CIRCULAR WAIT AND PREEMPTIVE DMP PACKET SCHEDULING SCHEME FOR WIRELESS SENSOR NETWORK

[1]P. Palanisamy, [2]N.Jeevitha,
[1] Research Scholar, Department of ECE, Sriguru Institute of Technology, Coimbatore, India
[2]Assistant Professor, Department of ECE, Sriguru Institute of Technology, Coimbatore, India
[1]pvpalanisami@gmail.com, [2]jeevithan@gmail.com

Abstract: Scheduling different types of packets, such as real-time and non-real-time data packets, at sensor nodes with resource constraints in Wireless Sensor Networks (WSN) is of vital importance to reduce sensors energy consumptions and end-to-end data transmission delays. Most of the existing packet-scheduling mechanisms of WSN use First Come First Served (FCFS), non-preemptive priority and preemptive priority scheduling algorithms. These algorithms incur a high processing overhead and long end-to-end data transmission delay due to the FCFS concept, and improper allocation of data packets to queues in multilevel queue scheduling algorithms. In the proposed scheme, Circular Wait and Preemptive Dynamic Multilevel Priority (DMP) Packet Scheduling Scheme for Wireless Sensor Network 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. Data packets sensed by nodes at different levels are processed using a TDMA scheme.

Keywords*:* Wireless sensor network, packet scheduling, preemptive priority scheduling, non-preemptive priority scheduling, Dynamic Multilevel Priority scheduling, real-time, non-real-time, data waiting time, FCFS.

1. INTRODUCTION

Among many network design issues, such as routing protocols and data aggregation, that reduce sensor energy consumption and data transmission delay, packet scheduling (interchangeably use as task scheduling) at sensor nodes is highly important since it ensures delivery of different types of data packets asked on their priority and fairness with a minimum latency. For instance, data sensed for real-time applications have higher priority than data sensed for non-real-time applications. That the schedule the processing of data packets available at a sensor node and also reduces energy consumptions. Indeed, most existing Wireless Sensor Network operating systems use First Come First Serve 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.

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.

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. 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. Furthermore most existing packet scheduling algorithms of WSN are neither dynamic nor suitable for large scale application since this scheduler

2. RELATED WORK

2.1 Factor: Deadline

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

- **First come First Served:** Most existing WSN applications use First Come First Served 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 but data from nearby neighbouring 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:** 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.

2.2 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 t1 starts execution, task t carries on even if a higher priority packet than the currently running packet t. It arrives at the ready queue. Thus t has to wait in the ready queue until the execution of t 1.
- **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.

2.3 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.

2.4 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. The main concept behind multi-level queue scheduling algorithms.

3. ROPOSED METHODOLOGY

A proposed Circular Wait and Preemptive DMP Packet Scheduling Scheme for WSNs in which sensor nodes are virtually organized into a hierarchical structure. Nodes that have the same hop distance from the BS are considered to be located at the same hierarchical level.

Data packets sensed by nodes at different levels are processed using a TDMA scheme. For instance, nodes that are located at the lowest level and one level upper to the lowest level can be allocated timeslots 1 and 2, respectively. Each node maintains three levels of priority queues shown in Fig 1. This is because we classify data packets as real-time (priority 1), (ii) non-real-time remote data packet that are received from lower level nodes (priority 2), and (iii) non-real-time local data packets that are sensed at the node itself (priority 3). Nonreal- time data traffic with the same priority are processed using the shortest job first scheduler scheme since it is very efficient in terms of average task waiting time.

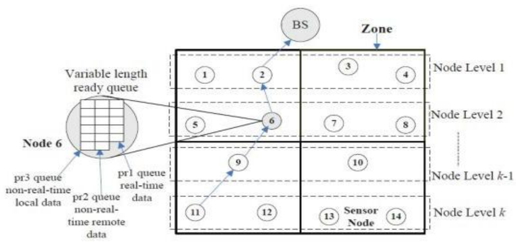


Figure 1: Proposed Packet Scheduling

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 preempt able 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, nonreal-time packets that arrive from the sensor nodes at lower level are placed in the preempt able 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., interms of data size) will have higher priority.

