A Feasible Approach to Assigning System Components to Hybrid Task Sets in Real-Time Sensor Networking Platforms

Kyunghoon Jung¹, Byoungsoon Kim¹, Changsoo Kim², and Sungwoo Tak^{1,*}

¹School of Computer Science and Engineering, Pusan National University, San-30,
Jangjeon-dong, Geumjeong-gu, Busan, 609-735, Republic of Korea
swtak@pusan.ac.kr

²Pukyong National University, Dept. of Computer Science, Busan, Republic of Korea
cskim@pknu.ac.kr

Abstract. This paper studies an efficient periodic and aperiodic task decomposition technique for real-time sensor networking platforms from two perspectives. First, wireless sensor networking platforms available in literature have not addressed the concept of real-time scheduling for hybrid task sets where both types of periodic and aperiodic tasks exist. Second, individual system components in real-time sensor networking platforms must be assigned to either a periodic or aperiodic task in order to provide the concept of fully optimal real-time scheduling given real-time requirements of user applications. The proposed approach guarantees the feasible scheduling of hybrid task sets while meeting deadlines of all the periodic tasks. A case study based on real system experiments is conducted to illustrate the application and efficiency of the proposed technique. It shows that our method yields efficient performance in terms of guaranteeing the deadlines of all the periodic tasks and aiming to provide aperiodic tasks with good average response time.

Keywords: Networking Platform, Communication Software Architecture, Real-Time Scheduling.

1 Introduction

In order to realize real-time wireless sensor networking platforms in many different kinds of critical applications, three fundamental software mechanisms for real-time processing have to be studied. First is the principal division of tasks and functionalities in a sensor node, such as kernel architecture and communication protocol stacks. Second, the real-time sensor software architecture on a single node has to be extended to a network architecture, where the division of tasks between systems, not only on a single node, becomes the relevant question – for example, how to structure periodic or aperiodic tasks for communication protocol components. The last is the question of how to design periodic and aperiodic task decomposition technique which yields efficient performance in terms of guaranteeing the deadlines of all the periodic tasks and

* Corresponding author.

aiming to provide aperiodic tasks with good average response time. Existing sensor networking platforms, such as TinyOS [1] and DCOS [2], have been focused on solving the efficient exploitation of limited resources. Envisioned application scenarios applied in sensor networking platforms need to activate real-time sensing as well as provide good average response time with task communication. In this paper, we propose an efficient periodic and aperiodic task decomposition technique for real-time sensor networking platforms from two perspectives. First, wireless sensor networking platforms available in literature have not addressed the concept of real-time scheduling for periodic and aperiodic tasks. Second, individual system components in real-time sensor networking platforms must be assigned to either a periodic or aperiodic task in order to provide the concept of fully optimal real-time scheduling given real-time requirements of user applications.

2 Real-Time Sensor Networking Platform

Fig. 1 shows hybrid task sets consisting of periodic and aperiodic tasks in the proposed real-time sensor networking platforms. In User tasks, sensing tasks which activate sensors and transmit/receive the sensing data periodically are included in a set of periodic tasks. Application tasks perform application specific operations and may be either periodic or aperiodic according to their own specific operations. The network protocol is classified into IP, TCP, UDP, ARP, and ARP Update tasks. IP, TCP, UDP and ARP tasks are included in a set of aperiodic tasks because of not having the period of their execution. The TCP task is divided into two subtasks, the TCP-IN task for transmitting packets and the TCP-OUT task for receiving packets. Most characteristics of the ARP task are aperiodic except updating the ARP table every 20 minutes. The ARP Update task updates the ARP table as a periodic task.

Fig. 1. Architecture of real-time sensor networking platforms

The proposed technique consists of three primary components, a set of periodic and aperiodic tasks, and PATS (periodic and aperiodic tasks scheduler). The periodic task set component contains Sensing, Application and ARP Update tasks. The PATS component is composed of four sub-modules which are SC (Schedulability Checker), STMAT (Slack Time Manager), ATS (Aperiodic Task Scheduler), and PTS (Periodic

Task Scheduler) modules. The SC module decides if a given periodic task set can be schedulable. The STMAT evaluates the processor idle time called slack time. It consists of two sub-modules, STCAT (Slack Time Calculator) and STAAT (Slack Time Allocation). The STCAT computes the slack time and stores it in the STT (Slack Time Table) and the STAAT assigns the Slack time to the aperiodic task chosen by the ATS. The ATS schedules the aperiodic tasks, allocated by the STMAT, in the FIFO policy. The PTS schedules a set of periodic and aperiodic tasks according to the priority-driven policy. We make full use of the RM (Rate Monotonic) algorithm for periodic tasks and the Slack Stealing algorithm for aperiodic tasks [3-4].

