Compositional Real-Time Scheduling Framework with Periodic Model

INSIK SHIN and INSUP LEE
University of Pennsylvania

It is desirable to develop large complex systems using components based on systematic abstraction and composition. Our goal is to develop a compositional real-time scheduling framework to support abstraction and composition techniques for real-time aspects of components. In this paper, we present a formal description of compositional real-time scheduling problems, which are the *component abstraction* and *composition* problems. We identify issues that need be addressed by solutions and provide our framework for the solutions, which is based on the *periodic interface*. Specifically, we introduce the periodic resource model to characterize resource allocations provided to a single component. We present exact schedulability conditions for the standard Liu and Layland periodic task model and the proposed periodic resource model under EDF and RM scheduling, and we show that the component abstraction and composition problems can be addressed with periodic interfaces through the exact schedulability conditions. We also provide the utilization bounds of a periodic task set over the periodic resource model and the abstraction bounds of periodic interfaces for a periodic task set under EDF and RM scheduling. We finally present the analytical bounds of overheads that our solution incurs in terms of resource utilization increase and evaluate the overheads through simulations.

Categories and Subject Descriptors: D.4.1 [**Operating Systems**]: Process Management—*Scheduling*; D.4.7 [**Operating Systems**]: Organization and Design—*Real-time systems and embedded systems*

General Terms: Algorithm, Design, Performance, Theory

Additional Key Words and Phrases: Hierarchical, real-time, scheduling, interface, component, abstract, composition

ACM Reference Format:

Shin, I., and Lee, I. 2008. Compositional Real-Time scheduling framework with periodic model. ACM Trans. Embedd. Comput. Syst. 7, 3, Article 30 (April 2008), 39 pages. DOI = 10.1145/1347375.1347383 http://doi.acm.org/10.1145/1347375.1347383

Authors' addresses: Insik Shin and Insup Lee, University of Pennsylvania, 3330 Walnut St., Philadelphia, Pennsylvania 19104; email: {ishin,lee}@cis.upenn.edu.

1. INTRODUCTION

As embedded systems become more complex because of increased functionalities, it is necessary to develop techniques and methods that facilitate the designing of large complex systems from subsystems. Component-based design has been widely accepted as a methodology for designing large complex systems through systematic abstraction and composition. Component-based design provides a means for decomposing a system into components, allowing the reduction of a single complex design problem into multiple simpler design problems, and composing components into a system through component interfaces that abstract and hide their internal complexity. Component-based design also facilitates the reuse of components that may have been developed in different environments. A central idea in component-based design is to assemble components into a system without violating the principle of *compositionality* such that properties that have been established at the component level also hold at the system level.

For embedded systems that are also real-time systems, it is important that the timing properties of components be analyzed compositionally as components are assembled hierarchically. Otherwise, real-time embedded systems would not benefit much from component-based design. However, traditional real-time scheduling frameworks do not support abstraction and composition techniques for timing properties of component that preserve compositionality, except in trivial cases. In this paper, we define what it means for a real-time scheduling framework to support compositionality. We then identify issues to develop such a real-time scheduling framework and present our approach to the problems.

Our primary goal is to develop a compositional real-time scheduling framework where global (system level) timing properties are established by composing together independently (specified and) analyzed local (component-level) timing properties. To develop such a framework, we address the component-abstraction and- composition problems. The *component-abstraction* problem is to combine and abstract the collective real-time requirements of a component as a single real-time requirement, called *real-time interface*. The *component-composition* problem is to compose independently analyzed local (component level) timing properties under their local schedulers into a global (system-level) timing property under a global scheduler. For the real-time interface model, this paper assumes the standard Liu and Layland periodic model [Liu and Layland 1973]. For scheduling algorithms, this paper assumes the EDF and RM algorithms, which are optimal dynamic and static uniprocessor scheduling algorithms for the standard periodic tasks. With these assumptions, the component abstraction problem then becomes how to abstract a set of periodic tasks under EDF or RM scheduling into a periodic interface. After a component has been abstracted as a periodic interface, the component can be treated as a single periodic task at the system level. Thus, the component composition problem is then how to combine a set of periodic interfaces, which will be treated as a set of periodic tasks, under EDF or RM scheduling into a single periodic interface. We note that the same periodic model is used for both the component abstraction and composition problems and, thus, the same technique can be used to find the solutions of the both problems.

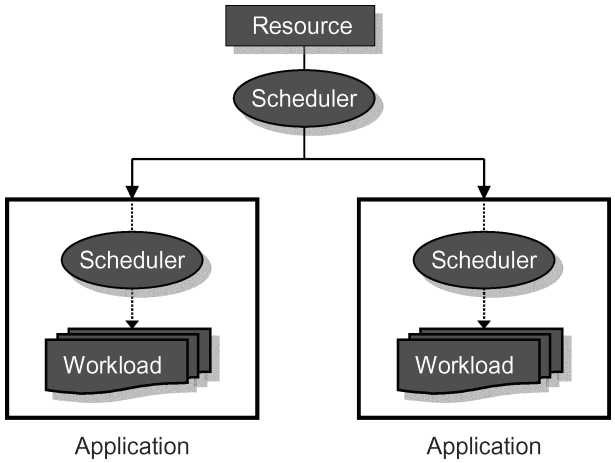

Fig. 1. Hierarchical scheduling framework: a resource is scheduled by a scheduler and each share of the resource is subsequently scheduled by a different scheduler.

1.1 Related Work

A hierarchical scheduling framework has been introduced to support hierarchical resource sharing among applications under different scheduling services. The hierarchical scheduling framework can be generally represented as a tree of nodes, where each node represents an application with its own scheduler for scheduling internal workloads (e.g., threads), and resources are allocated from a parent node to its children nodes, as illustrated in Figure 1. Goyal et al. [1996] first proposed a hierarchical scheduling framework for supporting different scheduling algorithms for different application classes in a multimedia system.

The hierarchical scheduling framework can be used to support multiple applications while guaranteeing independent execution of those applications. This can be correctly achieved when the system provides *partitioning*, where the applications may be separated functionally for fault containment and for compositional verification, validation, and certification. Such a partitioning prevents one partitioned function from causing a failure of another partitioned function.

The hierarchical scheduling framework is particularly useful in the domain of open systems, where applications may be developed and validated independently in different environments. For example, the hierarchical scheduling framework allows an application to be developed with its own scheduling algorithm internal to the application and then later included in a system that has a different meta-level scheduler for scheduling applications.

For real-time systems, there has been a growing attention to hierarchical scheduling frameworks [Deng and Liu 1997; Kuo and Li 1999; Lipari and Baruah 2000; Feng and Mok 2002; Regehr and Stankovic 2001; Saewong et al. 2002; Lipari and Bini 2003; Shin and Lee 2003; Shin and Lee 2004; Almeida and Pedreiras 2004; Matic and Henzinger 2005].

Deng and Liu [1997] proposed a two-level real-time scheduling framework for open systems. Kuo and Li [1999] presented an exact schedulability condition for such a two-level framework with the RM system scheduler, under the

assumption that all periodic tasks across components are harmonic. Lipari and Baruah [2000] and Lipari et al. [2000] presented exact schedulability conditions for the two-level framework with the EDF system scheduler, while placing a server between the system scheduler and each component scheduler. The main role of the server is to compute the resource requirements of components at runtime so that the EDF system scheduler can use them in scheduling components. Each server is assumed to have knowledge of the task-level deadlines of its corresponding component. The common assumption shared by these previous approaches is that the system scheduler has a (schedulable) utilization bound of 100%. In open systems, however, it is desirable to be more general, since there could be more than two-levels and different schedulers may be used at different levels.

Mok et al. [2001] proposed the bounded-delay resource partition model for a hierarchical scheduling framework. Their model can specify the real-time guarantees that a parent component provides to its child components, where the parent and child components have different schedulers. Their approach allows the schedulability of a child node to be analyzed independent of its context in a sufficient manner and thus does not require the assumption that the parent component's scheduler has a utilization bound of 100%. In their hierarchical framework, a parent component and its child components are separated such that they interact with each other only through their resource partition model. Feng and Mok [2002] and Shin and Lee [2004] have considered the component abstraction and composition problems with the bounded-delay resource partition model.

There have been studies on the component-abstraction problem with the periodic resource model. This periodic resource model can specify the periodic resource allocation guarantees provided to a component from its parent component [Shin and Lee 2003]. Saewong et al. [2002] introduced an exact RM schedulability conditions based on the worst-case response time analysis and Lipari and Bini [2003] presented another exact RM schedulability condition based on time demand calculations.[1] Shin and Lee [2003] presented an exact EDF schedulability condition. Saewong et al. [2002] and Shin and Lee [2003] also presented utilization bounds. Almeida and Pedreiras [2004] considered the component-abstraction problem in the presence of mutually exclusive resources within a component.

Matic and Henzinger [2005] considered the issue of addressing the component-abstraction problem in the presence of interacting tasks within a component. They considered two approaches, the RTW (real-time workshop) [Mathworks 2005] and the LET (logical executioni time) [Henzinger et al. 2003] semantics, for supporting tasks with intra- and intercomponent data dependencies. They showed that our proposed real-time compositional framework can be extended together with either approach for supporting tasks with both intra- and intercomponent dependencies. They also showed that

[1]They presented their schedulability condition as a sufficient condition. However, their conditions are the sufficient and necessary conditions based on the notion of schedulability under the worst-case resource supply used in this paper.

both approaches produce a trade-off between the end-to-end latency and the component-abstraction overhead.

Regehr and Stankovic [2001] introduced another hierarchical scheduling framework that considers various kinds of real-time guarantees. Their work focused on converting one kind of guarantee to another kind of guarantee such that whenever the former is satisfied, the latter is also satisfied. With their conversion rules, the schedulability of a child component is sufficiently analyzed such that it is schedulable if its parent component provides real-time guarantees that can be converted to the real-time guarantee that the child component demands.

1.2 Contributions

In our earlier work [Shin and Lee 2003], we presented exact schedulability conditions for a periodic task set scheduled by EDF or RM over the worst-case resource supply of a periodic resource model. In this paper, we extend this initial study with an enhanced definition of the proposed compositional real-time scheduling framework and the following results:

—We provide new utilization bounds that refine previous results [Saewong et al. 2002; Shin and Lee 2003] under EDF and RM scheduling with a new parameter. The proposed utilization bounds use as an additional parameter the relationship between a periodic resource model and the smallest period of tasks within a component.[2]

—We provide lower bounds to the interface utilizations of schedulable periodic interfaces of components, where the components include a periodic task set under EDF or RM scheduling.