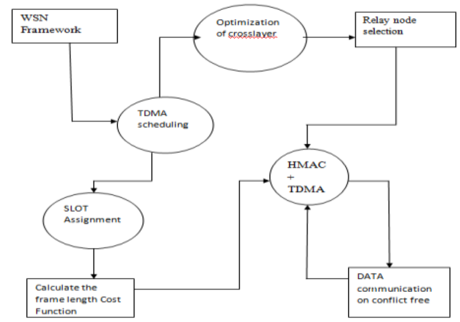


Figure 2: Data flow diagram

We present a new MAC protocol, which is referred to as hybrid MAC, which is suitable for WSNs in terms of energy efficiency, latency, and design complexity.

HMAC combines channel-allocation schemes from existing contention-based and time- division multiple-access based MAC protocols to allow the realization of tradeoffs between different performance metrics.

It uses a short slotted frame structure and a novel wakeup scheme to achieve high-energy performance, low delivery latency, and improved channel utilization.

Our proposed protocol combines energy efficient features of the existing contention-based and time-division multiple accesses based MAC protocols and adopts a short frame structure to expedite packet delivery.

HMAC is simple and scalable since each node does not have to maintain neighborhood information.

HMAC provides routing layer course- grained quality of-service support at the MAC layer. To the best of our knowledge, very few existing MAC layer works handle such QoS issues in WSNs. Quality of service-aware medium access control assigns each flow a channel-access priority to reduce the queuing delay for high-priority flows but it still suffers from a long end-to-end delay.

The MAC protocols presented in reduce the endto-end delivery latency while increasing control overhead without considering different performance demands between flows.

Pseudocode for scheduling a slot for all each link (*i*, *j*) $\in E[G]$ do

1. $F[i, j] \leftarrow 0$
2. $S[i, j] \leftarrow 0$

3. end for
4. $Q[G] \leftarrow$ NIL
5. while More than one link $(i, j) \in E[G]$ do
6. while More than one link $(i, j) \in$
$E[G]$where
$S[i, j] = 0$
7. do
8. Randomly select a link $(i, j) \in E[G]$
such that
9. $F[i, j] = 0$ and $S[i, j] = 0$
10. Add link (i, j) to $Q[G]$
11. UPDATE NETWORK
CONFIGURATION(G,E,
Q, S)
12. end while
13. Select a link $(i, j) \in Q[G]$ such that $D[i,$
$j]/R[i, j]$ is minimal
14. for each required slot m in Mi do
15. Try assigning slot s = 1;
16. while any of the 3 interference criteria is not
17. satisfied do
18. Try assigning the next slot s[i,j] = s + 1;
19. end while
20. Assign slot s to required slot m of node i;
21. end for
22. $F[i, j] \leftarrow 0$ and $D[i, j] \leftarrow 0$
23. for all link $(m, n) \in Q[G]$ where $D[m, n]$ _= 0 **do**
24. $D[m, n] \leftarrow D[m, n] - D[i, j] \times R[m,$
$n]/R[i, j]$
25. end for
26. UPDATE NETWORK
CONFIGURATION(G,E, Q, S)
27. end while

4. PERFORMANCE EVALUATION

The simulation model is implemented using the TCL language. It is used to evaluate the performance of the proposed Circular Wait and Preemptive 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.

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	$250Kbps$
Propagation Speed	$198 \times 10^6 meter/sec$

Figure 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.