Schedulability Checker

We use the schedulability test addressed in [3]. The basic terms are as follows. C_i is the execution time of the periodic task τ_i , and T_i is the period of τ_i . Then a workload of τ_i , $W_i(t)$, is defined as follow

$$W_i(t) = \sum_{j=1}^i C_j \cdot \lceil t/T_j \rceil, \quad t = \{k \cdot T_j \mid j = 1, \dots, i; k = 1, \dots, \lfloor T_i/T_j \rfloor\}. \quad (1)$$

Let $L_i(t)$ be the processor utilization of τ_i . By (1), $L_i(t) = W_i(t)/t$. Let L_i be the actual processor utilization of τ_i and L be the total processor utilization of task sets. Then L can be computed as follow

$$L = \max_{\{1 \leq i \leq n\}} L_i = \max_{\{1 \leq i \leq n\}} \min_{\{0 < t \leq T_i\}} L_i(t) = \max_{\{1 \leq i \leq n\}} \min_{\{0 < t \leq T_i\}} \sum_{j=1}^i C_j \cdot \lceil t/T_j \rceil / t \quad (2)$$

If $L \leq 1$, then the task set is decided to be schedulable.

Slack Time Manager

We use the method of [4] for computing and allocating the slack time. Let $\alpha_i(t)$ be the total execution time of the aperiodic tasks which have the priorities higher or equal to the priority i . Let $P_i(t)$ be the ready time of the periodic tasks in $[0, t]$ and $I_i(t)$ be the processor idle time or the execution time of the tasks of which priorities are lower than the priority i . Let A_{ij} be the maximum of the slack times in $[0, C_{ij}]$ where C_{ij} is the completion time of τ_{ij} . Then, the total processor time $W_i(t)$ is computed as follows

$$W_i(t) = \alpha_i(t) + P_i(t) + I_i(t) = \min_{\{0 \leq t \leq D_{ij}\}} \{(A_{ij} + P_i(t))/t\} = 1 \text{ for } 0 \leq t \leq D_{ij} \quad (3)$$

$$A^*(t) = \min_{\{1 \leq i \leq n\}} A_i(t), \quad C_{ij-1} \leq t < C_{ij} \text{ for } A_i(t) = A_{ij} \text{ and } C_{ij-1} \leq t < C_{ij}, \quad j \geq 1 \quad (4)$$

$A^*(t)$ represents the maximum of the slack times.

Aperiodic and Periodic Task Scheduler

The ATS can schedule the aperiodic task only when the value in the STT is only positive. The PTS schedules the periodic and aperiodic tasks by the priority-driven policy. The priority of the periodic task is assigned by the RM and the aperiodic task is given by the system designer. PTS would select and schedule the task of the highest priority of priority tables for periodic and aperiodic tasks.

3 Experiments and Conclusion

All of the experiments in this paper were carried out using the CC2420DB wireless sensor mote. The CC2420DB hardware contains an IEEE802.15.4 compliant RF transceiver CC2420, an Atmega128L microcontroller, one internal sensor (a temperature sensor) and two external sensors (light and humidity sensors) we installed. The software platform incorporating the proposed task decomposition technique is implemented in this paper by modifying and improving uC/OS-II v2.52 [5] and uIP v0.9 [6]. For measuring the performance of our method, three experimental models, Case 1, 2 and 3, are evaluated. The Case 1 model is the original uC/OS-II incorporating the original uIP. The Case 2 model is the original uC/OS-II incorporating the TCP/IP software communications architecture which is decomposed into task-per-protocol. We develop the task-based TCP/IP software communications architecture by exploiting the uIP v0.9. The Case 3 model is the modified uC/OS-II incorporating the proposed task decomposition technique containing the task-based TCP/IP software communications architecture. We evaluate the performance of Case 1, 2, and 3 respectively by executing each one of them on two CC2420DB motes communicating with each other. The unique IP address is assigned to each of two CC2420DB motes. A case study based on real system experiments is conducted to illustrate the application and efficiency of the proposed technique. It shows that our periodic and aperiodic task decomposition technique yields efficient performance in terms of guaranteeing the deadlines of all the periodic tasks and aiming to provide aperiodic tasks with good average response time. Our research gives valuable insight into how to deploy real-time sensor networking platforms that has an unique software architecture with task-based components.

Acknowledgement

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (The Regional Research Universities Program/Research Center for Logistics Information Technology).

References

1. Subramonian V., Huang-Ming H., Datar S., Chenyang L.: Priority scheduling in TinyOS - a case study. Washington University Technical Report (WUCSE-2003-74), December (2003)
2. Hofmeijer T.J., et. al.: DCOS, A real-time light-weight data centric operating system. The IASTED International Conference on Advances in Computer Science and Technology(ACST 2004), St. Thomas, US Virgin Islands, November (2004)
3. Lehoczky J., Sha L., Ding Y.: The rate monotonic scheduling algorithm: exact characterization and average case behavior. Real Time Systems Symposium, December (1989), 166-171
4. Lehoczky J.P., Ramos-Thuel S.: An optimal algorithm for scheduling soft-aperiodic tasks in fixed-priority preemptive systems. Proc. of 13th Real-Time Systems Symposium, December (1992), 110-123
5. Labrosse J.J.: uC/OS-II the real-time kernel. CMP Books, Kansas, (2002)
6. Dunkels A.: Full TCP/IP for 8Bit architectures. In Proc. of the First ACM/Usenix International Conference on Mobile Systems, Applications and Services (MobiSys 2003), San Francisco, USA, May (2003), 85-98