Journal of information and communication convergence engineering
http://jicce.org

J. Inf. Commun. Converg. Eng. 10(4): 396-404, Dec. 2012 **Regular Paper**

Improved Schedulability Analysis of Real-Time Sporadic Tasks with EDF Preemptive Scheduling

Armaghan Darbandi and Myung-Kyun Kim*, ***Member*, *KIICE***

School of Electrical Engineering, University of Ulsan, Ulsan 680-749, Korea

Abstract

This paper proposes an analysis method to check the schedulability of a set of sporadic tasks under earliest deadline first (EDF) scheduler. The sporadic task model consists of subtasks with precedence constraints, arbitrary arrival times and deadlines. In order to determine the schedulability, we present an approach to find exact worst case response time (WCRT) of subtatsks. With the technique presented in this paper, we exploit the precedence relations between subtasks in an accurate way while taking advantage of the deadline of different subtasks. Another nice feature of our approach is that it avoids calculation time overhead by exploiting the concept of deadline busy period. Experimental results show that this consideration leads to a significant improvement compared with existing work.

Index Terms: EDF scheduling, Preemptive scheduling, Schedulability, Sporadic tasks

I. INTRODUCTION

There are many industrial embedded systems consisting of millions of lines of code, and containing many tasks, where some tasks may have real-time constraints. Looking closer at these systems, independent tasks may consist of subtasks that exhibit strong dependencies. Subtask dependencies are necessary for realizing some control activities. That is, some subtasks have to respect a processing order due to message exchange between subtasks or usage of various resources. A significant problem in such systems is that missing deadlines can cause disastrous failures. One desirable approach to avoid timing-related errors in such complex systems is to use exact worst case response time (WCRT) analysis to provide a reliable guarantee.

Spuri [1] derived WCRT for sporadically periodic tasks (independent task instances that arrive within a certain period for a time interval, and then does not rearrive for a longer period) with arbitrary deadlines where tasks are scheduled using a preemptive earliest deadline first (EDF) scheduler. Another approach proposed by Zhang and Burns [2] obtains necessary and sufficient conditions for EDF schedulability of a set of sporadic tasks. Within a sporadic task subtasks inter-arrival time is greater than or equal to the task period. The proposed schedulability conditions reduce the calculation overhead of previous results. Zhang et al. [3] model WCRT under preemptive fixed priority, for another extension of sporadic tasks. The sporadic task model introduces a bound on the number of task arrivals allowed in a specific time window. Further, they allow a different number of task arrivals for different time windows. However, the approaches in [1-3] did not consider the precedence constraints among subtasks.

A different approach proposed in [4-7] derives WCRT analysis for a sort of sporadic task with precedence-

Received 17 September 2012, **Revised** 20 October 2012, **Accepted** 29 October 2012
***Corresponding Author** E-mail: mkkim@ulsan.ac.kr

Open Access **http://dx.doi.org/10.6109/jicce.2012.10.4.396** print ISSN: 2234-5973 online ISSN: 2234-8883

constrained subtasks where task activation results in activation of all subtasks at the same time. That is, all subtasks arrive simultaneously as an external event occurs. To consider the precedence constraints between subtasks, the authors model a sporadic task by a directed acyclic graph (DAG). These approaches adopt a technique for transforming the DAG of the task under analysis to a simple chain by modification of the deadline of the subtasks. This technique is still somehow an approximation of the response times, because in some particular cases, transformation of a graph to a chain, and consequently introducing an artificial deadline for subtasks, is not able to exploit sufficiently the priorities among subtasks.

This paper extends the aforementioned results to a larger class of sporadic task models. Our proposed sporadic task allows an arbitrary arrival time and arbitrary deadline of subtasks with acyclic precedence constraints, i.e., within a sporadic task, subtasks include acyclic precedence constraints and arrive at arbitrary time instances with arbitrary deadlines. Subtasks are implemented upon a uniprocessor platform and are scheduled under a preemptive EDF scheduler such that subtasks with precedence constraints are scheduled according to their precedence relations, while subtasks with no precedence relations are scheduled according to the EDF algorithm. In contrast to approaches in [4-7], to acquire WCRTs precisely, no deadline modification is allowed and the analysis is performed on the original sporadic graph.

Since by adopting offset-based techniques, the co-rresponding analysis of a distributed system or a multi-processor system can be transformed to an equivalent single processor analysis, our approach can be easily adopted for schedulability analysis of distributed systems or multiprocessors. In this case, task executions and message transmissions are modeled as preemptive/non-preemptive tasks executed in a single processor. In the study of schedulability of a distributed real-time system, compared with the approach proposed by Palencia and Harbour [8], our contribution is shown to have significant improved performance by a factor that grows larger with processor utilization.

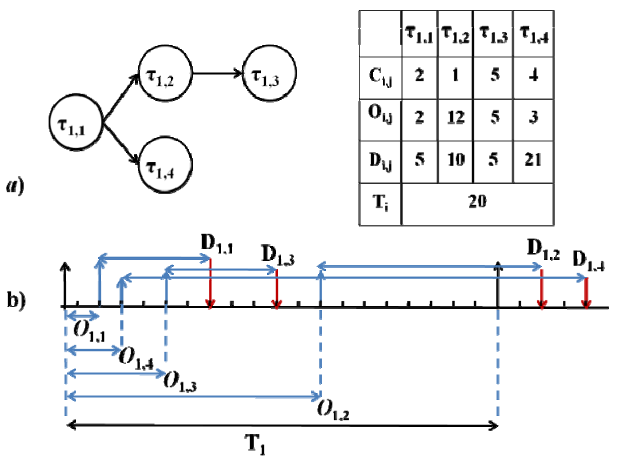
This paper starts by introducing related works in Section II, and the computational model in Section III. The proposed WCRT analysis is described in Section VI. A case study is presented in Section V. Finally, our conclusions are stated in Section VI.