—We evaluate the overheads that periodic interfaces incur in terms of utilization increase. We present analytical bounds to the overheads under EDF and RM scheduling and evaluates the overheads through extensive simulations, for providing insights regarding how to find a good periodic resource model.

The rest of this article is organized as follows: Section 2 describes the proposed compositional real-time scheduling framework and presents system models and problem statements. Section 3 introduces the periodic resource model. For a periodic task set scheduled by EDF or RM over a periodic resource model, Section 4 presents exact schedulability conditions and Section 5 provides the utilization bounds of the periodic task set. Section 6 presents the schedulable component abstraction bounds of periodic interfaces on the components that consist of a periodic task set scheduled by EDF or RM. Section 7 defines the overhead that the periodic interface incurs and presents its evaluations through analytical bounds and simulation results. Section 8 discusses the extensions of the proposed framework for supporting nonperiodic tasks and interacting tasks through data dependency or mutually exclusive resources. Finally, we conclude in Section 9 with discussion on future research.

[2]The effect of the smallest task period of a component upon component abstraction has been considered in Feng [2004], and here we develop utilization bounds considering such effect.

2. COMPOSITIONAL REAL-TIME SCHEDULING FRAMEWORK

This section presents an overview of the proposed compositional real-time scheduling framework and identifies the problems that need to be addressed for the proposed framework. Section 2.1 defines terms and notions used in the proposed framework and Section 2.2 provides the system models and assumptions of the framework. Section 2.3 describes the proposed framework; Section 2.4 presents the problem statement.

2.1 Scheduling Unit, Component, and Interface

Scheduling assigns resources according to scheduling algorithm in order to service workloads. We use the term *scheduling unit* to mean the basic unit of scheduling and define scheduling unit S as a triple (W, R, A), where W describes the workloads supported in the scheduling unit, R is a resource model that describes the resource allocations available to the scheduling unit, and A is a scheduling algorithm which describes how the workloads share the resources at all times. For scheduling unit $S(W, R, A)$, we assume that the workload W and the resource model R may not be synchronized. That is, if the workloads in W start at time t_w and R begins providing resource allocations at time t_r, then we assume t_w is not necessarily equal to t_r.

We consider that *component* C consists of a workload set W and a scheduling algorithm A for W, denoted as $C(W, A)$. The *resource demand* of component $C(W, A)$ represents the collective resource requirements that its workload set W requests under its scheduling algorithm A. The *demand-bound function* $\mathtt{dbf}_A(W, t)$ calculates the maximum possible resource demand that W could request to satisfy the timing requirements of all individual workloads under A within a time interval of length t.

Resource model R is said to be *dedicated* if it is exclusively available to a single scheduling unit, or *shared* otherwise. The *resource supply* of resource model R represents the amount of resource allocations that R provides. The *supply-bound function* $\mathtt{sbf}_R(t)$ calculates the minimum possible resource supplies that R provides during a time interval of length t. Resource model R is said to *satisfy* the resource demand of component $C(W, A)$ if $\mathtt{dbf}_A(W, t) \leq \mathtt{sbf}_R(t)$ for all interval length $t > 0$.

We now define the schedulability of scheduling unit as follows: scheduling unit $S(W, R, A)$ is said to be *schedulable*, if, and only if, the minimum possible resource supply of R can satisfy the maximum resource demand of W under A, i.e.,

$$\forall t \quad \mathtt{dbf}_A(W, t) \leq \mathtt{sbf}_R(t). \tag{1}$$

It should be noted that we define the schedulability of scheduling unit with the notion of worst-case resource supply of resource model R. Under the assumption that the workload W and the resource model R may not be synchronized, it is possible that W and R are aligned at the worst-case scenario, where W receives the minimum possible resource supply from R. Considering this, we define the schedulability such that all the timing requirements of W should

be satisfiable under A even with the (worst-case) minimum possible resource supply of R.

The *real-time interface* I of a component $C(W, A)$ specifies the collective real-time requirements of the component C, which W demands under A, without revealing the internal information of the component, such as the number of its workloads and its scheduling algorithm. A real-time interface I of component $C(W, A)$ is said to be *schedulable* if scheduling unit $S(W, R, A)$ is schedulable with $R = I$.

2.2 System Models

As a workload model, we consider the Liu and Layland periodic task model $T(p, e)$ [Liu and Layland 1973], where p is a period and e is an execution time requirement ($e \leq p$). The task utilization U_T of task T is defined as e/p. For a workload set $W = \{T_i\}$, a workload utilization U_W is defined as $\sum_{T_i \in W} U_{T_i}$. In this paper, let P_{min} denote the smallest task period in the workload set W. We consider that all tasks in a scheduling unit are synchronous, i.e., they release their initial jobs at the same time. We assume that each task is independent and preemptable with no preemption cost.

As a scheduling algorithm, we consider the earliest-deadline-first (EDF) algorithm, which is an optimal dynamic uniprocessor scheduling algorithm [Liu and Layland 1973] and the rate-monotonic (RM) algorithm, which is an optimal fixed-priority uniprocessor scheduling algorithm [Liu and Layland 1973].

As a resource model, we consider the periodic resource model $\Gamma(\Pi, \Theta)$ [Shin and Lee 2003] that can characterize the periodic behavior of resource allocations, where Π is a resource period and Θ is a periodic resource allocation time. The resource capacity U_Γ of Γ is defined as Θ/Π. Section 3 explains this periodic resource model in detail.

As a real-time interface model, we consider the periodic interface model $\mathcal{P}(P, E)$, where P is a period and E is an execution time requirement. The interface utilization $U_\mathcal{P}$ of $\mathcal{P}$ is E/P.

2.3 Compositional Framework Overview

In a hierarchical scheduling framework, a parent component provides resource allocations to its child components. Once a child component C_1 finds a schedulable periodic interface $\mathcal{P}_1$, it exports the real-time interface to its parent component. The parent component treats the real-time interface $\mathcal{P}_1$ as a single periodic task T_1. As long as the parent component satisfies the resource requirements imposed by the single periodic task T_1, the parent component is able to satisfy the resource demand of the child component C_1. This scheme makes it possible for a parent component to supply resources to its child components without controlling (or even understanding) how the child components schedule resources for their own tasks.

We define the *(periodic) component-abstraction* problem as deriving a real-time interface of a component. That is, the problem is to abstract the collective real-time requirements of the component as a single real-time requirement, called the real-time interface, without revealing the internal structure of the

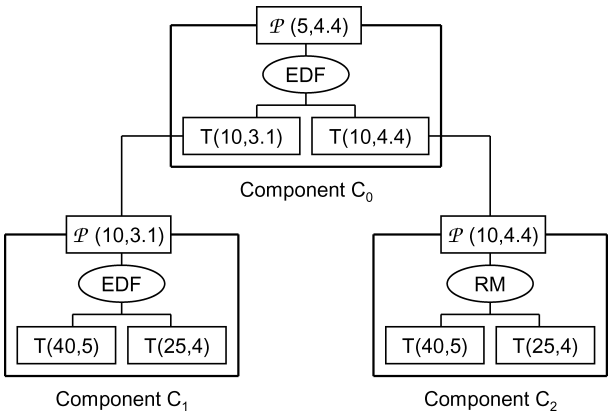

Fig. 2. Compositional real-time scheduling framework.

component, e.g., the number of tasks and its scheduling algorithm. We formulate this problem as follows: given a component $C_0 = C(W_0, A_0)$, the problem is to find an "optimal" schedulable periodic interface $\mathcal{P}_0 = \mathcal{P}(P_0, E_0)$. We define the optimality with respect to minimizing the interface utilization $U_{\mathcal{P}_0}$ of the periodic interface $\mathcal{P}_0$. That is, this problem is to find an optimal periodic resource model $\Gamma^* = \Gamma(\Pi^*, \Theta^*)$ that makes scheduling unit $S_0 = S(W_0, \Gamma^*, A_0)$ schedulable, if any. Here, the optimal schedulable periodic interface $\mathcal{P}_0 = \mathcal{P}(P_0, E_0)$ is $P_0 = \Gamma^*$ and $E_0 = \Theta^*$.

Example 2.1. As an example, let us consider component C_1 in Figure 2. Component C_1 has two periodic tasks under EDF scheduling, i.e., $C_1 = C(W_1, A_1)$, where $W_1 = \{T(40, 5), T(25, 4)\}$ and $A_1 = \mathsf{EDF}$. Now, we consider the problem of finding an optimal schedulable periodic interface $\mathcal{P}_1$ of C_1. This problem is equivalent to the problem of finding a periodic resource model $\Gamma^* = \Gamma(\Pi^*, \Theta^*)$ that makes scheduling unit $S_1 = S(W_1, \Gamma^*, A_1)$ schedulable with the minimum resource capacity U_{Γ^*}. When the period Π^* of Γ^* is given as 10, the optimal periodic resource model is $\Gamma^* = \Gamma(10, 3.1)$. Here, $\mathcal{P}_1 = \mathcal{P}(10, 3.1)$ is a solution to the component-abstraction problem.

We define the *(periodic) component-composition* problem as combining multiple components into a single component through real-time interfaces, preserving the principle of compositionality, i.e., the properties of components are held in a larger component. We formulate the component composition problem as follows: given component $C_0 = C(W_0, A_0)$ that consists of two subcomponents C_1 and C_2 under A_0, the problem is to find an "optimal" periodic interface $\mathcal{P}_0$ of C_0. Our approach is to develop the optimal schedulable periodic interfaces $\mathcal{P}_1$ and $\mathcal{P}_2$ of C_1 and C_2, respectively, and to consider $C_0 = C(W_0, A_0)$ as consisting of two periodic tasks, i.e., $W_0 = \{T_1, T_2\}$, where $T_i = \mathcal{P}_i$, $i = 1, 2$. The component-composition problem then becomes equivalent to the component-abstraction problem. Thus, we can address the component composition problem.

