An Efficient Priority Packet Scheduling Algorithm for Wireless Sensor Network

Lutful Karim[1], Nidal Nasser[2], Tarik Taleb[3], and Abdullah Alqallaf[4]

[1]School of Computer Science, University of Guelph, Ontario, Canada
[2]College of Engineering, Alfaisal University, KSA
[3]NEC Europe Ltd
[4]Electrical Engineering Dept., College of Engineering, and Petroleum, Kuwait University
Email: lkarim@uoguelph.ca, nnasser@alfaisal.edu, talebtarik@ieee.org , al.qallaf@ku.edu.kw

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 three-class priority packet scheduling scheme. Emergency real-time packets are placed into the highest priority queue and can preempt the processing of packets at other queues. Other packets are prioritized based on the location of sensor nodes and are placed into two other queues. Lowest priority packets can preempt the processing of their immediate higher priority packets after waiting for a certain number of timeslots. Simulation results show that the proposed three-class priority packet scheduling scheme outperforms FCFS and multi-level queue schedulers in terms of end-to-end data transmission delay.

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

I. INTRODUCTION

Scheduling packets, interchangeably used as task scheduling in this paper, at sensor nodes are significantly important for prioritizing applications of Wireless Sensor Networks (WSNs). For instance, real-time applications shall have higher priority than non-real time applications. However, most existing task scheduling schemes use First-Come-First-Served (FCFS) schedulers that process data packets according to their arrival time. It is not considered efficient in terms of end-to-end delay and sensors energy consumptions. Hence, intermediate nodes require changing the delivery order of data in their ready queue based on priorities and delivery deadline in the existing scheduling techniques. Additionally, existing task scheduling algorithms of WSNs do not adapt to traffic dynamics. For instance, for real-time applications, a real-time priority scheduler is statically used and cannot be changed during the operation of WSN applications.

Hence, we introduce three-class priority packet scheduling scheme, where each node maintains three levels into its queue for three different types of data. This is because we classify data as (i) real-time (highest or priority1), (ii) non-real-time remote packets, i.e., packets that arrive from the sensors nodes at lower levels (priority2), and (iii) non-real-time local packets, i.e., the packets that are sensed at the current sensor node (lowest priority3). Non-real-time data packets are classified based on the location of sensor nodes to reduce the end-to-end delay of data packets that are generated at different locations. Real-time data can preempt data at other queues. If there is no data available at the real-time highest priority1 queue, then data at the priority2 queue are processed. If the priority2 data are processed at a node for α consecutive timeslots, the lowest priority3 data can then preempt the priority2 data. Thus, the proposed scheduling scheme achieves fairness via an appropriate setting of α. 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. Tasks 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. The proposed packet scheduling scheme is more suitable for heterogeneous WSN applications where both real-time and non-real time data are transmitted. For instance, it can be used in a smart home to monitor temperature and humidity (non-real time) and health condition of elderly people (real-time). Whenever a sensor transmits data to BS through intermediate nodes, the packet type is inserted into the packet header.

The remainder of this paper is organized as follows. Section II presents a literature study of several existing packet scheduling algorithms of WSNs. In Section III, we present the proposed three-class priority packet scheduling algorithm with its general working principle, and pseudo-code. Section IV analyzes the performance of the proposed scheduling algorithm in terms of end-to-end delay for different traffic types. In Section V, the performance of the packet scheduling algorithm is simulated and compared with existing FCFS, and multi-level queue scheduler algorithms [4]. Finally, conclusions and future work are presented in Section VI.

