node and data packets that are sensed at a node are scheduled among a number of levels in the ready queue. Then, a number of data packets in each level of the ready queue are scheduled. For instance, demonstrates that the data packet, *Data*1 is scheduled to be placed in the first level, Queue1. Then, *Data*1 and *Data*3 of Queue1 are scheduled to be transmitted based of different criteria. The general working principle of the proposed DMP scheduling scheme is illustrated in Figure 3.

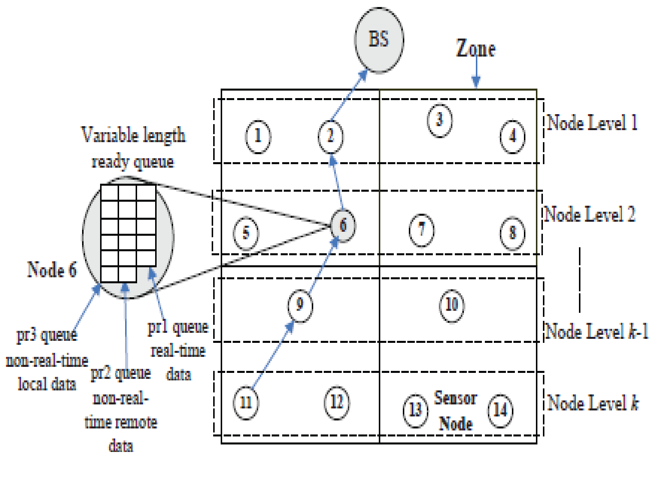

Fig. 3. Proposed dynamic multilevel priority packet scheduling scheme.

The proposed scheduling scheme assumes that nodes are virtually organized following a hierarchical structure. Nodes that are at the same hop distance from the base station (BS) are considered to be located at the same level. Data packets of nodes at different levels are processed using the Time Division Multiplexing Access (TDMA) scheme. For instance, nodes that are located at the lowest level and the second lowest level can be allocated timeslots 1 and 2, respectively. We consider three-level of queues, that is, the maximum number of levels in the ready queue of a node is three: priority 1 (*pr*1), priority 2 (*pr*2), and priority 3 (*pr*3) queues. Real-time data packets go to *pr*1, the highest priority queue, and are processed using FCFS. Non-real-time data packets that arrive from sensor nodes at lower levels go to *pr*2, the second highest priority queue. Finally, non-real time data packets that are sensed at a local node go to *pr*3, the lowest priority queue. The possible reasons for choosing maximum three queues are to process (i) real-time *pr*1 tasks with the highest priority to achieve the overall goal of WSNs, (ii) non real-time *pr*2 tasks to achieve the minimum average task waiting time and also to balance the end-to-end delay by giving higher priority to remote data packets, (iii) non-real-time *pr*3 tasks with lower priority to achieve fairness by preempting *pr*2 tasks if *pr*3 tasks wait a number of consecutive timeslots.

In the proposed scheme, queue sizes differ based on the application requirements. Since preemptive priority scheduling incurs overhead due to the context storage and switching in resource constraint sensor networks, the size of the ready queue for preemptive priority schedulers is expected to be smaller than that of the preemptable priority schedulers. The idea behind this is that the highest-priority real-time/emergency tasks rarely occur. They are thus placed in the preemptive priority task queue (*pr*1 queue) and can preempt the currently running tasks. Since these processes are small in number, the number of preemptions will be a few. On the other hand, non-real- time packets that arrive from the sensor nodes at lower level are placed in the preemptable priority queue (*pr*2 queue). The processing of these data packets can be preempted by the highest priority real-time tasks and also after a certain time period if tasks at the lower priority *pr*3 queue do not get processed due to the continuous arrival of higher priority data packets. Real-time packets are usually processed in FCFS fashion. Each packet has an ID, which consists of two parts, namely level ID and node ID. When two equal priority packets arrive at the ready queue at the same time, the data packet which is generated at the lower level will have higher priority. This phenomenon reduces the end-to-end delay of the lower level tasks to reach the BS. For two tasks of the same level, the smaller task (i.e., in terms of data size) will have higher priority.

Moreover, it is expected that when a node x senses and receives data from lower-level nodes, it is able to process and forward most data within its allocated timeslot; hence, the probability that the ready queue at a node becomes full and drops packets is low.