Example 2.2. As an example, let us consider component $C_0 = C(W_0, A_0)$ that consists of two subcomponents, C_1 and C_2, in Figure 2. Let $A_0 = \mathsf{EDF}$. We first determine the optimal schedulable periodic interfaces of the two subcomponents, C_1 and C_2. An optimal schedulable periodic interface $\mathcal{P}_1$ of C_1 is $\mathcal{P}_1 = \mathcal{P}(10, 3.1)$, and that of C_2 is $\mathcal{P}_2 = \mathcal{P}(10, 4.4)$. By treating $\mathcal{P}_i$ as a single periodic task T_i, for $i = 1, 2$, we can consider the workload set W_0 has two periodic tasks, i.e., $W = \{T_1, T_2\}$, where $T_1 = T(10, 3.1)$ and $T_2 = T(10, 4.4)$. Then, the problem of finding an optimal schedulable periodic interface $\mathcal{P}_0$ of C_0 is equivalent to the component-abstraction problem for C_0. Here, assuming the period Π^* of Γ^* is given as 5, $\Gamma^* = \Gamma(5, 4.4)$ is the optimal periodic resource model that makes scheduling unit $S(W_0, \Gamma^*, A_0)$ schedulable. Thus, $\mathcal{P}_0 = \mathcal{P}(5, 4.4)$ is a solution to the component-composition problem for C_0.

We define the *compositional real-time scheduling framework* as a hierarchical scheduling framework that supports the abstraction and composition of components. That is, it supports abstracting the collective real-time requirements of a component as a real-time interface and composing independently analyzed local timing properties into a global timing property.

It is desirable that the interface utilization $U_{\mathcal{P}}$ of a periodic interface $\mathcal{P}$ is equal to the workload utilization U_W of a workload set W. However, $U_{\mathcal{P}}$ can be larger than U_W. We define the *component-abstraction overhead* as $U_{\mathcal{P}}/U_W - 1$ to represent a normalized resource utilization increase.

2.4 Problem Statement

Based on the periodic resource model $\Gamma(\Pi, \Theta)$ that characterizes resource supply with a periodic behavior, we address the following problems:

—for a scheduling unit $S_0 = S(W_0, R_0, A_0)$, where $W_0 = \{T(p_i, e_i)\}$, $R_0 = \Gamma(\Pi_0, \Theta_0)$, and $A_0 = \mathsf{EDF/RM}$, the problems are to (1) analyze its schedulability and (2) derive its schedulable workload utilization;

—for a component $C_0 = C(W_0, A_0)$, where $W_0 = \{T(p_i, e_i)\}$ and $A_0 = \mathsf{EDF/RM}$, the problems are to (1) develop an optimal periodic interface $\mathcal{P}_0$ of C_0 and (2) derive the upper bound on the interface utilization of $\mathcal{P}_0$.

In addition, we define the notion of component-abstraction overhead for periodic interface. We then derive upper bounds on the overheads and evaluate the overheads through simulation results.

3. PERIODIC RESOURCE MODEL

To be able to analyze the schedulability of a scheduling component independent of its context, it is necessary to calculate the resource supply provided to the scheduling component. A resource model is to specify resource allocations that are provided to a scheduling component and to calculate the resource supply to the component. In this section, we briefly review a periodic resource model and its charateristics.

In our earlier work [Shin and Lee 2003], we proposed a periodic resource model $\Gamma(\Pi, \Theta)$, where Π is a period ($\Pi > 0$) and Θ is a periodic allocation time

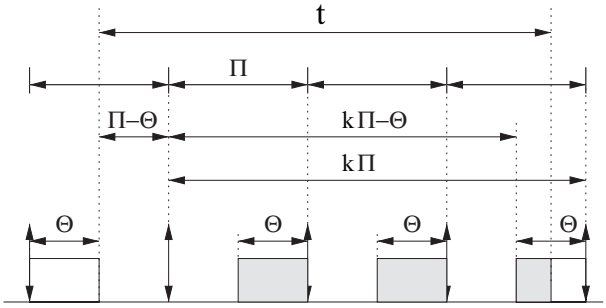

Fig. 3. The supply bound function of a periodic resource model $\Gamma(\Pi, \Theta)$ for $k = 3$.

$(0 < \Theta \leq \Pi)$. A resource capacity U_Γ of a periodic resource $\Gamma(\Pi, \Theta)$ is defined as Θ/Π. The periodic model $\Gamma(\Pi, \Theta)$ is defined to characterize the following property:

$$\text{supply}_\Gamma\big(k\Pi, (k+1)\Pi\big) = \Theta, \qquad \text{where } k = 0, 1, 2, \ldots. \tag{2}$$

For schedulability analysis, it is important to calculate the minimum resource supply of a resource model accurately. For a periodic model Γ, its supply-bound function $\texttt{sbf}_\Gamma(t)$ is defined to compute the minimum resource supply for every interval length t as follows:

$$\texttt{sbf}_\Gamma(t) = \begin{cases} t - (k+1)(\Pi - \Theta) & \text{if } t \in [(k+1)\Pi - 2\Theta, (k+1)\Pi - \Theta], \\ (k-1)\Theta & \text{otherwise,} \end{cases} \tag{3}$$

where $k = \max(\lceil (t - (\Pi - \Theta))/\Pi \rceil, 1)$. Figure 3 illustrates how the supply-bound function $\texttt{sbf}_\Gamma(t)$ is defined for $k = 3$.

Since the supply-bound function $\texttt{sbf}_\Gamma(t)$ is a discrete function, its linear lower-bound function $\texttt{lsbf}_\Gamma(t)$ is as follows:

$$\texttt{lsbf}_\Gamma(t) = \begin{cases} \frac{\Theta}{\Pi}\Big(t - 2(\Pi - \Theta)\Big) & \text{if } (t \geq 2(\Pi - \Theta)), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

We define the *service time* of a resource as the duration that it takes for the resource to provide a resource supply. For a periodic resource $\Gamma(\Pi, \Theta)$, we define a service time-bound function $\texttt{tbf}_\Gamma(t)$ to calculate the maximum service time of Γ required for a t-time-unit resource supply as follows:

$$\texttt{tbf}_\Gamma(t) = (\Pi - \Theta) + \Pi \cdot \left\lfloor \frac{t}{\Theta} \right\rfloor + \epsilon_t, \tag{5}$$

where

$$\epsilon_t = \begin{cases} \Pi - \Theta + t - \Theta\left\lfloor \frac{t}{\Theta} \right\rfloor & \text{if } \Big(t - \Theta\left\lfloor \frac{t}{\Theta} \right\rfloor > 0\Big). \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

4. SCHEDULABILITY ANALYSIS

For a scheduling unit $S(W, R, A)$, schedulability analysis is to determine whether a set of timing requirements imposed by the workload set W can be

satisfied with the resource supply of R under the scheduling algorithm A. The schedulability analysis is essential to solve the component-abstraction and-composition problems. In this section, we present exact conditions under which the schedulability of a scheduling unit can be satisfied, when the scheduling unit consists of a periodic workload set scheduled by EDF or RM and a periodic resource model. We then address the component-abstraction and-composition problems with a periodic interface using the exact schedulability conditions.

4.1 Schedulability Analysis under EDF Scheduling

For a periodic task set W under EDF scheduling, Baruah et al. [1990a] proposed the *demand-bound function* $\mathtt{dbf}_{\mathsf{EDF}}(W, t)$ that computes the total resource demand of W for every interval length t.

$$\mathtt{dbf}_{\mathsf{EDF}}(W, t) = \sum_{T_i \in W} \left\lfloor \frac{t}{p_i} \right\rfloor \cdot e_i. \tag{7}$$

For an approximate mathematical analysis, we define a linear upper-bound function $\mathtt{ldbf}_{\mathsf{EDF}}(W, t)$ of the demand-bound function $\mathtt{dbf}_{\mathsf{EDF}}(W, t)$ as follows:

$$\mathtt{ldbf}_{\mathsf{EDF}}(W, t) = U_W \cdot t \geq \mathtt{dbf}_{\mathsf{EDF}}(W, t),$$

where U_W is the utilization of the workload set W.

We present the following theorem to provide an exact condition under which the schedulability of scheduling unit $S(W, R, \mathsf{EDF})$ can be satisfied for the periodic resource model R.

THEOREM 4.1. *Scheduling unit $S(W, R, A)$ is schedulable, where $W = \{T_i = T(p_i, e_i)\}$ and $A = \mathsf{EDF}$, if, and only if,*

$$\forall 0 < t \leq LCM_W \quad \mathtt{dbf}_{\mathsf{EDF}}(W, t) \leq \mathtt{sbf}_R(t), \tag{8}$$

where LCM_W is the least common multiple of p_i for all $T_i \in W$.

PROOF. To show the necessity, we prove the contrapositive, i.e., if Equation (8) is false, there are some workload members of W that are not schedulable by EDF. If the total resource demand of W under EDF scheduling during t exceeds the total resource supply provided by R during t, there is clearly no feasible schedule.

To show the sufficiency, we prove the contrapositive, i.e., if all workload members of W are not schedulable by EDF, then Equation (8) is false. Let t_2 be the first instant at which a job of some workload member T_i of W misses its deadline. Let t_1 be the latest instant at which the resource supplied to W was idle or was executing a job whose deadline is after t_2. By the definition of t_1, there is a job whose deadline is before t_2 and which was released at t_1. Since T_i misses its deadline at t_2, the total demand placed on W in the time interval $[t_1, t_2)$ is greater than the total supply provided by R in the same time interval length $t_2 - t_1$. □

The schedulability condition in Theorem 4.1 is necessary in addressing the component-abstraction problem for a component with the EDF scheduling algorithm. We illustrate this with the following example.

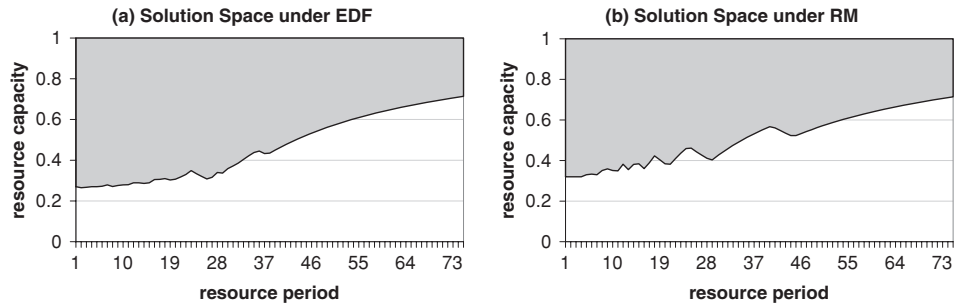

Fig. 4. Schedulable region of periodic resource $\Gamma(\Pi, \Theta)$: (a) under EDF scheduling in Example 4.1 and (b) under RM scheduling in Example 4.2.