II. RELATED WORKS

In the past, periodic and sporadic task scheduling has received considerable attention. A well-known result for periodic tasks is that the preemptive EDF algorithm is optimal in the sense that it will successfully generate a processor schedule for any periodic task system that can be scheduled [9]. In contrast to periodic tasks, a sporadic task is invoked at arbitrary times but with a specified minimum time interval between invocations. Kim and Naghibzadeh [10] showed the optimality of non-preemptive EDF, for the scheduling of a set of sporadic tasks with relative deadlines equal to their periods. The authors proposed the feasibility condition for a set of sporadic tasks with relative deadlines equal to their respective periods. [2] obtains the necessary and sufficient conditions for EDF schedulability of sporadic tasks, where subtasks' inter-arrival time is greater than or equal to the task period. [3] considered WCRT for a sort of sporadic task model where a bounded number of tasks arrive in a specific time window. Spuri [1] proposed WCRT analysis of sporadically periodic task sets with arbitrary timing requirements scheduled under EDF. In this approach, sporadically periodic tasks are identified by two periods: an inner period and an outer period. The outer period is the worst-case inter-arrival time between bursts and the inner period is the worst-case inter-arrival time between task instances within a burst [1]. However, the computational model in the above-mentioned works does not cover dependency among subtasks of a sporadic task. A common form of dependency arises when one subtask produces the communication data of another consumer subtask to proceed with its computations. Further, even when no explicit data exchanges are involved, subtasks might be required to run in certain orders.

The approaches proposed by Zhao et al. [4, 5], concerns WCRT analysis for a set of independent sporadic tasks, scheduled under non-preemptive EDF scheduling and with fixed priority scheduling, respectively. Each sporadic task is represented by a graph to indicate subtask dependencies. The main idea is to transform the graph to a canonical chain by modification of the deadline of subtasks appropriately so that each subtask in the obtained canonical chain has a deadline larger than its predecessors. The idea of a graph to chain transformation was inspired by the approaches proposed by Harbour et al. [6, 7]. This approach uses the graph to chain transformation to provide a theoretical framework for analyzing task sets scheduled through a fixed priority preemptive scheduler. However, [4-7] cover simultaneous arrival of subtasks only.

In the study of task level precedence constraints, the meaning of a precedence constraint is applicable in practice only to tasks that have the same rate. Mangeruca et al. [11] generalizes the concept of precedence constraints between tasks to allow tasks to have different rates. Moreover, they consider integer linear problem formalization of the problems of optimum priority/deadline assignment for preemptive EDF scheduling under precedence constraints.



	$\tau_{1,1}$	$\tau_{1,2}$	$\tau_{1,3}$	$\tau_{1,4}$
$C_{i,j}$	2	1	5	4
$O_{i,j}$	2	12	5	3
$D_{i,j}$	5	10	5	21
T_i	20			

Fig. 1. Computational model of a sporadic task. (a) Task τ_1 specifications, (b) timeline of activation time and deadline of subtasks.

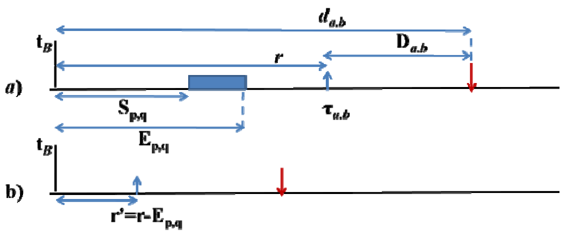


Fig. 2. An illustration of the equivalent deadline$-d$ busy period with a regular busy period. (a) Busy period which includes execution of $\tau_{p,q}$ with a latter absolute deadline than $d_{a,b}$. The rectangle illustrates the execution of $\tau_{p,q}$,(b) deadline$-d$ busy period starts with subtasks released at time interval ($t_B + S_{p,q}, t_B + E_{p,q}$).

As for distributed hard real-time systems, Spuri [12] adapted the Holistic analysis technique to the systems that were scheduled under the EDF.

In this work, the analysis requires global knowledge of the task routes to predict the worst-case end-to-end delay of a task. Later, [8, 13-15] improved the estimations of WCRTs by developing the offset-based technique. The fundamental principle behind it is that, given the offset information of the tasks at a communication resource, the authors transform a distributed system to an equivalent uniprocessor. In this case, estimations depend only on the load that the analyzed task encounters. Another extension was proposed by Redell [16], which allows each task to have one or more immediate successors. The main problem with these techniques is that they take into account the precedence relations between tasks only indirectly, causing pessimistic results to quickly increase along with the system scale. In Section V, we show the outperformmance of our work compared with the work in [8].