Timeslots at each level are not fixed. They are rather calculated based on the data sensing period, data transmission rate, and CPU speed. They are increased as the levels progress through BS. However, if there is any real-time or emergency response data at a particular level, the time required to transmit that data will be short and will not increase at the upper levels since there is no data aggregation. The remaining time of a timeslot of nodes at a particular level will be used to process data packets at other queues. Since the probability of having real-time emergency data is low, it is expected that this scenario would not degrade the system performance. Instead, it may improve the perceived Quality of Service (QoS) by delivering real-time data fast. Moreover, if any node x at a particular level completes its task before the expiration of its allocated timeslot, node x goes to sleep by turning its radio off for the sake of energy efficiency.

B. Pseudo-code

In our proposed DMP packet scheduling scheme, nodes at the lowest level, l_k, sense, process and transmit data during their allocated timeslots, whereas nodes at level l_k-1 and upper levels receive data in addition to sensing, processing and transmitting data. Now, we present the pseudo-code of our proposed DMP packet scheduling scheme.

```
while $task_{k,i}$ is received by $node_i$ at level $k$, i.e., $l_k$ do
if $Type(task_{k,i}) = real - time$ then
put $task_{k,i}$ into $pr_1$ queue
else if $node_i$ is not at lowest levels then
if $task_{k,i}$ is not local then
put $task_{k,i}$ into $pr_2$ queue
else
put $task_{k,i}$ into $pr_3$ queue
end if
else
put $task_{k,i}$ into $pr_2$ queue
end if
Assume, the duration of a timeslot at $l_k \leftarrow t(k)$
Data sensing time of $nodei$ at $l_k \leftarrow senseT\ imek(t)$
Remaining time after data sensing, $t_1(k) = t(k) -$
$senseT\ ime_k(t)$
Let total real-time tasks for $nodei$ at $l_k \leftarrow n_k(pr1)$
Let $procT\ imepr_1(k) \leftarrow \_nk(pr1)$
$j$=1 $procT\ ime(j)$
if $procT\ imepr_1(k) < t_1(k)$ then
All $_{pr1}$ tasks of $nodei$ at $l_k$ are processed as FCFS
Remaining time $t_2(k) \leftarrow t_1(k) - procT\ imepr_1(k)$
Let, total $pr_2$ tasks for $nodei$ at $l_k \leftarrow n_k(pr_2)$
Let $procT\ imepr_2(k) \leftarrow \_n_k(pr_2)$
$j$=1 $procT\ ime(j)$
if $procT\ imepr_2(k) < t_2(k)$ then
All $pr_2$ tasks are processed as FCFS
$Pr_3$ tasks are processed as FCFS for the remaining
time, $t_3(k) \leftarrow t_2(k) - procT\ imepr_2(k)$
else
$pr_2$ tasks are processed for $t_2(k)$ time
no $pr_3$ tasks are processed
end if
else
only $pr_1$ tasks are processed for $t_1(k)$ time
no $pr_2$ and $pr_3$ tasks are processed
end if
if $pr_1$ queue empty & $pr_2$ tasks are processed $\alpha$ consecutive
timeslots since $t(k) \leq procT\ imepr_2(k)$ then
$pr_2$ tasks are preempted at $\alpha + 1, \ldots, \alpha + j$ timeslots
by $pr_3$ tasks
if $pr_1$ task arrives during any of $\alpha$+1, $\alpha$+2, . . ., $\alpha$+$j$
timeslots then
$pr_3$ tasks are preempted and $pr1$ tasks are processed
context are transferred again for processing $pr_3$ tasks
end if
end if
end while
```

We consider only two levels in the ready queue of sensor nodes that are located at the lowest level since these nodes do not receive packets from any lower level nodes. Other nodes have three levels in the ready queue and place non-real time local tasks into pr_3 queue. We also consider that each node requires time to sense data packets and also process local and/or remote data packets. For instance, $t_1(k)$ in the pseudo-code represents the real-time data sensing time at a $node_i$. If the processing time of real-time data at $node_i$ is less than $t_1(k)$ then $node_i$ will have time remaining to process non-real-time pr_2 data packets. Similarly, if $node_i$ still has some remaining time, it can process non-real-time pr_3 data packets. The pseudo-code also shows that if the pr_1 queue is empty and pr_2 packets are processed α consecutive timeslots, the processing of pr_2 data packets will be preempted for j timeslots.

IV. PERFORMANCE ANALYSIS

In this section, we analyze the performance of the proposed DMP task scheduling scheme in terms of end-to-end delay, and total waiting time of different types of traffic at the ready queues of active nodes.