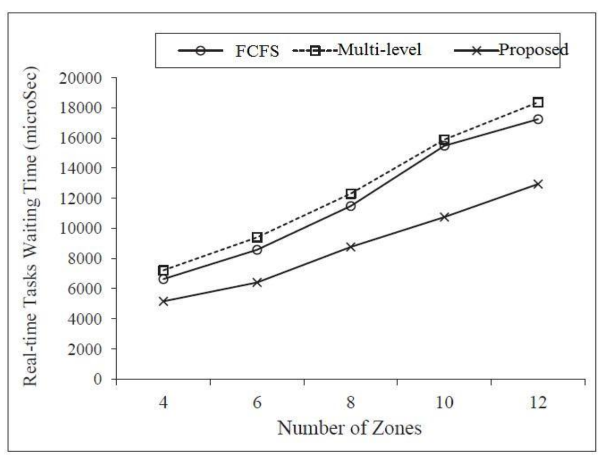


Figure 3: End-to-end delay of real-time data over a number of zones

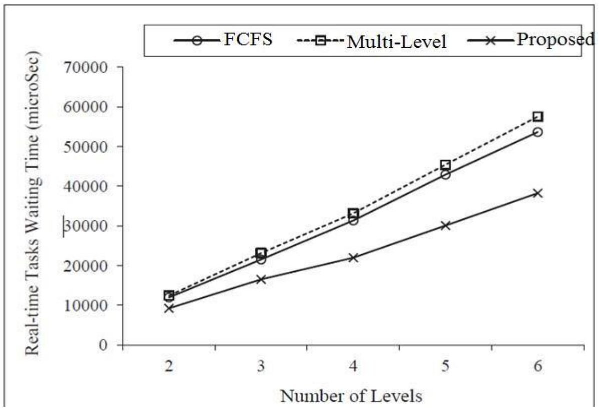


Figure 4: End-to-end delay of real-time data over a number of levels

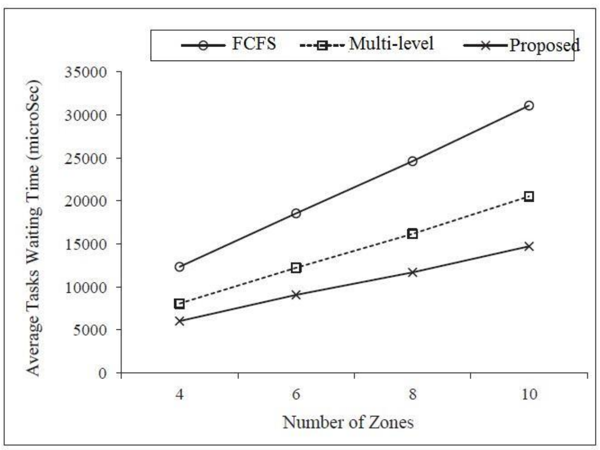


Figure 5: End-to-end delay of all types of data over a number of zones.

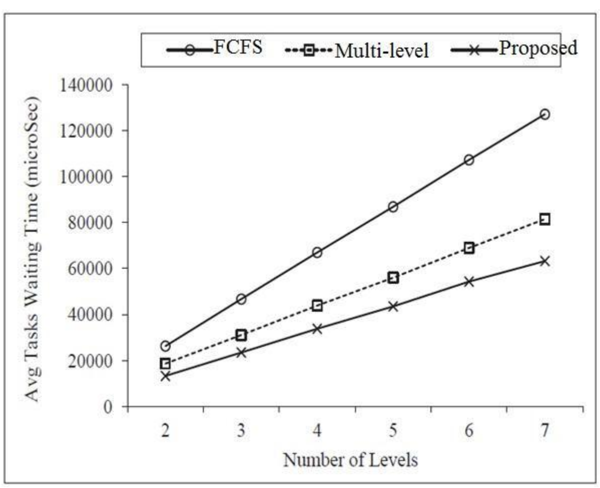
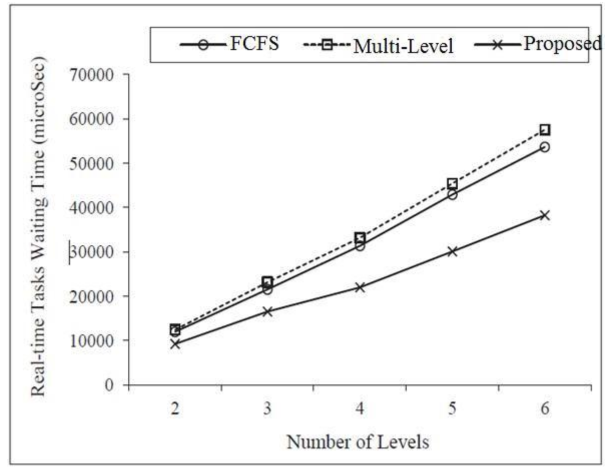