III. SYSTEM MODEL AND NOTATIONS

Definition. Sporadic Task (τ_i): Sporadic task τ_i consists of n_i number of preemptive subtasks $\tau_i = \{\tau_{i,1}, \dots, \tau_{i,ni}\}$ with precedence constraints and arbitrary timing requirements. The inter-arrival time of each task is separated by a minimum time T_i. Each task activation at time t results in activation of all subtasks, each one at time $t_{i,j} = t + O_{i,j}$, where $O_{i,j}$ denotes the arrival time relative to the activation of the task, Fig. 1. Each subtask has an execution time $C_{i,j}$, and relative deadline $D_{i,j}$. The absolute deadline $d_{i,j}$ of subtask $\tau_{i,j}$ requested at time $t_{i,j}$ equals $d_{i,j} = t_{i,j} + D_{i,j}$. The relative deadline of task τ_i is denoted by D_i, where D_i equals $D_i = \max_j(O_{i,j} + D_{i,j})$.

Definition. Busy Period: the time interval delimited by two successive idle times, during which the CPU is not idle. Let us denote the length of the longest busy period by L and the starting instance of the longest busy period by t_B.

Definition. Deadline-*d* Busy Period: A processor busy period in which only the subtasks with an absolute deadline smaller than or equal to d execute.

Definition. Worst-case Response Time (WCRT) ($R_{i,j}$): maximum possible delay between arrival time and completion time of subtask $\tau_{i,j}$ on each activation time of the subtask $\tau_{i,j}$ inside the longest busy period:

$$R_{i,j} = max\,\{R_{i,j}(r)\}; \forall r\; t_B \le r < t_B + L, \qquad (1)$$

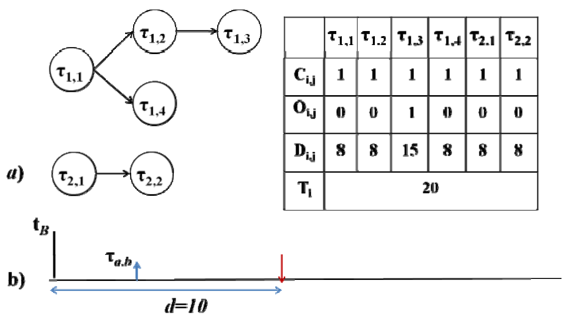
Procedure 1 determines the order of scheduling of subtasks:

```
Procedure 1
// τ_{k,l} denotes the subtask that is released inside the busy period
Set t=0
Do
Among subtasks with no predecessors select subtasks that
arrived in the interval $[t_B, t_B + t]$; Execute a subtask $\tau_{k,l}$ with
minimum deadline; $t = t + c_{k,l}$;
Remove the executed subtask, $\tau_{k,l}$ from the corresponding task
graph;
Until processor becomes idle
```

IV. RESPONSE TIME ANALYSIS

According to the generalization of an original result described by Liu and Layland [9] we simply need to study the schedule of tasks in the most demanding arrival pattern, the longest busy period. As noted in Section III, the length of the longest busy period in our model is denoted by L and the starting instance of the longest busy period by t_B. When the schedulability of periodic tasks under EDF is considered, from [17] we have: the completion time of a task's instance with absolute deadline d, must be the end of a busy period in which all executed instances have absolute deadlines less than or equal to d (which is defined as deadline$-d$ busy period). If we are able to examine all such periods, by taking the maximum length we can find the WCRT of a task [17]. Let us consider the validity of the above argument for our proposed task model with precedence constrained subtasks. To do so, using a simple example illustrated in Fig. 2a, we shall demonstrate that it suffices to consider the worst-case scenario of the subtask $\tau_{a,b}$ with absolute deadline d that was released at r unit times after t_B, inside a deadline$-d$ busy period. Consider a busy period where the coincidence of some subtasks $\tau_{p,q}$, with $D_{p,q} > d$ occurs at t_B. Assume that the subtask $\tau_{p,q}$ with $D_{p,q} > d$ is executed before starting instance of execution of the analyzed subtask $\tau_{a,b}$, as shown in Fig. 2a. Even though $\tau_{a,b}$ with an earlier absolute deadline is released earlier than $\tau_{p,q}$, due to the highest urgency of the predecessors of $\tau_{a,b}$ and their exposed precedence relations, it is possible that $\tau_{p,q}$ would be executed earlier. However, we show that it is possible to avoid the conservative analysis of such busy periods by introducing an equivalent deadline$-d$ busy period. Let us denote the phase between the starting instance of execution of the subtask $\tau_{p,q}$ with t_B, by $S_{p,q}$ and the phase between the period can be examined that starts at arrival of those subtasks.

In this case, the subtask $\tau_{a,b}$ would be released at time $r' = r - E_{p,q}$, see Fig. 2b because in both of the cases, the response time of subtask $\tau_{a,b}$ is identical. This means that, if execution of subtask $\tau_{p,q}$ with $D_{p,q} > d$ occurs before starting the instance of execution of analyzed subtask $\tau_{a,b}$, as shown in Fig. 2a, the considered deadline$-d$ busy period would be finished at time $t_B + S_{p,q}$.



	$\tau_{1,1}$	$\tau_{1,2}$	$\tau_{1,3}$	$\tau_{1,4}$	$\tau_{2,1}$	$\tau_{2,2}$
$C_{i,j}$	1	1	1	1	1	1
$O_{i,j}$	0	0	1	0	0	0
$D_{i,j}$	8	8	15	8	8	8
T_i	20					

Fig. 3. An example of a critical instance given a deadline$-d$ busy period with maximum length of 10. (a) Directed acyclic graph of tasks τ_1 and τ_1 and characteristics of each corresponding subtask, (b) the absolute deadline of subtask $\tau_{a,b}$ is d =10, t_B is the beginning of the busy period.