A. End-to-End Delay

In the following, we formulate the average end-to-end delay of transmitting different priority data packets to the base station (BS). Again, we interchangeably use task and data to represent the data packets that are sensed at a sensor node.

Real-time Priority 1 Queue Data: Let us assume that a node x, residing at level l_k is sensing a real-time, emergency event, e.g., fire detection. This node transmits the emergency priority 1 data to BS through l_k-1 intermediate levels. We consider the following scenario whereby every time a real-time data packet reaches a neighboring active node, y at an upper level, a non-real time lower priority data is being processed at that node. Hence, data delivery at y is preempted to send real-time data. However, a real-time task t_1 has to wait if there is a number, npr_1, of a real-time task ahead of t_1 at the pr_1 queue. We assume that all real-time data have the same size. Therefore, the end-to-end delay for a real-time task t_1 considering that t_1 has npr_1 number of real-time tasks ahead of it.

Non-real time Priority 2 Queue Data: Tasks at pr_2 queue can be preempted by real-time ones. Taking the scenario of Figure 3 as an example, we first consider the scenario when a real-time task is sensed at node 11 and is forwarded to BS through relay nodes 9, 6, and 2. It should be observed that tasks are available at the pr_2 queue at nodes 9, 6 and 2. Since one real-time task is available at the pr_1 queue of nodes 9, 6, and 2, real-time tasks will be processed and transmitted first during the timeslot of nodes 9, 6, and 2. The pr_2 tasks are processed in the remaining time of the timeslots. Thus, the total end-to-end delay for a pr_2 task that can be processed in the same timeslot exceeds

Non-real time Priority 3 Queue Data: In the best case, when no task is available at the pr_1 and pr_2 queues, the end-to- end delay of the pr_3 tasks will be almost equal to that of the pr_1 queue tasks (Equation 1) although it can differ slightly based on the size of the pr_3 queue task. We assume that the pr_3 queue tasks are processed by preempting pr_2 queue tasks if for α consecutive timeslots there is no task at the pr_1 queue but there are tasks available at the pr_2 queue. Let tk denote the length of a timeslot of nodes at level l_k . However, during the processing of the pr_3 queue tasks, these tasks can be preempted by real time tasks. They are processed again after the completion of real-time tasks.

V. SIMULATION RESULTS AND COMPARISON

The simulation model is implemented using the C programming language. It is used to evaluate the performance of the proposed DMP packet scheduling scheme, comparing it against the FCFS, and Multilevel Queue scheduling schemes. The comparison is made in terms of average packet waiting time, and end-to-end data transmission delay. We use randomly connected Unit Disk Graphs (UDGs) on a surface of 100 meter × 100 meter as a basis of our simulations. The number of simulated zones varies from 4 to 12 zones. Nodes are distributed uniformly over the zones. The ready queue of each node can hold a maximum of 50 tasks. Each task has a Type ID that identifies its type. For instance, type 0 is considered to be a real-time task. Data packets are placed into the ready queue based on the processing time of the task. Moreover, each packet has a hop count number that is assigned randomly, and the packet with the highest hop count number is placed into the highest-priority queue. We run the simulation both for a specific number of zones, and levels in the network until data from a node in each zone or level reach BS. Simulation results are presented for both real-time data and all types of data traffic. Table I presents simulation parameters, and their respective values.

TABLE I
SIMULATION PARAMETERS, AND THEIR RESPECTIVE VALUES

Parameter	**Value**
Network Size	100m X 100m
Number of Nodes Maximum	200
Number of Zones	4 – 12
Base station position	55m X 101m
Transmission Energy Consumptions	50 *nJoule/bit*
Energy Consumption in free space or air	0.01 *nJoule/bit/m*2
Initial Node Energy	2 Joule
Transmission Speed	250*Kbps*
Propagation Speed	198 × 106*meter/sec*

Figures 4 and 5 illustrate the end-to-end data transmission delay of real-time tasks over a number of zones and levels, respectively. In both case, we observe that the proposed DMP scheduling scheme outperforms the existing FCFS, and Multilevel Queue scheduler. This is because the proposed scheduling scheme gives the highest priority to real-time tasks and also allows real-time data packets to preempt the processing of non-real time data packets. Thus, real-time data packets have lower data transmission delays. Figure 4 illustrates the *p*-values which are 0.0453 between FCFS and DMP schemes and 0.0137 between Multi-level queue and DMP schedulers.