Example 4.1 Let us consider a workload set $W_0 = \{T(50, 7), T(75, 9)\}$ and a scheduling algorithm $A_0 = \mathsf{EDF}$. The workload utilization U_{W_0} is 0.26. We now consider the problem of finding a schedulable periodic interface $\mathcal{P}_0 = \mathcal{P}(P_0, E_0)$ of component $C_0 = C(W_0, A_0)$. This problem is equivalent to finding a periodic resource model $\Gamma_0 = \Gamma(\Pi_0, \Theta_0)$ that makes scheduling unit $S_0 = S(W_0, \Gamma_0, A_0)$ schedulable. We can obtain a solution space of Γ_0 to this problem by simulating Equation (8). For any given resource period Π_0, we can find the smallest Θ_0 such that the scheduling unit S_0 is schedulable according to Theorem 4.1. Figure 4a shows such a solution space as the gray area for each integer resource period $\Pi_0 = 1, 2, \ldots, 75$. For instance, when $\Pi_0 = 10$, the minimum resource capacity U_{Γ_0} that guarantees the schedulability of S_0 is 0.28. So, $\Theta_0 = 2.8$. That is, the periodic interface $\mathcal{P}_0 = \mathcal{P}(10, 2.8)$ is an optimal schedulable interface of C_0, when P is given as 10.

4.2 Schedulability Analysis under RM Scheduling

For a periodic task set W under RM scheduling, Lehoczky et al. [1989] proposed a demand-bound function $\mathtt{dbf}_{\mathsf{RM}}(W, t, i)$ that computes the worst-case cumulative resource demand of a task T_i for an interval of length t.

$$\mathtt{dbf}_{\mathsf{RM}}(W, t, i) = e_i + \sum_{T_k \in \mathtt{HP}_W(i)} \left\lceil \frac{t}{p_k} \right\rceil \cdot e_k, \tag{9}$$

where $\mathtt{HP}_W(i)$ is the set of higher-priority tasks than T_i in W.

For a task T_i over a resource model R, the worst-case response time $r_i(R)$ of T_i can be computed as follows:

$$r_i(R) = \min\{t \mid \mathtt{dbf}_{\mathsf{RM}}(W, t, i) \leq \mathtt{sbf}_R(t)\}.$$

We present the following theorem to provide an exact condition under which the schedulability of scheduling unit $S(W, R, \mathsf{RM})$ can be satisfied for the periodic resource model R.

THEOREM 4.2. *Scheduling unit $S(W, R, A)$ is schedulable, where $W = \{T_i = T(p_i, e_i)\}$ and $A = \mathsf{RM}$, if, and only if,*

$$\forall T_i \in W \quad \exists t_i \in [0, p_i] \quad \mathtt{dbf}_{\mathsf{RM}}(W, t_i, i) \leq \mathtt{sbf}_R(t_i). \tag{10}$$

PROOF. Task T_i completes its execution requirement at time $t \in [0, p_i]$, if, and only if, all the execution requirements from all the jobs of higher-priority tasks than T_i and e_i, the execution requirement of T_i, are completed at time t_i.

The total of such requirements is given by $\mathtt{dbf}_{\mathsf{RM}}(W, t_i, i)$, and they are completed at t_i if, and only if, $\mathtt{dbf}_{\mathsf{RM}}(W, t_i, i) = \mathtt{sbf}_R(t_i)$ and $\mathtt{dbf}_{\mathsf{RM}}(W, t_i', i) > \mathtt{sbf}_R(t_i')$ for $0 \leq t_i' < t_i$. It follows that a necessary and sufficient condition for T_i to meet its deadline is the existence of a $t_i \in [0, p_i]$ such that $\mathtt{dbf}_{\mathsf{RM}}(W, t_i, i) = \mathtt{sbf}_R(t_i)$.

The entire task set is schedulable if, and only if, each of the tasks is schedulable. This means that there exists a $t_i \in [0, p_i]$ such that $\mathtt{dbf}_{\mathsf{RM}}(W, t_i, i) = \mathtt{sbf}_R(t_i)$ for each task $T_i \in W$. □

We present the following example to show that using Theorem 4.2, we can address the component-abstraction problem for a component with the RM scheduling algorithm.

Example 4.2 Let us consider a workload set $W_0 = \{T(50, 7), T(75, 9)\}$ and a scheduling algorithm $A_0 = \mathsf{RM}$. The workload utilization U_{W_0} is 0.26. We now consider the problem of finding a schedulable periodic interface $\mathcal{P}_0 = \mathcal{P}(P_0, E_0)$ of component $C_0 = C(W_0, A_0)$. This problem is equivalent to finding a periodic resource model $\Gamma_0 = \Gamma(\Pi_0, \Theta_0)$ that makes scheduling unit $S_0 = S(W_0, \Gamma_0, A_0)$ schedulable. We can obtain a solution space of Γ_0 to this problem by simulating Equation (10). For any given resource period Π_0, we can find the smallest Θ_0 such that the scheduling unit S_0 is schedulable according to Theorem 4.2. Figure 4b shows such a solution space as the gray area for each integer resource period $\Pi_0 = 1, 2, \ldots, 75$. For instance, when $\Pi_0 = 10$, the minimum resource capacity U_{Γ_0} that guarantees the schedulability of S_0 is 0.35. So, $\Theta_0 = 3.5$. That is, the periodic interface $\mathcal{P}_0 = \mathcal{P}(10, 3.5)$ is an optimal schedulable interface of C_0, when P is given as 10.

5. SCHEDULABLE WORKLOAD UTILIZATION BOUNDS

This section starts by presenting the definition of (schedulable workload) utilization bound and provides the utilization bounds on the periodic resource model under EDF and RM scheduling.

Definition 5.1. The *(schedulable workload) utilization bound*, denoted as UB, of a periodic task set W on resource model R under scheduling algorithm A is defined such that the scheduling unit $S(W, R, A)$ is schedulable if the workload utilization (U_W) is no greater than the utilization bound.

The utilization bound is particularly suited for on-line acceptance tests. When checking whether or not a new periodic task can be scheduled with the existing tasks, computing the utilization bound takes a constant amount of time, which is much faster than doing an exact schedulability analysis based on a demand-bound function.

Liu and Layland [1973] presented the utilization bounds of a periodic task set W on a dedicated resource R_D under EDF and RM scheduling; the utilization bounds are 1 under EDF and $n(2^{1/n} - 1)$ under RM, respectively, where n is

the number of tasks in W. This section extends these seminal results with the periodic resource model.

We observe that a key relationship between a workload set W and a periodic resource model $\Gamma(\Pi, \Theta)$ is the *period multiple relationship*, which basically indicates how many times that $\Gamma(\Pi, \Theta)$ can provide W with the whole periodic allocation time of Θ during a time interval of length P_{min} under the worst-case resource supply scenario, where P_{min} is the smallest task period of W. Here, we define the functions, $K_A(P_{min}, \Gamma(\Pi, \Theta))$, that computes this period multiple relationship under RM and EDF scheduling as follows:

—under RM scheduling, $K_{\mathsf{RM}}(P_{min}, \Gamma(\Pi, \Theta))$ is defined as follows:

$$K_{\mathsf{RM}}(P_{min}, \Gamma(\Pi, \Theta)) = \max\{k \text{ is an integer } \mid (k+1)\Pi - \Theta < P_{min}\}; \quad (11)$$

—under EDF scheduling, $K_{\mathsf{EDF}}(P_{min}, \Gamma(\Pi, \Theta))$ is defined as follows:

$$K_{\mathsf{EDF}}(P_{min}, \Gamma(\Pi, \Theta)) = \max\{k \text{ is an integer } \mid (k+1)\Pi - \Theta - \frac{k\Theta}{k+2} < P_{min}\}. \quad (12)$$

Based on the period multiple relationship between $\Gamma(\Pi, \Theta)$ and W with P_{min}, this section presents the utilization bounds of W on $\Gamma(\Pi, \Theta)$ as a function of P_{min} under EDF scheduling and as a function of n and P_{min} under RM scheduling, respectively. In this direction, this section presents new utilization bound results that refine the previous results [Saewong et al. 2002; Shin and Lee 2003].

5.1 Utilization Bound under EDF Scheduling

For scheduling unit $S(W, R, A)$, where $W = \{T(p_i, e_i)\}$, $R = \Gamma(\Pi, \Theta)$, and $A = \mathsf{EDF}$, we represent its utilization bound as a function of P_{min}, denoted as $\mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$, such that $S(W, R, A)$ is schedulable if $U_W \leq \mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$. We present the following theorem to introduce the utilization bound $\mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$.

THEOREM 5.1. *For scheduling unit $S(W, R, A)$, where $W = \{T(p_i, e_i)\}$, $R = \Gamma(\Pi, \Theta)$, and $A = \mathsf{EDF}$, its utilization bound $\mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$, as a function of P_{min}, is*

$$\mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min}) = \frac{k \cdot U_\Gamma}{k + 2(1 - U_\Gamma)}, \quad (13)$$

where $k = K_{\mathsf{EDF}}(P_{min}, R)$.

PROOF. Let P^*_{EDF} denote $(k+1)\Pi - 2\Theta$, where $k \geq 1$. It is shown in Figure 5 that P^*_{EDF} is the largest time instant $(k+1)\Pi - 2\Theta$, for integer k, that is smaller than P_{min}. Let $\epsilon = 2\Theta/(k+2)$.