II. LITERATURE REVIEW

The task scheduling scheme presented in [6] is used in TinyOS - the widely used operating system in Wireless Sensor Networks (WSNs). The TinyOS scheduling mechanism is classified as cooperative and preemptive. In cooperative scheduling, if a higher priority task t_2 arrives at the ready queue after a task t_1 has started its execution, t_2 has to wait until t_1 finishes its executions. Co-operative scheduling schemes are based on Earliest Deadline First (EDF) mechanism [6], whereby priority of a task is adjusted at the ready queue and the shortest deadline task has highest priority and vice versa. Cooperative schedulers are suitable for applications with limited system resources and no hard real-time requirements. In preemptive scheduling algorithms, higher priority tasks can preempt lower priority tasks by saving the context of lower priority tasks if they are already running. Preemptive schedulers are based on Emergency Task First Rate Monotonic (EF-RM) [6] - an extension of Rate Monotonic (RM), a static priority scheduling, where shortest deadline task has the highest priority. Though task scheduling mechanisms of TinyOS is simple and used extensively in sensor nodes, they cannot be applied to all applications since due to the long execution time of certain tasks the real-time tasks might be placed into starvation.

The works in [1] presents real-time communication architecture for large scale sensor networks using priority-based scheduler. Data packets that travel the longest distance from a source node to BS but have the shortest deadline have the highest priority. If the deadline of a data packet expires before reaching BS, the data packet is simply discarded. Though this approach reduces network traffic and data processing overhead, it is not efficient since it consumes resources such as memory, computation power and increases processing delay. In [7], RACE, a packet scheduling policy and routing algorithm for real-time large scale sensor networks, is proposed. RACE uses EDF scheduling algorithm and a prioritized Medium Access Control (MAC) scheme. However, local prioritization at each individual node in RACE is not sufficient because packets from different senders can compete against each other for a shared radio channel.

Though priority scheduling schemes solve the problem of starvation, they do not ensure that the local data processing will get a fair chance if the data forwarding operations have higher priority and vice versa. The task scheduling scheme introduced in [9] proposes a scheduling mechanism that solves “unfair sensing problem” and properly distributes the high priority tasks (network tasks) and low priority tasks (sensing). During a percentage of each duty cycle P, the priority of local tasks is less than the priority of network tasks and during the remaining time of the duty cycle, the priority of local tasks becomes higher than the priority of network tasks. The multi-level-queue scheduler scheme, proposed in [4], uses different numbers of queues according to the location of sensor nodes in the network. If the lowest level is L_k, nodes that are located at level L_{k-1} have only one queue but there are two queues for nodes at level L_{k-2}. When a node receives a packet, it decides the packet's priority according to the hop count of the packet and accordingly sends it to the appropriate queue.

III. PROPOSED SCHEDULING ALGORITHM

In this section, we present the proposed three-class priority packet scheduling mechanism for WSN.

A. General Working Principle

We consider three-level 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 pr1, 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. Figure 1 illustrates the three-class priority packet scheduling scheme.

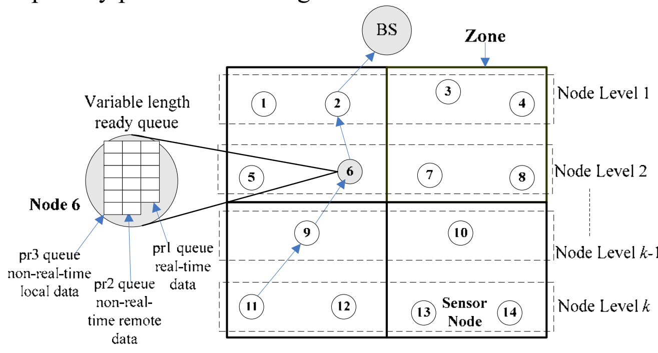

Figure 1: Three-class priority packet scheduling scheme.

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 ready queue for non-preemptive priority scheduler is expected to be smaller than that of the preemptive priority scheduler. The idea behind this is that the highest priority real-time emergency tasks are processed with a minimum possible delay. They are placed in non-preemptive priority1 tasks queue and can preempt the currently running data. Thus, they are expected not to reserve a queue location for a longer period of time. Moreover, the number of preemptions will be low since real-time data is in general small in number. On the other hand, non-real-time packets that arrive from the sensor nodes at lower level are placed in the preemptive priority2 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 priority3 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 data to reach the BS. For two tasks of the same level, the smaller task will have higher priority.