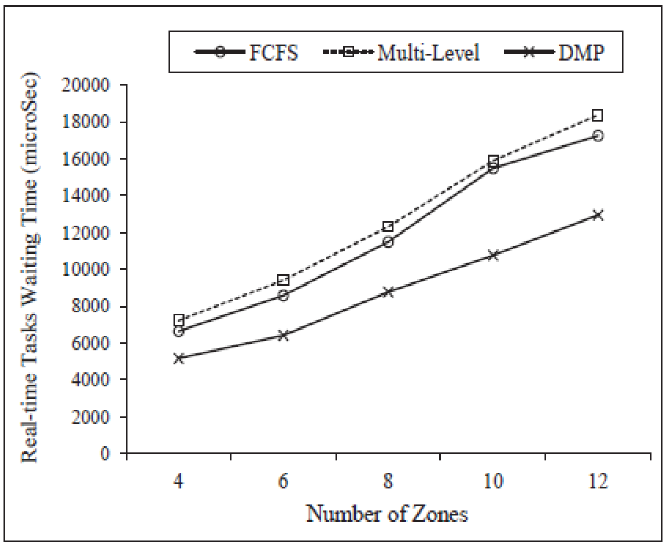

Fig. 4. End-to-end delay of real-time data over a number of zones.

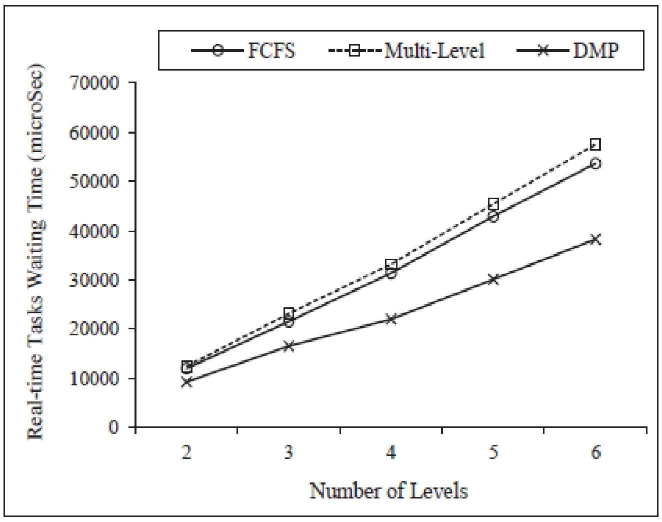

Fig. 5. End-to-end delay of real-time data over a number of levels

Similarly, Figures 6 demonstrate the end-to-end delay of all types of data traffic over a number of zones and levels, respectively. From these results, we find that the DMP task scheduling scheme outperforms FCFS, and Multilevel Queue scheduler in terms of end-to-end data transmission delay. This is because in the proposed scheme, the tasks that arrive from the lower level nodes are given higher priority than the tasks at the current node. Thus, the average data transmission delay is shortened. Figure 6 shows the *p*-values of student's *t*-test, which are 0.01156 between Multi-level and DMP schedulers, 0.000000005 between FCFS and DMP schedulers. Thus, DMP outperforms both FCFS and Multi-level queue schedulers at 95% confidence interval.

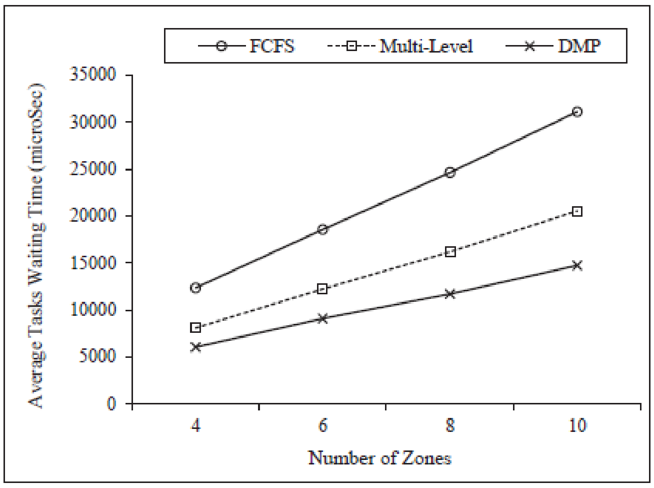

Fig. 6. End-to-end delay of all types of data over a number of zones.

Figures 7 demonstrate that the DMP task scheduler has better performance than the FCFS, and Multilevel Queue scheduler in terms of average task waiting time, both for real time tasks, and all types of tasks. We have already explained the possible reasons for this performance differences. We also perform student's *t*-test at a 95% confidence level and find the *p*-value to be less than 0.05 in most cases. This test validates our claim about the performance of the proposed DMP scheduling scheme.