In this proof, we consider four cases in terms of an interval length t: (1) $0 \leq t \leq P^*_{\mathsf{EDF}} + \epsilon$, (2) $P^*_{\mathsf{EDF}} + \epsilon < t \leq P^*_{\mathsf{EDF}} + \Theta$, (3) $P^*_{\mathsf{EDF}} + \Theta < t \leq P^*_{\mathsf{EDF}} + \Pi$, and (4) $t < P^*_{\mathsf{EDF}} + \Pi$. For each of the above four cases, we want to show that if $U_W \leq \mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$, then

$$\mathtt{dbf}_{\mathsf{EDF}}(W, t) \leq \mathtt{sbf}_\Gamma(t).$$

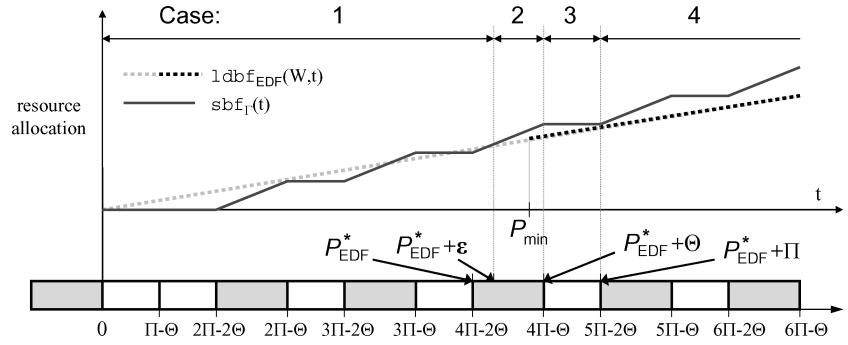

Fig. 5. Four cases to consider in deriving a utilization bound under EDF scheduling.

Case 1. We first consider a case where $0 \le t \le P^*_{\mathsf{EDF}} + \epsilon$. Since $P^*_{\mathsf{EDF}} + \epsilon < P_{min}$, this is the case where $0 \le t < P_{min}$. From the definition of $\mathtt{dbf}_{\mathsf{EDF}}(W,t)$, then,

$$\forall 0 \le t \le P^*_{\mathsf{EDF}} + \epsilon \quad \mathtt{dbf}_{\mathsf{EDF}}(W,t) = 0.$$

Since $\mathtt{sbf}_\Gamma(t) \ge 0$ for all $t \ge 0$, it follows that

$$\forall 0 \le t \le P^*_{\mathsf{EDF}} + \epsilon \quad \mathtt{dbf}_{\mathsf{EDF}}(W,t) \le \mathtt{sbf}_\Gamma(t).$$

Case 2. We consider a case where an interval length t can be longer than P_{min}, but is no longer than $P^*_{\mathsf{EDF}} + \Theta$. That is,

$$P^*_{\mathsf{EDF}} + \epsilon < t \le P^*_{\mathsf{EDF}} + \Theta.$$

It is easy to see that, from the definition of $\mathtt{sbf}_\Gamma(t)$,

$$\forall P^*_{\mathsf{EDF}} + \epsilon < t \le P^*_{\mathsf{EDF}} + \Theta \quad \mathtt{sbf}_\Gamma(t) = (k-1)\Theta + t - P^*_{\mathsf{EDF}}.$$

When $t = P^*_{\mathsf{EDF}} + \epsilon$, it follows that

$$\begin{aligned} \mathtt{sbf}_\Gamma(P^*_{\mathsf{EDF}} + \epsilon) &= (k-1)\Theta + \epsilon \\ &= k\Theta - \frac{k}{k+2}\Theta. \end{aligned} \tag{14}$$

When $t = P^*_{\mathsf{EDF}} + \epsilon$, it follows that

$$\begin{aligned} \mathtt{ldbf}_{\mathsf{EDF}}(W,t) &= U_W \cdot t \\ &\le \frac{k \cdot U_\Gamma}{k + 2(1-U_\Gamma)} \cdot \left(P^*_{\mathsf{EDF}} + \epsilon\right) \\ &= \frac{k\Theta}{(k+2)\Pi - 2\Theta} \cdot \left(\left((k+2)\Pi - 2\Theta\right) + \left(\epsilon - \Pi\right)\right) \\ &= k\Theta + \frac{k\Theta}{(k+2)\Pi - 2\Theta} \cdot \left(\epsilon - \Pi\right) \\ &= k\Theta + \frac{k\Theta}{(k+2)\Pi - 2\Theta} \cdot \frac{-(k+2)\Pi + 2\Theta}{k+2} \\ &= k\Theta - \frac{k}{k+2}\Theta. \end{aligned} \tag{15}$$

From Equations (14) and (15), we can see that when $t = P^*_{\mathsf{EDF}} + \epsilon$, $\mathtt{ldbf}_{\mathsf{EDF}}(W,t) = \mathtt{sbf}_\Gamma(t)$. In this case, one can see that $\mathtt{sbf}_\Gamma(t)$ increases linearly in t with a slope of 1, and so does $\mathtt{ldbf}_{\mathsf{EDF}}(W,t)$ with a smaller slope of U_W than 1. Thus, considering $\mathtt{dbf}_{\mathsf{EDF}}(W,t) \le \mathtt{ldbf}_{\mathsf{EDF}}(W,t)$, in this case, it is easy to see that

$$\forall P^*_{\mathsf{EDF}} + \epsilon < t \le P^*_{\mathsf{EDF}} + \Theta \quad \mathtt{dbf}_{\mathsf{EDF}}(W,t) \le \mathtt{sbf}_\Gamma(t).$$

Case 3. We consider a case where an interval length t can be longer than P_{min}, particularly longer than $P^*_{\mathsf{EDF}} + \Theta$, but is no longer than $P^*_{\mathsf{EDF}} + \Pi$. That is,

$$P^*_{\mathsf{EDF}} + \Theta < t \le P^*_{\mathsf{EDF}} + \Pi.$$

From the definition of $\mathtt{dbf}_{\mathsf{EDF}}(W,t)$, thus,

$$\forall P^*_{\mathsf{EDF}} + \Theta < t \le P^*_{\mathsf{EDF}} + \Pi \quad \mathtt{dbf}_{\mathsf{EDF}}(W,t) < U_W \cdot (P^*_{\mathsf{EDF}} + \Pi).$$

We then have

$$\begin{aligned}
\mathtt{dbf}_{\mathsf{EDF}}(W,t) &< U_W \cdot (P^*_{\mathsf{EDF}} + \Pi) \\
&\le \frac{k \cdot U_\Gamma}{k + 2(1 - U_\Gamma)}\big((k+2)\Pi - 2\Theta\big) \\
&= k \cdot \Theta \qquad\qquad \text{since } U_\Gamma = \Theta/\Pi \\
&= \mathtt{sbf}_\Gamma(t).
\end{aligned}$$

Case 4. We consider a case where an interval length t is longer than $P^*_{\mathsf{EDF}} + \Pi$, which is longer than P_{min}. That is,

$$P^*_{\mathsf{EDF}} + \Pi < t.$$

We then have

$$\begin{aligned}
\mathtt{dbf}_{\mathsf{EDF}}(W,t) &\le U_W \cdot t \\
&\le \frac{k \cdot U_\Gamma}{k + 2(1 - U_\Gamma)} \cdot t \\
&\le U_\Gamma \cdot t \qquad\qquad \text{since } 0 < U_\Gamma \le 1 \\
&= \mathtt{lsbf}_\Gamma(t) \\
&\le \mathtt{sbf}_\Gamma(t).
\end{aligned}$$

For each of the four cases, we showed that if $U_W \le \mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$, then $\mathtt{dbf}_{\mathsf{EDF}}(W,t) \le \mathtt{sbf}_\Gamma(t)$, that is, the scheduling unit $S(W, \Gamma(\Pi,\Theta), \mathsf{EDF})$ is schedulable according to Theorem 4.1. □

It should be noted that the utilization bound $\mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$ becomes 1 without regard to P_{min} if the resource capacity U_Γ of periodic resource model Γ is 1, i.e., Γ represents a dedicated resource. Thus, $\mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$ is a generalization of the result of Liu and Layland [1973].

Example 5.1 As an example, we consider a scheduling unit $S_0 = S(W_0, \Gamma_0, A_0)$, where $\Gamma_0 = \Gamma(10, 4)$ and $A_0 = \mathsf{EDF}$. Then, the resource capacity U_{Γ_0} of Γ_0 is 0.4. We assume that the smallest task period P_{min}

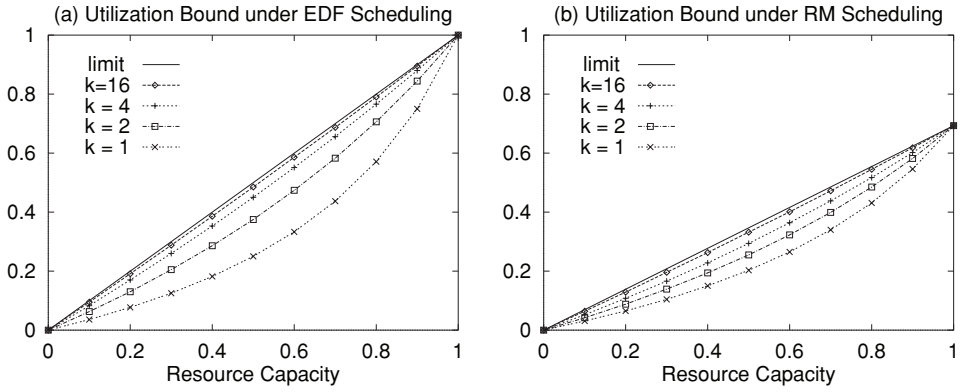

Fig. 6. Utilization bound as a function of its resource capacity: (a) under EDF scheduling and (b) under RM scheduling.

of the workload set W_0 is greater than 50, i.e., $P_{min} \geq 50$. Here, k is 4 by the definition of $K_{\mathsf{EDF}}(P_{min}, R)$. According to Theorem 5.1, the EDF utilization bound $\mathsf{UB}_{\Gamma_0,\mathsf{EDF}}(50)$ is 0.32. That is, if $U_{W_0} \leq 0.32$, then the scheduling unit S_0 is schedulable.

Figure 6a shows the effect of resource period, in terms of k, on the utilization bound as a function of resource capacity under EDF scheduling, where $k = K_{\mathsf{EDF}}(P_{min}, \Gamma(\Pi, \Theta))$. The solid line, labeled "limit," shows the limit of the utilization bound of a periodic resource, which is obtained when $k = \infty$. The other curves show the utilization bounds of a periodic resource when k is given as shown in the corresponding labels. It is shown that as k increases, the utilization bound of a periodic resource converges to its limit.