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/emergency 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 this section, we present the pseudo-code of the proposed three-class priority packet scheduling algorithm.

```
Algorithm: Three-class priority data scheduling scheme
while $task_{k,i}$ received by $node_i$ at level $k$ i.e., at $l(k)$ do
  if $task_{k,i}$ type = real-time then
      put $task_{k,}$ in $pr_1$ queue
  else if $node_i$ is not at lowest levels //three levels in queue
      if $task_{k,i}$ is not local then
          put $task_{k,i}$ in $pr_2$ queue //non-real time remote tasks
      else
          put $task_{k,i}$ in $pr_3$ queue //non-real-time local task
      end if
  else //only two levels in queue
      put $task_{k,i}$ in $pr_2$ queue //non-real-time local task
  end if
Assume, the duration of a timeslot at $l(k) \leftarrow t(k)$
Data sensing time of $node_i$ at $l(k) \leftarrow senseTime_k(t)$
$\therefore$ Remaining time after data sensing,
              $t_1(k) \leftarrow t(k) - senseTime_k(t)$
  Let, total real-time tasks for $node_i$ at level $l(k) \leftarrow n_k(pr_1)$
  Let, $procTimepr_1(k) \leftarrow \sum_{j=1}^{n_k(pr_1)} procTime(j)$
  if $procTimepr_1(k) < t_1(k)$ then
    All $pr_1$ tasks of $node_i$ at $l(k)$ are processed as FCFS
    Remaining time $t_2(k) \leftarrow t_1(k) - procTimepr_1(k)$
    Let, total $pr_2$ tasks for $node_i$ at level $l(k) \leftarrow n_k(pr_2)$
    Let, $procTimepr_2(k) \leftarrow \sum_{j=1}^{n_k(pr_2)} procTime(j)$
     if $procTimepr_2(k) < t_2(k)$ then
       All $pr_2$ tasks are processed as FCFS
       $pr_3$ tasks are processed as FCFS for
       the remaining time, $t_3(k) \leftarrow t_2(k) - procTimepr_2(k)$
     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 $t(k) \leq procTimepr_2(k)$ then
    $pr_2$ tasks are preempted at $\alpha+1, \alpha+2, \ldots, \alpha+j$ timeslots
    by $pr_3$ tasks
    if $pr_1$ task arrives during any of $\alpha+1, \alpha+2, \ldots, \alpha+j$
    timeslots then
    $pr_3$ tasks are preempted & $pr_1$ 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 $node_i$. 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 pr2 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. If the maximum number of active nodes in the network is n_a and maximum number packets in the ready queue of each active node is n_t the scheduling algorithm runs in $O(n_a \times n_t)$ or $O(n^2)$ whenever $O(n_a) \approx O(n_t) \approx O(n)$.

IV. PERFORMANCE ANALYSIS

In the following, we formulate the average end-to-end delay for transmitting different priority tasks to BS.

Real-time Priority 1 Queue Data

Let's consider a node x, residing at level $l(k)$ and sensing a real-time, emergency event e.g., fire detection. This node transmits the priority1 data to BS through $l(k$ -1) intermediate nodes. We consider the following scenario whereby every time a real-time data reaches a neighboring active node at an upper level y, 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. Transmission delay that is required to place a real-time data from a node into the medium is equal to $\frac{data_{pr1}}{s_t}$. The propagation time or delay to transmit data from the source to destination can be formulated as $\frac{d}{s_p}$.