Figure 6: End-to-end delay of all types of data over a number of levels



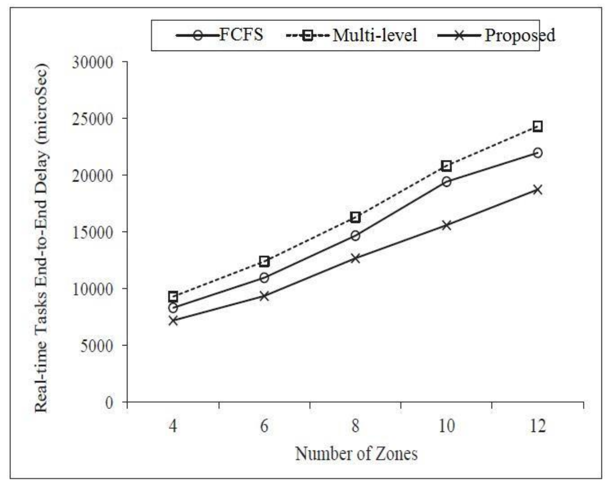


Figure 7: Waiting time of real-time data over a number of zones.

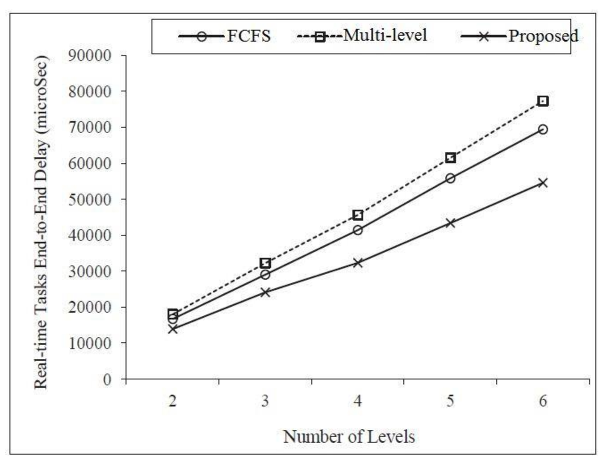


Figure 8: Waiting time of real-time data over a number of levels.

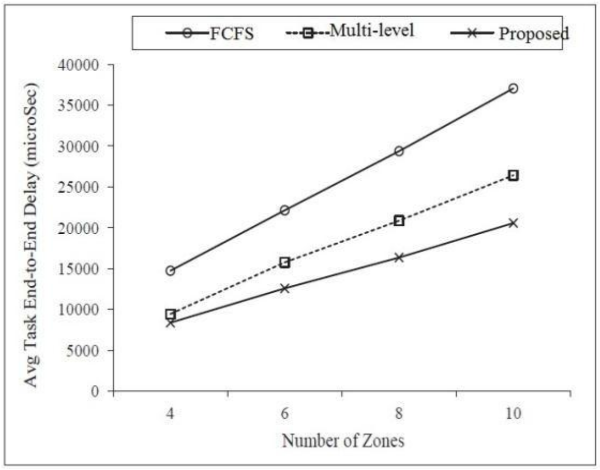


Figure 9: Waiting time of all types of data over a number of zones

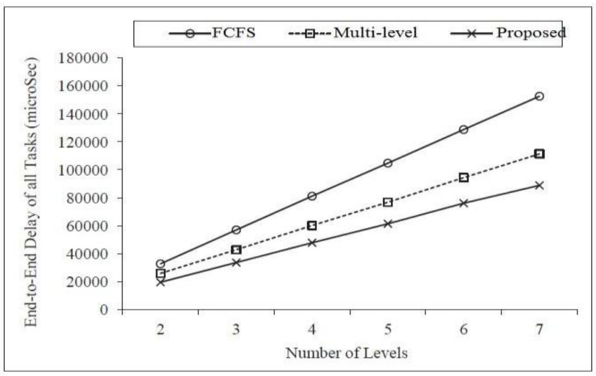


Figure 10: Waiting time of all types of data over a number of levels.