5.2 Utilization Bound under RM Scheduling

For scheduling unit $S(W, R, A)$, where $W = \{T(p_1, e_1), \ldots, T(p_n, e_n)\}$, $R = \Gamma(\Pi, \Theta)$, $A = \mathsf{RM}$, Saewong et al. [2002] represented its utilization bound as a function of n, denoted as $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n)$. They presented the following result to introduce the utilization bound $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n)$, derived from the Liu and Layland [1973]'s utilization bound.

LEMMA 5.1 (THEOREM 7 IN SAEWONG ET AL. [2002]). *For scheduling unit $S(W, R, A)$, where $W = \{T(p_1, e_1), \ldots, T(p_n, e_n)\}$, $R = \Gamma(\Pi, \Theta)$, $A = \mathsf{RM}$, and $p_i \geq 2\Pi - \Theta$, $1 \leq i \leq n$, its utilization bound $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n)$ is*

$$\mathsf{UB}_{\Gamma,\mathsf{RM}}(n) = n\left[\left(\frac{3 - U_\Gamma}{3 - 2 \cdot U_\Gamma}\right)^{1/n} - 1\right].$$

We now present a different utilization bound as a function of n and P_{min}, denoted as $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min})$. We provide the following theorem to introduce $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min})$.

THEOREM 5.2. *For scheduling unit $S(W, R, A)$, where $W = \{T(p_1, e_1), \ldots, T(p_n, e_n)\}$, $R = \Gamma(\Pi, \Theta)$, $A = \mathsf{RM}$, and $p_i \geq 2\Pi - \Theta$, $1 \leq i \leq n$, its*

utilization bound $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min})$ *is*

$$\mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min}) = U_\Gamma \cdot n\Big[\Big(\frac{2 \cdot k + 2(1 - U_\Gamma)}{k + 2(1 - U_\Gamma)}\Big)^{1/n} - 1\Big], \tag{16}$$

where $k = K_{\mathsf{RM}}(P_{min}, R)$.

PROOF. The proof is shown in the Appendix. □

It should be noted that the utilization bound $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min})$ becomes the Liu and Layland [1973]'s RM utilization bound, which is $n(2^{1/n} - 1)$, without regard to P_{min} if the capacity of periodic resource U_Γ is 1, i.e., the periodic resource is essentially a dedicated resource. Thus, $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min})$ is a generalization of the result of Liu and Layland [1973].

Example 5.2 As an example, we consider a scheduling unit $S_0 = S(W_0, \Gamma_0, A_0)$, where $\Gamma_0 = \Gamma(10, 4)$ and $A_0 = \mathsf{RM}$. Let $\Pi_0 = 10$ and $\Theta_0 = 4$. Then, the resource capacity U_{Γ_0} of Γ_0 is 0.4. We assume that the smallest task period P_{min} of the workload set W_0 is greater than 50, i.e., $P_{min} \geq 50$. Here, k is 4 according to Equation (16). According to Theorem 5.2, the RM utilization bound $\mathsf{UB}_{\Gamma_0,\mathsf{EDF}}(50)$ is 0.27. That is, if $U_{W_0} \leq 0.27$, then the scheduling unit S_0 is schedulable.

Figure 6b shows the effect of resource period, in terms of k, on the utilization bound as a function of resource capacity under RM scheduling, where $k = K_{\mathsf{RM}}(P_{min}, \Gamma(\Pi, \Theta))$. The solid line, labeled "limit," shows the limit of the utilization bound, which is obtained when $k = \infty$. The other curves show the utilization bound when k is given as shown in their labels. It is shown in the graph that as k increases, the utilization bound of a periodic resource converges to its limit.

6. SCHEDULABLE COMPONENT ABSTRACTION BOUNDS

For a component $C(W, A)$, we now explain how to derive a lower-bound B of the interface utilization ($U_\mathcal{P}$) of $\mathcal{P}(P, E)$, such that $\mathcal{P}(P, E)$ is a schedulable interface of $C(W, A)$ if $B \leq U_\mathcal{P}$. If there exists such a bound B, we call it the *(schedulable-component) abstraction bound* of periodic interface on the component $C(W, A)$.

We define the period multiple relationship between a periodic task set W and a periodic interface $\mathcal{P}(P, E)$, similarly to that between W and a periodic resource model $\Gamma(\Pi, \Theta)$. Given P_{min} and $\mathcal{P}(P, E)$, where P_{min} is the smallest task period of W, this relationship basically represents how many times that $\mathcal{P}(P, E)$ can be given the whole periodic allocation time of E during a time interval $[t, t + P_{min})$, for all $t > 0$, at the worst-case resource supply scenario. Here, we define $K_A(P_{min}, \mathcal{P}(P, E))$ that computes this relationship by substituting Π and Θ in Equations (11) and (12) with P and E, respectively, as follows:

—under RM scheduling, $K_{\mathsf{RM}}(P_{min}, \mathcal{P}(P, E))$ is defined as follows:

$$K_{\mathsf{RM}}(P_{min}, \mathcal{P}(P, E)) = \max\{k \text{ is an integer } \mid (k + 1)P - E < P_{min}\}; \tag{17}$$

—under EDF scheduling, $K_{\mathsf{EDF}}(P_{min}, \mathcal{P}(P,E))$ is defined as follows:

$$K_{\mathsf{EDF}}(P_{min}, \mathcal{P}(P,E)) = \max\left\{k \text{ is an integer} \mid (k+1)P - E - \frac{kE}{k+2} < P_{min}\right\}. \tag{18}$$

Using the period multiple relationship between W and $\mathcal{P}(P,E)$, this section presents the abstraction bounds of periodic interface $\mathcal{P}(P,E)$ on a component $C(W,A)$ as a function of k, denoted as $\mathsf{AB}_{W,A}(k)$, under EDF and RM scheduling.

The abstraction bounds can be used in addressing the component abstraction problem as follows. Given a component $C^* = C(W^*, A^*)$ and k^*, for some positive integer k^*, we consider the problem of finding a periodic interface $\mathcal{P}^* = \mathcal{P}(P^*, E^*)$, such that $\mathcal{P}^*$ is a schedulable interface of C^* and $K_A(P^*_{min}, \mathcal{P}^*) = k^*$, where P^*_{min} is the smallest task period of W^*. We can obtain a solution space to this problem as follows: for all interface period $P^* \in [1, P^*_{min}]$, we compute a range of $[E^*_{min}, E^*_{max})$, where E^*_{min} and E^*_{max} are defined as follows:

– $E^*_{min} = \max(E_0, E_1)$ (19)

where

$$E_0 = \min\left\{E \mid (k^*+1)P^* - E - \frac{k^* \cdot E}{k^*+2} \le P_{min}\right\},$$
$$E_1 = P^* \cdot \mathsf{AB}_{W,A}(k^*).$$

and

– $E^*_{max} = \min(E_2, P^*)$ (20)

where
$$E_2 = \min\left\{E \mid (k^*+2)P^* - E - \frac{(k^*+1)E}{k^*+3} \le P_{min}\right\}.$$

A range of $[E^*_{min}, E^*_{max})$ is said to be *valid* if $E^*_{min} \le E^*_{max}$. According to the definition of the abstraction bound, a periodic interface $\mathcal{P}(P^*, E^*)$ is schedulable for C^* if $E^* \in [E^*_{min}, E^*_{max})$ with $E^*_{min} \le E^*_{max}$.

6.1 Abstraction Bound under EDF Scheduling

For component $C(W,A)$, where $W = \{T(p_i, e_i)\}$ and $A = \mathsf{EDF}$, we present the following corollary from Theorem 5.1 to introduce the abstraction bound of its periodic interface $\mathcal{P}$ as a function of k, denoted as $\mathsf{AB}_{W,\mathsf{EDF}}(k)$.

COROLLARY 6.1. *For component $C(W,A)$, where $W = \{T(p_i,e_i)\}$, and $A = \mathsf{EDF}$, the abstraction bound $\mathsf{AB}_{W,\mathsf{EDF}}(k)$ of a periodic interface $\mathcal{P}(P,E)$ is*

$$\mathsf{AB}_{W,\mathsf{EDF}}(k) = \frac{(k+2)\cdot U_W}{k + 2U_W}, \tag{21}$$

where k represents the period multiple relationship of W and $\mathcal{P}(P,E)$, such that $k = K_{\mathsf{EDF}}(P_{min}, \mathcal{P}(P,E))$ defined in Equation (18).

PROOF. Theorem 5.1 states that scheduling unit $S(W, \Gamma(\Pi,\Theta), \mathsf{EDF})$ is schedulable if $U_W \le \mathsf{UB}_{\Gamma,\mathsf{EDF}}(P_{min})$, i.e.,

$$U_W \le \frac{k \cdot U_\Gamma}{k + 2(1-U_\Gamma)}.$$

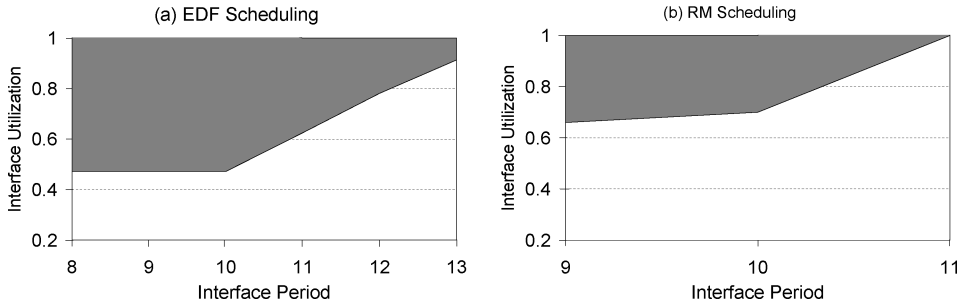

Fig. 7. Spaces of schedulable periodic interfaces, obtained through the abstraction bounds, for (a) example 6.1 under EDF scheduling and (b) example 6.2 under RM scheduling.