Considering the above mentioned scenario end-to-end delay for sending a real-time data satisfies the following inequality.

$$delay_{pr1} \geq l_k \times \left(\frac{data_{pr1}}{s_t} + pr1_{proc}(t) \right) + \frac{d}{s_p} + (l_k \times t_{overhead}) \quad (1)$$

where $data_{pr1}$denotes the real-time data size, s_t denotes the data transmission speed, d is the distance from the source node to BS, where $d = \sum_{i=1}^{l(k)} d_i$, s_p denotes the propagation speed over the wireless medium, $pr_{1proc}(t)$ is the processing time of real-time tasks at each node, and $t_{overhead}$ is an overhead in terms of context switching and queuing time (including time for preemption). However, a real-time task t_1 has to wait if there is n_{pr1}, 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 n_{pr1} real-time tasks ahead of it,

$$delay_{t_1} \geq \sum_{i=1}^{n_{pr1}} (delay_{pr_1})_i \quad (2)$$

Non-real time Priority 2 Queue Data

Tasks at pr_2 queue can be preempted by real-time ones. Taking the scenario of Fig. 1 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 data are available at the pr_2 queue at nodes 9, 6 and 2. Since one real-time data is available at the pr_1 queue of nodes 9, 6, and 2, real-time data 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

$$l_k \times \left(\frac{data_{pr1}}{s_t} + \frac{data_{pr2}}{s_t} + pr_{1proc}(t) + pr_{2proc}(t) \right) + \frac{d}{s_p} + (l_k \times t_{overhead}) \tag{3}$$

Non-real time Priority 3 Queue Data

In the best case, when no data is available at the pr1 and pr_2 queues, the end-to-end delay of the pr_3 data 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 data. We assume that the pr_3 queue tasks are processed by preempting pr_2 queue data if for α consecutive timeslots there is no task at the pr1 queue but there are tasks available at the pr_2 queue. Let $t(k)$ denotes the length of a timeslot of nodes at level $l(k)$. The transmission time or delay to place pr3 data from a node into the wireless medium is equal to $\frac{data_{pr3}}{s_t}$. However, during the processing of the pr_3 queue data, these data can be preempted by real-time data. They are processed again after the completion of real-time tasks. Thus, the end-to-end delay for processing pr_3 data will be exceeding

$$\alpha \times t(k) + l_k \times \left(\frac{data_{pr3}}{s_t} + pr_{3proc}(t) \right) + \frac{d}{s_p} + (l_k \times t_{overhead}) \tag{4}$$

V. SIMULATION RESULTS

We simulate the proposed three-class priority packet scheduling algorithm using C programming language and compare its performance against that of the FCFS and multi-level queue scheduling algorithms.

Table I. Simulation parameters and their values

Parameter	Value
Network Size	100 x 100 Meter
Number of Nodes	Maximum 200
Number of Zones	4 - 12