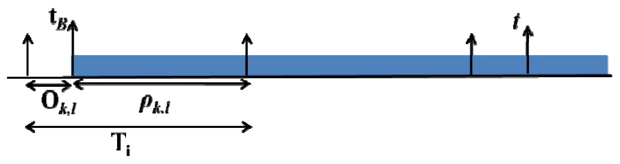


Fig. 4. A scenario to obtain complete activations of task τ_k up to time t.

Thus we are interested in deadline$-d$ busy periods with a length greater than or equal to r.

In a scenario where the model is composed of periodic tasks (as proposed in [1]) or the task model is defined as sporadic tasks with simultaneous arrival time of subtasks (as proposed in [5]) scheduled under EDF, the worst-case busy period occurs in the synchronous busy period, where all tasks arrive simultaneously. However, since in our model each task consists of subtasks with arbitrary release times and timing requirements, we must take into account that the critical instance may not be obtained by the occurrence of all tasks at the starting instance of the busy period. Let us consider the coincidence of which subtasks of t_B would result in the critical instance. Consider subtasks of the task τ_k that possibly delay execution of $\tau_{a,b}$. Each such subtask falls in the category Set_k^H, which is composed of subtasks with $D_{k,l} \le d$. According to the definition of the deadline $-d$ busy period, only the subtasks of task τ_k, $k \ne a$ that are included in Set_k^H are considered to delay the execution of the subtask $\tau_{a,b}$ by arriving inside a deadline$-d$ busy period. Therefore, a deadline$-d$ busy period could be constructed by releasing one subtask of each task with $D_{k,l} \le d$ at t_B. However, we do not know which subtask coincident to t_B corresponds to the critical instant for the analysis of the subtask $\tau_{a,b}$. Therefore, in the procedure of considering the critical instance of subtask $\tau_{a,b}$, we must study all possible deadline$-d$ busy periods created by choosing one higher priority subtask in each task to occur at time $t = t_B$. Fig. 3 illustrates an example showing that to construct the critical instance of $\tau_{a,b}$ in a deadline$-d$ busy period, we must consider all cases, where a higher priority subtask of each task occurs at t_B.

Consider a system consisting of three sporadic tasks, namely τ_1, τ_2, and τ_a. The task τ_a consists of one subtask $\tau_{a,b}$ with $O_{a,b} = 4$ and $D_{a,b} = 6$ where it is released at 4 unit times after t_B, as shown in Fig. 3. The sporadic graph of tasks τ_1, τ_2 and the characteristics of corresponding subtasks are shown in Fig. 3a. The timeline of the activation time and deadline of subtask $\tau_{a,b}$ is shown in Fig. 3b. For simplicity, let us assume that the computation time for each subtask of every task is one unit. In this case, it is easy to show that the longest deadline$-d$ busy period can be obtained by releasing τ_1 and τ_2 at t_B. Hence, the response time of subtask

$\tau_{a,b}$ equals $C_{1,1}+C_{1,2}+C_{1,4}+C_{2,1}+C_{2,2}+C_{a,b}-4=$ *2*. Let us assume another scenario defined in the same way, except that the computation time of subtask $\tau_{1,3}$ equals 7. In this case, releasing the subtask $\tau_{1,3}$ at t_B and releasing the task τ_2 at time t_B would lead to the longest deadline$-d$ busy period. Hence, the response time of subtask $\tau_{a,b}$ equals $C_{1,3}+C_{2,1}+C_{2,2}+C_{a,b}-4=$ *6*.

Theorem 1 helps us to find the worst-case scenario where $\tau_{a,b}$ is released at time $t_B \le r < t_B + L$:

Theorem 1: The worst-case response time of subtask $\tau_{a,b}$ with a release time at $t_B \le r < t_B + L$ and absolute deadline d, is found in a deadline$-d$ busy period of subtask $\tau_{a,b}$ where the first activation of some subtask $\tau_{k,l}$, of task τ_k, k $\neq a$, with $D_{k,l} \le d$ coincides with $t = t_B$ and task activations of τ_k occur periodically with their maximum rate inside the busy period.

Proof: Let $t_1 > t_B$ be the instant at which a subtask $\tau_{k,l}$, $k \neq a$ with absolute deadline $d_{k,l}$ is activated the first time inside the busy period, and let $d_{k,l} \le d$. Suppose that we move t_1 to coincide with $t = t_B$ in this circumstance; it is possible that an activation of successor subtasks of subtask $\tau_{k,l}$, denoted by $\tau_{k,j}$, with an absolute deadline after instant d may have been moved earlier to have a deadline before or at d; thus it would possibly increase the response time of the analyzed subtask.

Based on Theorem 1, in the procedure of finding the critical instance of subtask $\tau_{a,b}$ released at time $t_B \le r < t_B + L$, we must study all possible deadline$-d$ busy periods created by choosing one higher priority subtask in each task to occur at time $t = t_B$. Given all possible deadline$-d$ busy periods, the largest response time of subtask $\tau_{a,b}$ accounts for its WCRT, denoted by $R_{i,j}$.

In the rest of this paper, the coincident subtask of task τ_k to t_B is denoted by $\tau_{k,l}$. Through the analysis expressed in Eqs. (2)–(10) we shall assume that task τ_a consists of one subtask, $\tau_{a,b}$, only. In the latter part of this section, we will extend the analysis to consider the response time of $\tau_{a,b}$ where the task τ_a includes a sporadic graph with precedence constraints.

Let us first consider a complete activation of task τ_k that occurs in the time interval $[t_B, t]$, where $t > t_B$.