That is, $S(W, \Gamma(\Pi, \Theta), \mathsf{EDF})$ is schedulable if

$$U_\Gamma \geq \frac{(k+2) \cdot U_W}{k + 2U_W}. \tag{22}$$

Suppose, there is a periodic resource model $\Gamma(\Pi', \Theta')$ that satisfies Equation (22). Then, the periodic interface $\mathcal{P}(P', E')$, where $P' = \Pi'$ and $E' = \Theta'$, is a schedulable interface of $C(W, \mathsf{EDF})$. □

Example 6.2. As an example, we consider a component $C^* = C(W^*, A^*)$, where $W^* = \{T(33, 5), T(75, 7), T(100, 10)\}$ and $A^* = \mathsf{EDF}$. The workload utilization U_{W^*} is 0.35, and $P_{min} = 33$. We consider finding a periodic interface $\mathcal{P}^* = \mathcal{P}(P^*, E^*)$ of C^* through the abstraction bound. Assume that the period multiple relationship between W and $\mathcal{P}^*$ is given by $k = 3$, where $k = K_{\mathsf{EDF}}(P_{min}, \mathcal{P}^*)$. According to Corollary 6.1, $\mathsf{AB}_{W,\mathsf{EDF}}(3) = 0.47$. Under the constraint of $k = 3$, we can compute E^*_{min} and E^*_{max} for all interface period $P^* \in [1, 33]$, where E^*_{min} and E^*_{max} are defined in Equations (19) and (20). In this example, E^* has a valid range of $[E^*_{min}, E^*_{max})$ for all interface period $P^* \in [8, 13]$, as shown in Figure 7a. That is, the shaded region of this figure shows the space of schedulable periodic interfaces of the component C^*.

Figure 8a shows the effect of interface period, in terms of k, on the abstraction bound as a function of workload utilization under EDF scheduling, where $k = K_{\mathsf{EDF}}(P_{min}, \mathcal{P}(P, E))$. The solid line, labeled "limit," shows the limit of the abstraction bound, which is obtained when $k = \infty$. The other curves show the abstraction bounds when k is given as shown in their labels. It is shown that as k increases, the abstraction bound of a periodic interface converges to its limit, i.e., goes closer to the workload utilization.

6.2 Abstraction Bound under RM Scheduling

For component $C(W, A)$, where $W = \{T(p_i, e_i)\}$ and $A = \mathsf{RM}$, we present the following corollary from Theorem 5.2 to introduce the abstraction bound of its periodic interface $\mathcal{P}$ as a function of k, denoted as $\mathsf{AB}_{W,\mathsf{RM}}(k)$.

COROLLARY 6.2. *For component $C(W, A)$, where $W = \{T(p_1, e_1), \ldots, T(p_n, e_n)\}$ and $A = \mathsf{RM}$, the abstraction bound $\mathsf{AB}_{W,\mathsf{RM}}(k)$ of a periodic*

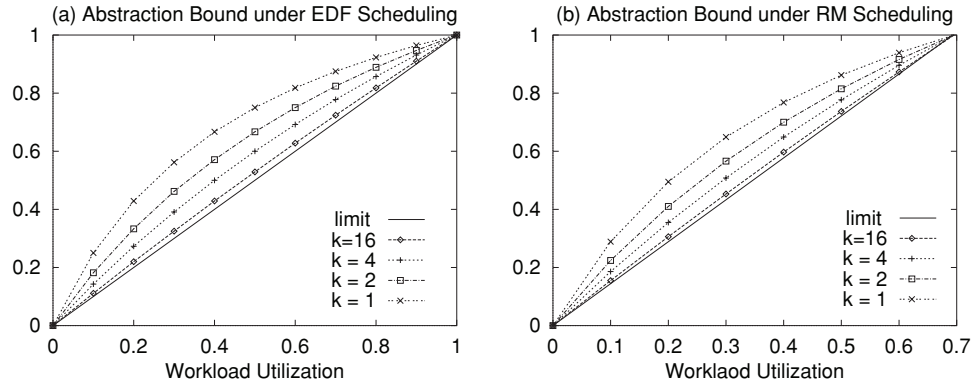

Fig. 8. Abstraction bound as a function of workload utilization: (a) under EDF scheduling and (b) under RM scheduling.

interface $\mathcal{P}(P, E)$ *is*

$$\mathsf{AB}_{W,\mathsf{RM}}(k) = \frac{U_W}{\log\left(\frac{2k+2(1-U_W)}{k+2(1-U_W)}\right)}, \tag{23}$$

where k *represents the period multiple relationship of* W *and* $\mathcal{P}(P, E)$*, such that* $k = K_{\mathsf{RM}}(P_{min}, \mathcal{P}(P, E))$ *defined in Equation (17).*

PROOF. Theorem 5.2 states that the scheduling unit $S(W, \Gamma(\Pi, \Theta), \mathsf{RM})$ is schedulable if $U_W \leq \mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min})$, i.e.,

$$U_W \leq U_\Gamma \cdot n\Big[\Big(\frac{2k + 2(1 - U_\Gamma)}{k + 2(1 - U_\Gamma)}\Big)^{1/n} - 1\Big]. \tag{24}$$

When n is large, we have

$$n\Big[\Big(\frac{2k + 2(1 - U_\Gamma)}{k + 2(1 - U_\Gamma)}\Big)^{1/n} - 1\Big] \simeq \log\Big(\frac{2k + 2(1 - U_\Gamma)}{k + 2(1 - U_\Gamma)}\Big). \tag{25}$$

From Equations (24) and (25), it follows

$$U_\Gamma \;\geq\; \frac{U_W}{\log\left(\frac{2k+2(1-U_\Gamma)}{k+2(1-U_\Gamma)}\right)}.$$

Since $U_W \leq U_\Gamma$, we have

$$\log\Big(\frac{2k + 2(1 - U_W)}{k + 2(1 - U_W)}\Big) \leq \log\Big(\frac{2k + 2(1 - U_\Gamma)}{k + 2(1 - U_\Gamma)}\Big). \tag{26}$$

From Equation (26), the scheduling unit $S(W, \Gamma(\Pi, \Theta), \mathsf{RM})$ is schedulable, if

$$U_\Gamma \;\geq\; \frac{U_W}{\log\left(\frac{2k+2(1-U_W)}{k+2(1-U_W)}\right)}.$$

Then, the periodic interface $\mathcal{P}(P, E)$ is a schedulable interface of $C(W, A)$, where $P = \Pi$ and $E = \Theta$. □

Example 6.3. As an example, we consider a component $C^* = C(W^*, A^*)$, where $W^* = \{T(33, 5), T(75, 7), T(100, 10)\}$ and $A^* = \mathsf{RM}$. The workload utilization U_{W^*} is 0.35, and $P_{min} = 33$. We consider finding a periodic interface $\mathcal{P}^* = \mathcal{P}(P^*, E^*)$ of C^* through the abstraction bound. Assume that the period multiple relationship between W and $\mathcal{P}^*$ is given by $k = 3$, where $k = K_{\mathsf{EDF}}(P_{min}, \mathcal{P}^*)$. According to Corollary 6.2, $\mathsf{AB}_{W^*,\mathsf{RM}}(3) = 0.66$. Under the constraint of $k = 3$, we can compute E^*_{min} and E^*_{max} for all interface period $P^* \in [1, 33]$, where E^*_{min} and E^*_{max} are defined in Equations (19) and (20). In this example, E^* has a valid range of $[E^*_{min}, E^*_{max})$ only for all interface period $P^* \in [9, 11]$, as shown in Figure 7b. That is, the shaded region of this figure shows the space of schedulable periodic interfaces of the component C^*.

Figure 8b shows the effect of interface period, in terms of k, on the abstraction bound as a function of workload utilization under RM scheduling, where $k = K_{\mathsf{RM}}(P_{min}, \mathcal{P}(P, E))$. The solid line, labeled "limit," shows the limit of the abstraction bound, which is obtained when $k = \infty$. The other curves show the abstraction bounds when k is given as shown in their labels. It is shown that as k increases, the abstraction bound of a periodic interface converges to its limit.

7. COMPONENT ABSTRACTION OVERHEADS: ANALYTICAL BOUNDS AND SIMULATION RESULTS

In this section, we address the problem of evaluating the overhead that a periodic interface incurs in terms of utilization increase, to solve the component abstraction problem. For a periodic interface $\mathcal{P}$ of component $C(W, A)$, we define its *(component) abstraction overhead* ($O_{\mathcal{P}}$) as

$$O_{\mathcal{P}} = \frac{U_{\mathcal{P}} - U_W}{U_W}, \tag{27}$$

where $U_{\mathcal{P}}$ and U_W are the utilizations of $\mathcal{P}$ and W, respectively.

Section 7.1 derives upper bounds to the abstraction overheads under EDF and RM scheduling; Section 7.2 evaluates the abstraction overheads through simulation.

7.1 Component-Abstraction Overhead Bounds

This section introduces an upper bound to the abstraction overhead of an optimal schedulable periodic interface. We define the *overhead bound* $O^*_{\mathcal{P}}(C)$ of component $C(W, A)$ as a number such that there exists a schedulable periodic interface $\mathcal{P}$ of C such that $O_{\mathcal{P}} \leq O^*_{\mathcal{P}}(C)$.

According to Corollary 6.1 and 6.2, a periodic interface $U_{\mathcal{P}}$ is schedulable if $U_{\mathcal{P}} \geq \mathsf{AB}_{W,A}$. That is, the minimum interface utilization $U_{\mathcal{P}}$ of a schedulable periodic interface $\mathcal{P}$ is upper bounded by $\mathsf{AB}_{W,A}$. We can then compute the overhead bound $O^*_{\mathcal{P}}(C)$ of C as

$$O^*_{\mathcal{P}}(C) = \frac{\mathsf{AB}_{W,A} - U_W}{U_W}.$$

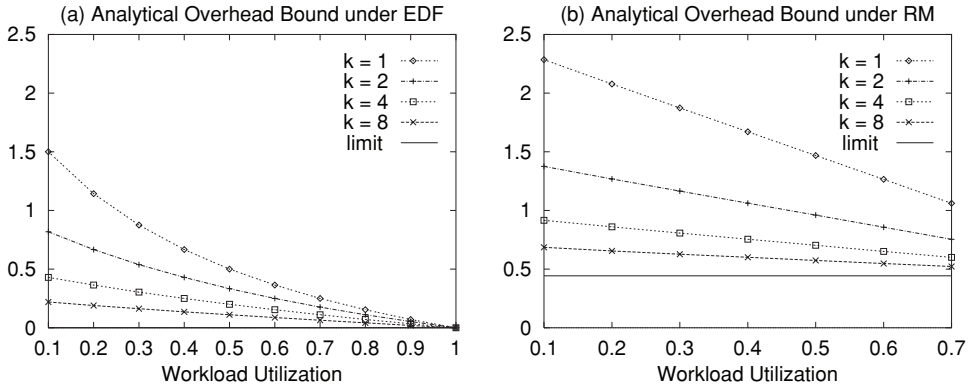

Fig. 9. Component abstraction overhead bound as a function of workload utilization: (a) under EDF scheduling and (b) under RM scheduling.