Base Station Position	55 x101 Meter
Transmission Energy Consumption	50 nJoule/bit
Energy Consumption in free space/air	0.01 nJoule/bit/m^2
Initial Node Energy	2 Joule
Transmission Speed	250 Kbps
Propagation Speed	198x10^6 m/sec

Initially, we measure the performance of real-time tasks in terms of end-to-end delay. This is because fast data delivery for real-time data packets is significantly important for WSN. We also evaluate the performance of the proposed scheduling algorithm for both real-time and non-real time traffic. In the simulation, the network size is assumed to be 100 meter x 100 meter. Nodes are distributed uniformly over the simulated zones. The ready queue of each node can hold a maximum of 50 tasks. Each task has a Type ID, which identifies its type. The simulation parameters and their respective values are presented in Table I. Each task has a hop count number that is randomly assigned to it. We consider nodes with the same hop count being located at the same level. A packet is placed at the highest priority1 queue if the type field indicates real-time task. Otherwise, the task with the highest hop count number is placed into the highest priority queue. The highest hop count represents tasks from a node at the farthest level. We run the simulation for a specific number of zones and levels until data reach the BS.

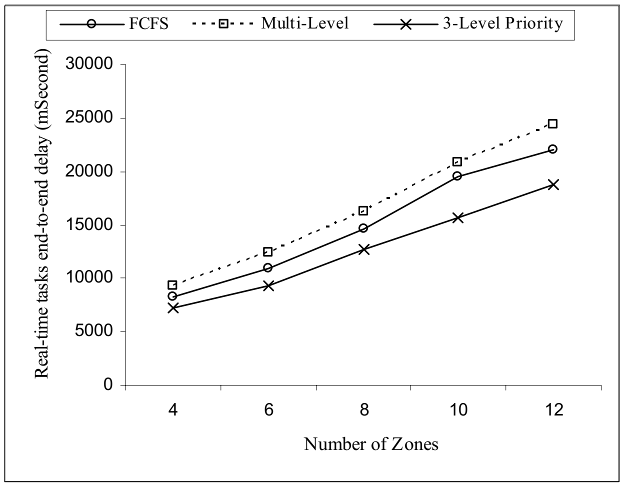

Figure 2: End-to-end delay of real-time data over no. of zones.

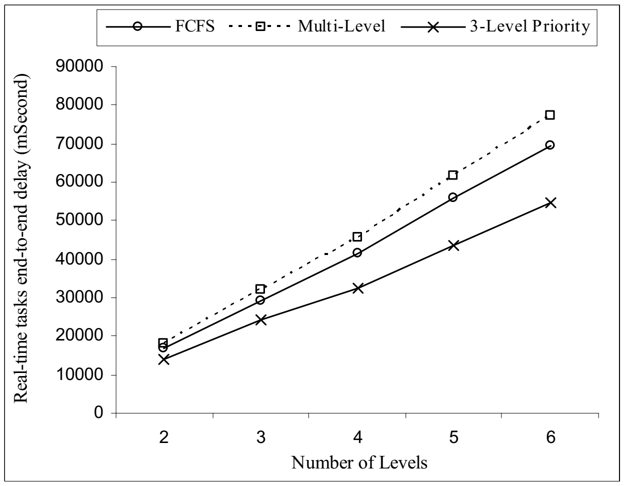

Figure 3: End-to-end delay of real-time data over no. of levels.

Figs. 2 and 3 illustrate that the proposed three-class priority packet scheduler has better performance for real-time tasks than FCFS and multi-level queue scheduler in terms of end-to-end delay. 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.

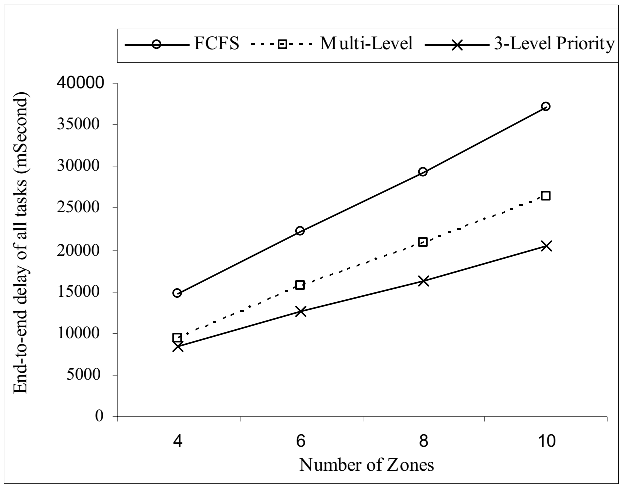

Figure 4: End-to-end delay of tasks over a number of zones.

Figs. 4 and 5 illustrate that when the proposed three-class priority packet scheduler is used, the end-to-end delay of all tasks (both real-time and non-real-time) at the active nodes of the network is much shorter compared to that experienced in case of FCFS and multi-level queue scheduler. This is because in the proposed scheme, the tasks that arrive from lower level nodes are given higher priority than tasks at the current node. Thus, the average data transmission delay is shortened. We also perform student's *t*-test at 95% confidence level to validate the performance of the proposed scheduler and in all cases we find that the p-value < 0.05. Hence, the proposed scheduler outperforms FCFS and multi-level queue scheduling approaches in terms of end-to-end delay.

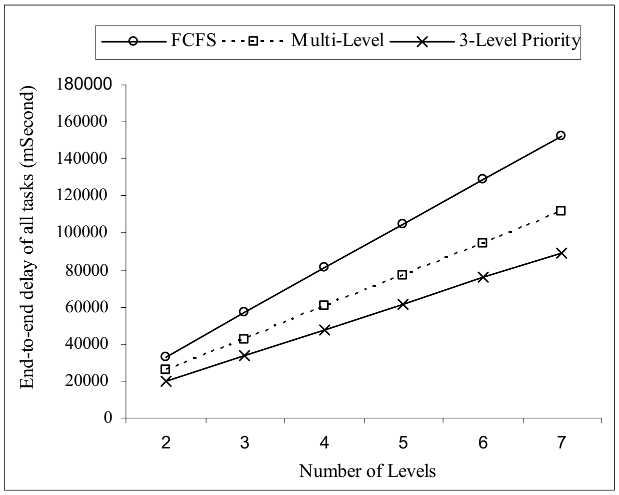

Figure 5: End-to-end delay of tasks over a number of levels

Admittedly, one of the concerns regarding the proposed three-class priority packet scheduling scheme pertains to its energy requirements. Indeed, the proposed scheduling mechanism could be less energy efficient in comparison to the other two approaches since it 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 three-priority task scheduling mechanism can be regarded as highly efficient.