We also validate these results using student's t-test at 95% confidence level. Figure 3 illustrates the pvalues which are 0.0453 between FCFS and Proposed schemes and 0.0137 between Multi-level queue and Proposed schedulers. Similarly, Figures 5 and 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 Circular Wait and Preemptive 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. Fig 5 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, Proposed outperforms both FCFS and Multi-level queue schedulers at 95% confidence interval. Figures 7 - 10 demonstrate that the Proposed task scheduler has better performance than the FCFS, and Multilevel Queue scheduler in terms of average task waiting time, both for realtime 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 scheduling scheme. Using the concept of three-level priority queues at each node, the proposed Circular Wait and Preemptive 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, realtime tasks are prioritized over other task types only if their hop counts are higher than those of non-realtime 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-toend delays, in comparison to the Circular Wait and Preemptive DMP task scheduling scheme. Furthermore, the average waiting time of a task contributes largely to the experienced end-to-end data transmission delay, hence the strong correlation between the results of Figures 9 and 6. In the Proposed 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, pr3 tasks can preempt pr2 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 2) since pr1 real-time tasks do not occur frequently compared to nonreal- 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 pr2 and pr3 queues. Non-real-time tasks that a node x receives from the lower level nodes are known as non-realtime remote tasks and processed with higher priority (pr2) than the non-real time local tasks that x senses.

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed a Circular Wait and Preemptive Dynamic Multilevel Priority packet scheduling scheme for Wireless Sensor Networks. The scheme uses three-level of priority queues to schedule data packets based on their types and priorities. It ensures minimum end-to-end data transmission for the highest priority data while exhibiting acceptable fairness towards lowest-priority data. Experimental results show that the proposed packet scheduling scheme has better performance than the existing FCFS and Multilevel Queue Scheduler in terms of the average task waiting time and end-to- end delay. The proposed scheme, we envision assigning task priority based on task deadline instead of the shortest task processing time. To reduce processing overhead and save bandwidth, we could also consider removing tasks with expired deadlines from the medium. Furthermore, if a real time task holds the resources for a longer period of time, other tasks need to wait for an undefined period time, causing the occurrence of a deadlock. This deadlock situation degrades the performance of task scheduling schemes in terms of end-to-end delay. Hence, we would deal with the circular wait and preemptive conditions to prevent deadlock from occurring.

This project may have the way for the evolution of new methods for multi-target tracking based on other optimization techniques that may be more energy and power efficient than this method.

REFERENCES

[1] R. Swain, R. C. Hansdah, and V. K. Chouhan, “An energy aware routing protocol with sleep scheduling for wireless sensor networks,” in Proc. 2010 IEEE International Conf. Adv. Inf. Netw. Appl., pp. 933–940.

[2] B. Zeng, Y. Dong, and D. Lu, “Cooperation based scheduling algorithm in wireless multimedia sensor networks,” in Proc. 2011 International Conf. Wireless Commun., Netw. Mobile Comput., pp.1–4.

[3] 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.

[4] 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.

[5] J. Liu, N. Gu, and S. He, “An energyaware coverage based node scheduling scheme for wireless \sensor networks,” in Proc. 2008 International Conf. Young Comput. Scientists, pp. 462–468.

[6] L. Karim, N. Nasser, and T. El Salti, “Efficient zone-based routing protocol of sensor network in agriculture monitoring systems,” in Proc. 2011 International Conf. Commun. Inf. Technol., pp. 167–170.

[7] M. Yu, S. J. Xiahou, and X. Y. Li, “Asurvey of studying on task scheduling mechanism for TinyOS,” in Proc. 2008 International Conf. Wireless Commun., Netw. Mobile Comput., pp. 1–4.

[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] 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.

[10] 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.