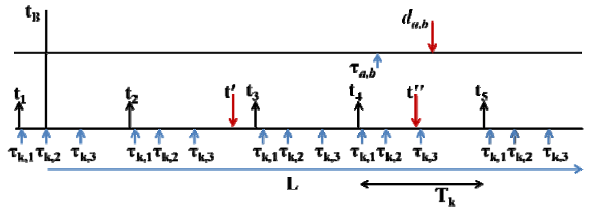


Fig. 5. An example of calculating the required time to schedule task τ_k up to t.

This activation occurs before t, so that $t-$'activation time' $\ge T_k$ and 'activation time' $- t_B \ge 0$. We denote the total number of complete activations of task τ_k that occur in the time interval $[t_B, t]$, by $N_{k,l}(t)$. As has been noted earlier, in our analysis, we assume that the subtask $\tau_{k,l}$ has been released at t_B. From Fig. 4, it is easy to see that $N_{k,l}(t)$ could be obtained by $N_{k,l}(t) = \left\lfloor \frac{t-\rho_{k,l}}{T_k} \right\rfloor$, where $\rho_{k,l}$ represents the phase between t_B and the first activation of task τ_k inside the busy period. Hence, we have $\rho_{k,l} = T - O_{k,l}$ if $O_{k,l} > 0$, and $\rho_{k,l} = 0$ otherwise. The total time required to schedule activations of task τ_k that occurred completely in the time interval $[t_B, t]$ equals:

$$\mathrm{C}_{k,l}(t) = \left\lfloor \frac{t-\rho_{k,l}}{T_k} \right\rfloor \cdot \sum_{i=1}^{n_k} c_{k,i}, \tag{2}$$

Fig. 5 represents a scenario for the alignment of the task τ_k arrival pattern after t_B. The upper part of this figure corresponds to the activation and deadline of subtask $\tau_{a,b}$, and the lower part represents the activation of subtasks of task τ_k. In Fig. 4, $N_{k,2}(t') = 0$, because $t' - t_2 < T_k$. Further, since $t'' - t_2 > T_k$, $t_2 - t_B > 0$ and also $t'' - t_3 > T_k$, $t_3 - t_B > 0$, we have $N_{k,2}(t'') = 2$.

It should be noted that in the above discussion, we only find the time required to execute $N_{k,l}(t)$ activations of task τ_k. However, among those, activations of task τ_k released after $d - D_k$ would not contribute to the response time of analyzed subtask [17]. Thus, at each given time interval $[t_B, t]$, only activations of task τ_k with deadlines before or at d, can contribute to the response time of analyzed subtask. At each given time interval $[t_B, t]$, $\mathrm{C}_{k,l}(t)$ is bounded by $G_{k,l}(d)$, which is given by following equation:

$$G_{k,l}(d) = \left(1 + \left\lfloor \frac{d-(D_k+\varphi_{k,l})}{T_k} \right\rfloor\right) \cdot \sum_{i=1}^{n_k} c_{k,i}, \tag{3}$$

Therefore, the contribution of $N_{k,l}(t)$ complete activations of task τ_k to the response time of the analyzed subtask, at each given time interval $[t_B, t]$, is given by:

$$W_{k,l}(t,d) = \min\left(\mathrm{C}_{k,l}(t), G_{k,l}(d)\right), \tag{4}$$

To see this, redoing the example in Fig. 5, let us assume that $D_{k,1} = D_{k,3} = T_k$, $D_{k,2} = 3 \cdot T_k - O_{k,2}$ and hence $D_k = 3 \cdot T_k$. In this case, we have $\mathrm{C}_{k,2}(t'') = 2.\sum_{i=1}^{3} c_{k,i}$, $G_{k,2}(d) = 0$ and consequently, $W_{k,l}(t'', d) = 0$.

Eq. (4) only obtains the contribution of complete activations of task τ_k that occur inside the time interval $[t_B, t]$. The problem that remains is to obtain the contribution of activations that do not interfere with the response time of the subtask under analysis, completely.

We shall categorize those contributions of task τ_k into two sets. Set A: activations included in this set occur completely inside the time interval $[t_B, t]$, but they have a deadline greater than d. Thus those activations do not interfere with the response time of $\tau_{a,b}$, completely. For example, in Fig. 5, consider $D_{k,1} = D_{k,3} = T_k$, $D_{k,2} = 3 \cdot T_k - O_{k,2}$. In this case, we have $W_{k,2}(t'', d) = 0$, but activations of task τ_k that occurred at times t_2 and t_3 consist of some subtasks that would delay the response time of $\tau_{a,b}$: subtasks $\tau_{k,1}^2$, $\tau_{k,1}^3$. We shall define $\delta_{k,l}^A(t, d)$ to obtain the total computation time of the subtasks of activations of task τ_k in Set A by

$$\delta_{k,l}^A(t, d) = \max(E_k), \qquad (5)$$

where $\max(E_k)$ represents the summation of computation time of the subtasks of activations in Set A that have an absolute deadline less than or equal to d and activation time before or at t, so that all their predecessors also have an absolute deadline less than or equal to d and activation time before or at t. In Fig. 5, consider $D_{k,1} = D_{k,3} = T_k$, $D_{k,2} = 3 \cdot T_k - O_{k,2}$, we have $\delta_{k,l}^A(t'', d) = 2 \cdot c_{k,1}$. Set B: activations included in this set do not occur completely inside the time interval $[t_B, t]$. This set may consist of two members only. The first one is the first activation that occurs immediately before t_B, where $O_{k,l} \neq 0$, for example, activation of task τ_k occurred at time t_1, in Fig. 5. The second member of Set B is the last activation, where $t - {}'activation\ time' < T_k$. Consider time interval $[t_B, t']$ in Fig. 5: activation of task τ_k occurs at time t_2 and corresponds to this element. We shall define $\delta_{k,l}^B(t, d)$ to account for the total computation time of the subtasks of activations of task τ_k in Set B by:

$$\delta_{k,l}^B(t, d) = \max(E_k), \qquad (6)$$

where $\max(E_k)$ is defined as for Eq. (5). Finally, we shall define $\delta_{k,l}(t, d)$ to obtain the total computation time of the subtasks of activations of task τ_k in Set A and Set B by

$$\delta_{k,l}(t, d) = \delta_{k,l}^A(t, d) + \delta_{k,l}^B(t, d). \qquad (7)$$

So far, we have studied the contribution of task τ_k to the response time of $\tau_{a,b}$ that has been released until time t. In general, the study of response time of a subtask $\tau_{a,b}$ with deadline d is an iterative procedure. The basic idea is that in each step, the obtained busy period must be the end of the execution of all instances of all tasks with deadlines less than or equal to d that have been released in the previous steps. Toward this, we shall define $LN_{a,b}(t, d)^{(k)}$ to represent the length of the resulting busy period in the k'th iteration of response time analysis of subtask $\tau_{a,b}$ with deadline d, where t is substituted with the length of the obtained busy period in the previous step. $LN_{a,b}(t, d)^{(k)}$ is ob-tained by the iterative equation, Eq. (9). This equation represents the iterative procedure, where at each step the length of a busy period is given by the summation of execution times obtained in Eqs. (4) and (7). We shall initiate this iterative procedure by Eq. (8). In Eq. (8), $c_{k,l}$ represents the execution time of the subtask coincident to t_B. The iteration is halted where the computations converge.

$$LN_{a,b}(t, d)^{(0)} = \textstyle\sum_{\forall k} c_{k,l}, \qquad (8)$$

$$LN_{a,b}(t, d)^{(m+1)} = \textstyle\sum_k W_{k,l}(LN_{a,b}(t, d)^{(m)}, d) + \sum_k \delta_{k,l}(LN_{a,b}(t, d)^{(m)}, d) \qquad (9)$$

The response time of subtask $\tau_{a,b}$ where it occurs at time $t_B \leq r < t_B + L$ is

$$R_{a,b}(r) = LN_{a,b}(t, d) + c_{a,b} - r, \qquad (10)$$

The problem that remains to be solved is to determine the response time of subtask $\tau_{a,b}$, where task τ_a includes a sporadic graph with precedence constraints. Consider a scenario where $\tau_{a,b}$ is released at r. In this case, the response time of $\tau_{a,b}$ is influenced by the order of execution of all subtasks of task τ_a that must be scheduled before $\tau_{a,b}$. Further, this sequence of subtasks of task τ_a may not be the same as the sequence of subtasks in another scenario where $\tau_{a,b}$ is released at r'. Therefore, it is of great importance to determine the order of subtasks of task τ_a that must be scheduled before $\tau_{a,b}$ for each release time of $\tau_{a,b}$.

In order to find the correct response time of subtask $\tau_{a,b}$, we shall use Procedure 1 to determine the correct sequence of subtasks of task τ_a that must be scheduled before $\tau_{a,b}$. This procedure would let us to consider the correct sequence of subtasks of task τ_a that must be scheduled before $\tau_{a,b}$ and consequently the correct response time of subtask $\tau_{a,b}$. To show this, let us denote the current subtask of task τ_a of which its response time has been considered by $\tau_{a,q}$ and the next chosen subtask by $\tau_{a,p}$. Consider a scenario where $\tau_{a,b}$ is released at r. In this case, $\tau_{a,q}$ and all the subtasks of task τ_a that have been considered by Eqs. (8) or (9) will be dropped from the sporadic graph of task τ_a, by Procedure 1. When $\tau_{a,p}$ is chosen as the analyzed subtask by Procedure 1, it must have the minimum deadline among all subtasks of task τ_a that arrived at or before the completion of execution of $\tau_{a,q}$, but they have no predecessors. This fact is in accordance with the definition of scheduling mentioned in Section III.

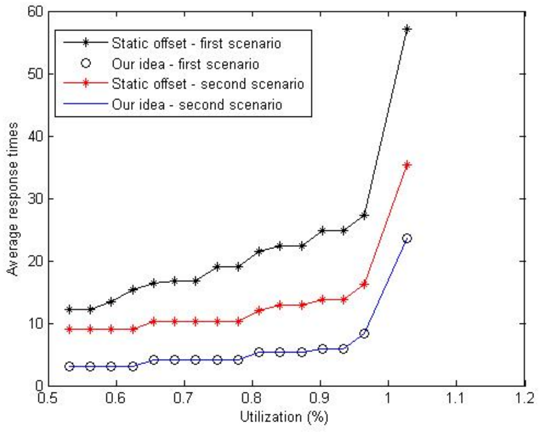


Fig. 6. Response time comparison of subtasks with offset = 0.