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed a three-class priority 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 three-class priority packet scheduling scheme has better performance than the FCFS and multi-level queue scheduler schemes [4] 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.

REFERENCE

[1] C. Lu, B.M. Blum, T.F. Abdelzaher, J.A. Stankovic, and T. He, "RAP: a Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks", *Technical Report*. University of Virginia.

[2] C. Wang and W. Wu, "A Light-weighted Operating System with Deadlock Prevention Strategy for Wireless Sensor Nodes". In *Proceedings of the 2009 WRI international Conference on Communications and Mobile Computing*, Washington, DC, pp. 578-583

[3] M. Chitnis, P. Pagano, G. Lipari and Yao Liang, "A survey on bandwidth resource allocation and scheduling in wireless sensor networks," in *Network-Based Information Systems, 2009. NBIS '09. International Conference on,* 2009, pp. 121-128.

[4] Eun-Mook Lee, A. Kashif, Dong-Hyun Lee, In-Tae Kim and Myong-Soon Park, "Location based multi-queue scheduler in wireless sensor network," in *Advanced Communication Technology (ICACT), 2010 the 12th International Conference on,* 2010, pp. 551-555.

[5] Hyun Jung Choe, "QoS-aware data report scheduling in heterogeneous wireless sensor networks," in *Pervasive Computing and Communications, 2009. PerCom* 2009*. IEEE International Conference on,* 2009, pp. 1-2.

[6] Min Yu, Si Ji Xiahou and Xin Yu Li, "A survey of studying on task scheduling mechanism for TinyOS," in *Wireless Communications, Networking and Mobile Computing WiCOM '08*, pp. 1-4.

[7] K. Mizanian, R. Hajisheykhi, M. Baharloo and A. H. Jahangir, "RACE: A real-time scheduling policy and communication architecture for large-scale wireless sensor networks," in *Communication Networks and Services Research Conference, 2009. CNSR '09. Seventh Annual,* 2009, pp. 458-460.

[8] H. Momeni, M. Sharifi and S. Sedighian, "A new approach to task allocation in wireless sensor actor networks," in *Computational Intelligence, Communication Systems and Networks, 2009. CICSYN '09. First International Conference on,* 2009, pp. 73-78.

[9] F. Tirkawi and S. Fischer, "Adaptive tasks balancing in wireless sensor networks," in *Information and Communication Technologies: From Theory to Applications, ICTTA 2008. 3rd International Conference on,* 2008, pp. 1-6.

[10] Wei Dong, Chun Chen, Xue Liu, Kougen Zheng, Rui Chu and Jiajun Bu, "FIT: A Flexible, Lightweight, and Real-Time Scheduling System for Wireless Sensor Platforms," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 21, pp. 126-138, 2010.

[11] Weitao Xu, Xiaohong Hao and Ping Zhang, "Research of task assignment and scheduling algorithms in wireless sensor networks," in *Mechatronics and Automation ICMA 2007. International Conference on,* 2007, pp. 751-756.