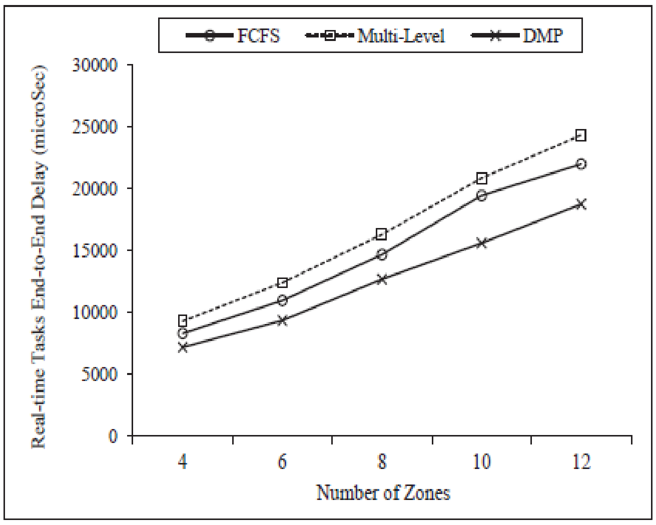

Fig. 7. Waiting time of real-time data over a number of zones.

We also measure, and compare the fairness of executing non-real-time task in terms of the total waiting time of non real- time tasks over total waiting time of all tasks. Figure 12 illustrates that the fairness index of DMP scheduling scheme is higher or better than that of the other two approaches. The number of levels in the network topology increases as the number of zones multiplies, which increases the average waiting time for non-real-time tasks over real-time tasks. Thus, the fairness index slightly decreases or remains almost same as the number of zones increases.

Using the concept of dynamic multilevel priority queues at each node, the proposed DMP task scheduling scheme allows different types of data packets to be processed based on their priorities. Since real-time, and emergency data should be processed with the minimum end-to-end delay, they are processed with the highest priority, and can preempt tasks with lower priorities located in the two other queues. On the other hand, in existing multilevel queue schedulers, a task with the highest hop count is given the highest priority. Hence, real-time tasks are prioritized over other task types only if their hop counts are higher than those of non-real-time tasks. Moreover, in FCFS and multilevel queue schedulers, the estimated processing time of a task is not considered when deciding the priority of a task. Thus, FCFS and Multilevel Queue schedulers exhibit longer task waiting times and end-to-end delays, in comparison to the DMP task scheduling scheme.

In the DMP task scheduling approach, the source of a data packet is used to define the priority of data packets other than real-time. The priority of non-real time data packet will be more if it is sensed at remote node rather than the current sending node. Moreover, when no real-time tasks are available, pr_3 tasks can preempt pr_2 tasks if they are in starvation for a long time. This allows the processing of different types of tasks with fairness. The memory is also dynamically allocated to three queues and the size of the highest-priority queue is usually smaller than the two other queues (Figure 3) since pr_1 real-time tasks do not occur frequently compared to non-real- time tasks. As the memory capacity of a sensor node is limited, this also balances memory usages. Moreover, tasks are mostly non-real-time and are processed in the pr_2 and pr_3 queues. Non-real-time tasks that a node x receives from the lower level nodes are known as non-real-time remote tasks and processed with higher priority (pr_2) than the non-real time local tasks that x senses. Thus, non-real-time remote tasks incur less average waiting time. In addition, the average waiting time will not be affected for real-time tasks that are processed using FCFS scheduling, since these real-time tasks occur infrequently with a short processing time.

Admittedly, one of the concerns regarding our proposed DMP task scheduling scheme pertains to its energy requirements. Indeed, the DMP task scheduling mechanism could be less energy efficient in comparison to the other two approaches since the DMP scheme requires a few more processing cycles to categorize and place the tasks into three different queues as well as for context saving and switching (for preemption). However, given the increased demand for WSN-based solutions that efficiently support real-time emergency applications and ensure them minimum average task waiting time and end-to-end delay, the proposed DMP task scheduling mechanism can be regarded as highly efficient.