V. COMPARISON WITH EXISTING TECHNIQUES

We have compared the results of our proposed analysis, with the results obtained by Palencia and Harbour [8]. This approach transforms a distributed system schedulability analysis to an equivalent analyzable uniprocessor schedulability test, by using task offsets. Since tasks are preemptive, without loss of generality, we applied the static offset technique on a set of chains of subtasks, where each node in a chain represents a preemptive subtask.

Fig. 6 shows the obtained average response time by our analysis and obtained average response time by the static offset technique proposed in [8]. We ran our simulation on a task set consisting of 20 tasks, each one including five subtasks. The average response time corresponded to five subtasks of an analyzed task. The x-axis represents the processor utilization which varies with the changing execution time of the subtasks in the task set. The deadline of subtasks in the task set is generated randomly, except the 2nd subtask of each task in the first scenario and 2nd and 4th subtask in the second scenario. In the first scenario, the deadline of the 2nd subtask of each of the 20 tasks is generated so that it is greater than the deadline of the analyzed task. In the second scenario, the deadline of the 2nd and 4th subtask of each of the 20 tasks is generated so that it is greater than the deadline of the analyzed task. For the first scenario, it can be seen that the average response time obtained by the static offset technique is between 3 to 5 times larger than that of our results. The reason is that the static offset technique accounts for the execution time of the 3rd, 4th, and 5th subtasks of the interfering tasks in the task set where they do not contribute to the response time of the analyzed subtasks. In fact, it accounts for the execution time of each interfering subtask assuming that the corresponding predecessors have a deadline less than the analyzed subtask. This is a result that is shown to be flawed by [7] in the case of fixed priority scheduling. As for the second scenario, it can be seen that the difference between the average response times is tighter. However, our simulation shows the same results as the first scenario. The obtained smaller average response times by the static offset method is due to accounting for the execution time of the 3rd and 5th subtasks of the interfering tasks in the task set. However, in our constructed scenario, since the task offsets are set to zero, by the definition of the deadline busy period, they do not contribute to the response time of the analyzed subtasks.

In order to evaluate the pessimism of the admission controllers of our analysis and the static offset-based method implemented on a uniprocessor, we conducted simulations to identify the lowest utilization at which deadlines are missed. We perform our simulation on a set of 30 tasks with 5 subtasks per task, in one processor, for the case in which the task offsets are zero. Task sets were generated randomly, and the subtask specifications do not represent worst-case scenarios. We add a new task, denoted by τ_a, with 5 subtasks with randomly generated deadlines, accordingly. We increased the utilization of the task set by increasing the execution time of subtasks uniformly, until deadline misses were observed. Fig. 7 presents the results of our analysis where the y-axis represents the utilization and x-axis represents 5 states. Under state 1 (2, 3, 4, 5) each task has the first (and second, third, fourth, and fifth, respectively) subtask in its corresponding chain with a deadline greater than D_a. Each bar in Fig. 7 presents the results from the simulations of the lowest utilization at which the deadline misses were observed for different task states in the system.

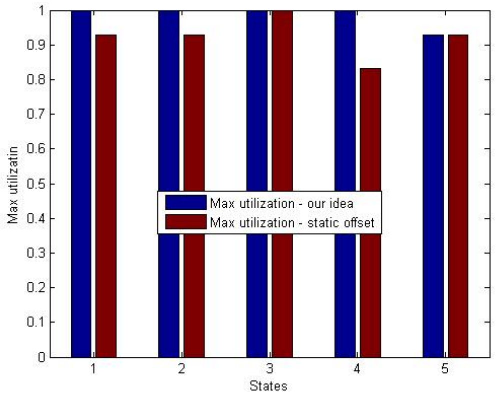


Fig. 7. Our analysis test vs. static offset-based technique with different states of deadlines of subtasks within tasks.

It can be observed that for our analyzed task model, the new task τ_a can be admitted for all utilizations in the first 4 states. In the 5th state, τ_a can be admitted for utilizations below 90%. In addition, for the first 4 states, the maximum utilization of the task set at which τ_a can be admitted, by the static offset based method, fluctuates between 80% in the 4th state and 100% in the 3rd state. As previously mentioned, the drawback of the static offset-based analysis is that it accounts for the execution time of each interfering subtask assuming that the corresponding predecessors have a deadline less than D_a. The 2nd, 3rd, and 4th state can be similarly described, indeed. However, in the 5th state, both methods have the same results.

In another experiment, we consider a scenario where the task sets represent the worst-case scenario. In this case, all the subtasks with a deadline less than or equal to D_a have a deadline less than or equal to the minimum deadline of the subtasks of task τ_a. Fig. 8 presents the results of our analysis where the Y axis represents the utilization and x-axis represents 10 states. The first 5 states are tested as explained for Fig. 7. State 6 represents a scenario where the 1st and 3rd subtask have a deadline strictly greater than D_a. State 7 represents a scenario where the 1st and 4th subtask have a deadline strictly greater than D_a. State 8 represents a scenario where the 2nd and 4th subtask have a deadline strictly greater than D_a in state 9 and 10, subtasks 1st, 4th and 5th and 2nd, 4th and 5th, respectively, have a deadline strictly greater than D_a. In this case, due to the drawback pointed out for the last experiments, it can be seen that the maximum task set utilization obtained by the offset based technique degrades significantly.

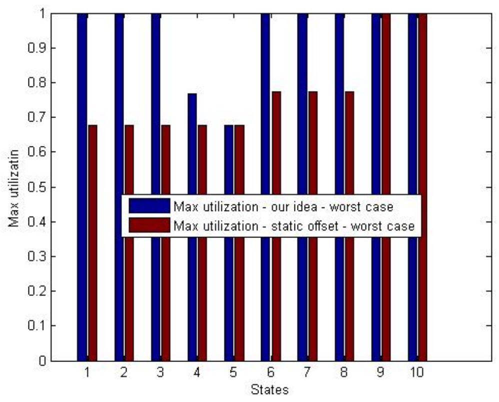


Fig. 8. Our analysis test vs. static offset-based technique with different states of deadline of subtasks within tasks in the worst case.

VI. CONCLUSIONS

In this paper, we studied the exact WCRT analysis of sporadic tasks that are modeled by DAG and scheduled under preemptive EDF scheduling. The objective is to obtain precise WCRTs of a generalized sporadic task model with arbitrary timing requirements that have not been considered in previous works. A nice feature of our work is that it exploits the precedence constraints between subtasks in an accurate way while taking advantage of the deadlines of subtasks. We find via simulation results that our methodology provides accurate results comparable with existing works.

ACKNOWLEDGMENTS

This work was supported by University of Ulsan, School of Excellence in Electrical Engineering.

REFERENCES

[1] M. Spuri, "Analysis of deadline scheduled real-time systems," Institut National de Recherche en Informatique et en Automatique (INRIA), Le Chesnay, France, Report no. 2772, 2006.

[2] F. Zhang and A. Burns, "Schedulability analysis for real-time systems with EDF scheduling," *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1250-1258, 2009.

[3] Y. Zhang, D. K. Krecker, C. Gill, C. Lu, and G. H. Thaker, "Practical schedulability analysis for generalized sporadic tasks in distributed real-time systems," in *Proceedings of the Euromicro Conference on Real-Time Systems*, Prague, Czech, pp. 223-232, 2008.

[4] H. X. Zhao, S. Midonnet, and L. George, "Worst case response time analysis of sporadic graph tasks with fixed priority scheduling on a uniprocessor," in *Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, Hong Kong, China, pp. 23-29, 2005.

[5] H. X. Zhao, L. George, and S. Midonnet, "Worst case response time analysis of sporadic graph tasks with EDF scheduling on a uniprocessor," in *Proceedings of the 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications,* Sydney, Australia, pp. 271-278, 2006.

[6] M. G. Harbour, M. H. Klein, and J. P. Lehoczky, "Timing analysis for fixed-priority scheduling of hard real-time systems," *IEEE Transactions on Software Engineering*, vol. 20, no. 1, pp. 13-28, 1994.

[7] M. G. Harbour, M. H. Klein, and J. P. Lehoczky, "Fixed priority scheduling periodic tasks with varying execution priority," in *Proceedings of the 12th Real-Time Systems Symposium*, San Antonio: TX, pp. 116-128, 1991.

[8] J. C. Palencia and M. G. Harbour, "Offset-based response time analysis of distributed systems scheduled under EDF," in

Proceedings of the 15th Euromicro Conference on Real-Time Systems, Porto, Portugal, pp. 3-12, 2002.

[9] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46-61, 1973.

[10] K. H. Kim and M. Naghibzadeh, "Prevention of task overruns in real-time non-preemptive multiprogramming systems," in *Proceedings of the International Symposium on Computer Performance Modelling, Measurement and Evaluation*, Toronto, Canada, pp. 267-276, 1980.

[11] L. Mangeruca, M. Baleani, A. Ferrari, and A. Sangiovanni-Vincentelli, "Uniprocessor scheduling under precedence constraints for embedded systems design," *ACM Transactions on Embedded Computing Systems*, vol. 7, no. 1, article no. 6, 2007.

[12] M. Spuri, "Holistic analysis for deadline scheduled real-time distributed systems," Institut National de Recherche en Informatique et en Automatique (INRIA), Le Chesnay, France, Report no. 2873, 1996.

[13] R. Pellizzoni and G. Lipari, "Improved schedulability analysis of real-time transactions with earliest deadline scheduling," in *Proceedings of the 11th IEEE Real Time on Embedded Technology and Applications Symposium*, San Francisco: CA, pp. 66-75, 2005.

[14] J. C. Palencia and M. G. Harbour, "Exploiting precedence relations in the schedulability analysis of distributed real-time systems," in *Proceedings of the 20th IEEE Real-Time Systems Symposium*, Phoenix: AZ, pp. 328-339, 1999.

[15] J. C. Palencia and M. G. Harbour, "Schedulability analysis for tasks with static and dynamic offsets," in *Proceedings of the 19th IEEE Real-Time Systems Symposium*, Madrid, Spain, pp. 26-37, 1998.

[16] O. Redell, "Analysis of tree-shaped transactions in distributed real time systems," in *Proceedings of the 16th Euromicro Conference on Real-Time Systems*, Catania, Italy, pp. 239-248, 2004.

[17] K. Tindell, "Adding time-offsets to schedulability analysis," Department of Computer Science, University of York, England, 1994.

Armaghan Darbandi
received the B.Sc. degrees in computer engineering from Polytechnic University of Tehran, in 2003. She is currently working toward the integrated M.S. and Ph.D. degree at the University of Ulsan, South Korea. Her research interests include real-time communications in wireless networks and real-time systems.

Myung-Kyun Kim
is currently a professor in the Department of Computer Science and Electrical Engineering, University of Ulsan, Korea. His research area is real-time communications in wireless networks and industrial communication networks. He received the B.Sc. degree from Seoul National University of Korea and the M.Sc. and Ph.D. degrees from KAIST, Korea.