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed a dynamic multilevel priority (DMP) packet scheduling algorithm for large scale wireless sensor networks considering the importance of prioritized processing of different types of tasks. The proposed scheme adapts well to the changing requirements of WSN applications and schedules real-time tasks with the highest priority ensuring a minimum end-to-end data transmission delay. It also schedules lowest priority tasks with fairness so that their delivery does not starve for a long period of time. Experimental results showed that the proposed DMP packet scheduling scheme has better performance than the FCFS and multi-level queue scheduler schemes in terms of end-to-end data transmission delay. As future work, we plan to consider the expiration deadline of packet transmission in task scheduling ensuring that tasks that have failed to meet the deadline are removed from the medium. This will eventually reduce the processing overhead and save the network's scarce bandwidth. Moreover, we will also consider more priority classes in the ready queue.

REFERENCES

[1] G. Anastasi, M. Conti, and M. Di Francesco, "Extending the lifetime of wireless sensor networks through adaptive sleep," *IEEE Trans. Industrial Informatics*, vol. 5, no. 3, pp. 351–365, 2009.

[2] G. Bergmann, M. Molnar, L. Gonczy, and B. Cousin, "Optimal period length for the CQS sensor network scheduling algorithm," in *Proc. 2010 International Conf. Netw. Services*, pp. 192–199.

[3] E. Bulut and I. Korpeoglu, "DSSP: a dynamic sleep scheduling protocol for prolonging the lifetime of wireless sensor networks," in *Proc. 2007 International Conf. Advanced Inf. Networking Appl.*, vol. 2, pp. 725–730.

[4] S. Chachra and M. Marefat, "Distributed algorithms for sleep scheduling in wireless sensor networks," in *Proc. 2006 IEEE International Conf. Robot. Autom.*, pp. 3101–3107.

[5] P. Guo, T. Jiang, Q. Zhang, and K. Zhang, "Sleep scheduling for critical event monitoring in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 2, pp. 345–352, Feb. 2012.

[6] F. Liu, C. Tsui, and Y. J. Zhang, "Joint routing and sleep scheduling for lifetime maximization of wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 9, no. 7, pp. 2258–2267, July 2010.

[7] J. Liu, N. Gu, and S. He, "An energy-aware coverage based node scheduling scheme for wireless sensor networks," in *Proc. 2008 International Conf. Young Comput. Scientists*, pp. 462–468.

[8] O. Khader, A. Willig, and A. Wolisz, "Distributed wakeup scheduling scheme for supporting periodic traffic in wsns," in *Proc. 2009 European Wireless Conf.*, pp. 287–292.

[9] B. Nazir and H. Hasbullah, "Dynamic sleep scheduling for minimizing delay in wireless sensor network," in *Proc. 2011 Saudi International Electron., Communications Photon. Conf.*, pp. 1–5.

[10] D. Shuman and M. Liu, "Optimal sleep scheduling for a wireless sensor network node," in *Proc. 2006 Asilomar Conf. Signals, Syst. Comput.*, pp. 1337–1341.

[11] N. Edalat, W. Xiao, C. Tham, E. Keikha, and L. Ong, "A price-based adaptive task allocation for wireless sensor network," in *Proc. 2009 IEEE International Conf. Mobile Adhoc Sensor Syst.*, pp. 888–893.

[12] H. Momeni, M. Sharifi, and S. Sedighian, "A new approach to task allocation in wireless sensor actor networks," in *Proc. 2009 International Conf. Computational Intelligence, Commun. Syst. Netw.*, pp. 73–78.

[13] F. Tirkawi and S. Fischer, "Adaptive tasks balancing in wireless sensor networks," in *Proc. 2008 International Conf. Inf. Commun. Technol.: From Theory Appl.*, pp. 1–6.

[14] X. Yu, X. Xiaosong, and W. Wenyong, "Priority-based low-power task scheduling for wireless sensor network," in *Proc. 2009 International Symp. Autonomous Decentralized Syst.*, pp. 1–5.

[15] W. Stallings, *Operating Systems*, 2nd edition. Prentice Hall, 1995.

[16] Y. Zhao, Q. Wang, W. Wang, D. Jiang, and Y. Liu, "Research on the priority-based soft real-time task scheduling in TinyOS," in *Proc. 2009 International Conf. Inf. Technol. Comput. Sci.*, vol. 1, pp. 562–565.

[17] E. M. Lee, A. Kashif, D. H. Lee, I. T. Kim, and M. S. Park, "Location based multi-queue scheduler in wireless sensor network," in *Proc. 2010 International Conf. Advanced Commun. Technol.*, vol. 1, pp. 551–555.

[18] E. Karimi and B. Akbari, "Improving video delivery over wireless multimedia sensor networks based on queue priority scheduling," in *Proc. 2011 International Conf. Wireless Commun., Netw. Mobile Comput.*, pp.1–4.