Under EDF scheduling, we can derive the overhead bound $O^*_{\mathcal{P}}(C)$ of component $C(W, A)$, where $W = \{T(p_i, e_i)\}$ and $A = \mathsf{EDF}$, from Corollary 6.1, i.e.,

$$O^*_{\mathcal{P}}(W, \mathsf{EDF}) = \frac{2(1 - U_W)}{k + 2 \cdot U_W}, \quad \text{where } k = K_{\mathsf{EDF}}(P_{min}, \mathcal{P}). \tag{28}$$

Under RM scheduling, we can derive the overhead bound $O^*_{\mathcal{P}}(C)$ of component $C(W, A)$, where $W = \{T(p_i, e_i)\}$ and $A = \mathsf{RM}$, from Corollary 6.2, i.e.,

$$O^*_{\mathcal{P}}(W, \mathsf{RM}) = \frac{1}{\log\left(\frac{2k+2(1-U_W)}{k+2(1-U_W)}\right)} - 1, \quad \text{where } k = K_{\mathsf{RM}}(P_{min}, \mathcal{P}). \tag{29}$$

Figure 9 shows the effect of period of a periodic interface, in terms of k in Equations (28) and (29), on the overhead bound as a function of workload utilization under EDF and RM scheduling. The solid line, labeled "limit," shows the limit of the bound, which is obtained when $k = \infty$. The limit is 0 under EDF scheduling and $1/log(2) - 1$, which is approximately 0.443, under RM scheduling. The other curves show the bounds when k is 1, 2, 4, and 8. It is shown that as k increases, the bounds converge to their limits.

7.2 Simulation Results

This section evaluates the component abstraction overhead through simulation results. During simulations, we have the following simulation parameters and value ranges:

—The number of tasks (n) in the workload set W is 2, 4, 8, 16, 32, or 64.
—The workload utilization (U_W) of the workload set W is in the interval [0.1, 0.7].
—Each periodic task $T(p, e)$ has a period p randomly generated in the range [5, 100] and an execution time requirement e generated in the range [1, 40].
—Scheduling algorithm (A) is EDF or RM.

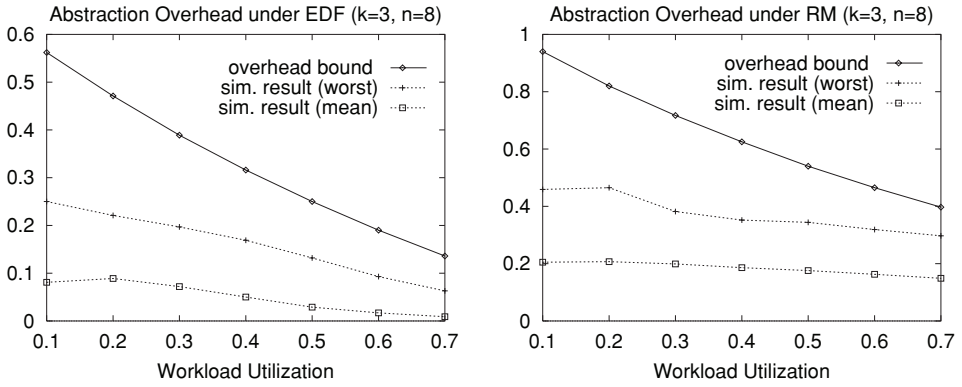

Fig. 10. Abstraction overheads as a function of workload utilization: (a) under EDF scheduling and (b) under RM scheduling.

—Period multiple relationship (k) between a workload set W and a periodic interface $\mathcal{P}(P, E)$ is 1, 2, 4, 8, 16, 32, or 64, where $k = K_A(P_{min}, \mathcal{P}(P, E))$ as defined in Equation (17) or (18) depending on A.

Our simulation procedure is as follows:

—For the ith simulation case, $i = 1, 2, \ldots, N_{sim}$, where N_{sim} is the number of total simulation cases, we do the following.

—Given n and U_W, we randomly generate a workload set W.

—We do the followings for a case where $A = \mathsf{EDF}$ and, for the other case, where for $A = \mathsf{RM}$, respectively.

—Given W and A, we do the following for all values of k, where $k = 1, 2, 4, 8, 16, 32, 64$: We find the periodic interface $\mathcal{P}(P^*, E^*)$ that is a schedulable interface of the component $C(W, A)$ with the minimum resource capacity requirement, such that $k = K_A(P_{min}, \mathcal{P}(P^*, E^*))$. We find such E^* by evaluating Equation (8) or (10), depending on A.

—We use $O_{W,A}(k, i)$ to denote the minimum abstraction overhead for k in the ith simulation case.

—As performance metrics, let $O_{W,A}(k, \mathsf{mean})$ denote the mean of $O_{W,A}(k, i)$ for all $i \in [1, N_{sim}]$, and let $O_{W,A}(k, \mathsf{worst})$ denote the worst-case (maximum) value of $O_{W,A}(k, i)$ for all $i \in [1, N_{sim}]$.

In Figures 10, 11, and 12, component abstraction overheads are plotted under EDF and RM scheduling as a function of workload utilization, interface period, and the number of tasks, respectively. In the graphs, the solid curve, labeled "overhead bound," shows the overhead bound, which is obtained through Equation (28) or (29), depending on scheduling algorithm. A dotted curve, labeled "simulation result (mean)," shows the mean of component-abstraction overheads, $O_{W,A}(k, \mathsf{mean})$, which are obtained through simulations. Each point in this curve represents the mean of 1000 simulation results unless specified otherwise. The 95% confidence intervals for data range from ± 0.01 to ± 0.13 of the means shown in the curve.

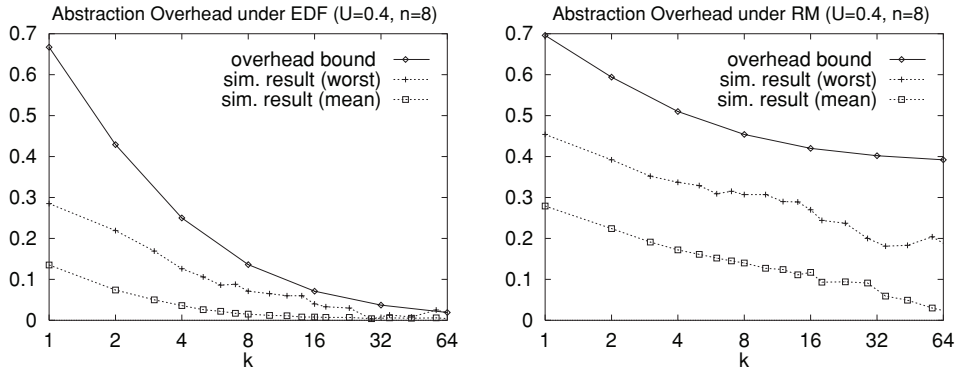

Fig. 11. Abstraction overheads as a function of interface period: (a) under EDF scheduling and (b) under RM scheduling.

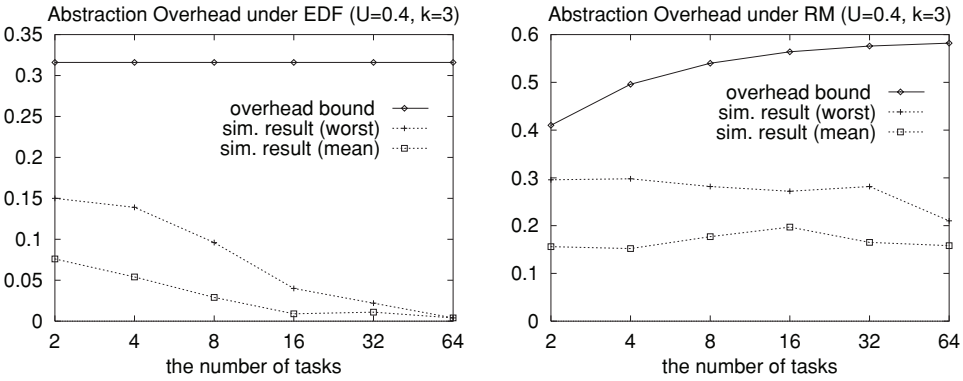

Fig. 12. Abstraction overheads as a function of the number of tasks: (a) under EDF scheduling and (b) under RM scheduling.

The other dotted curve, labeled "simulation result (worst)," shows the worst-case component-abstraction overhead, $O_{W,A}(k, \text{worst})$, during simulations.

Figure 10 shows the effect of workload utilization on the abstraction overheads under EDF and RM scheduling, where $k = 3$ and $n = 8$. It is shown that the abstraction overhead generally decreases as the workload utilization increases. The figure also shows that the abstraction overhead is clearly lower under EDF scheduling than under RM scheduling, which is consistently shown in Figures 11 and 12.

Figure 11 shows the effect of interface period, in terms of k, on the abstraction overheads under EDF and RM scheduling, where $U_W = 0.4$ and $n = 8$. It is shown that the abstraction overhead decreases as k increases.

Figure 12 shows the effect of the number of tasks on the abstraction overheads under EDF and RM scheduling, where $U_W = 0.4$ and $k = 3$. It is shown that the abstraction overhead decreases as the number of tasks increases under EDF scheduling. Under RM scheduling, however, the abstraction overhead is relatively independent of the number of tasks.

The implications of our simulation results can be summarized as follows: the abstraction overhead of a periodic interface is smaller under EDF scheduling than under RM scheduling. Interface period is a factor that most significantly affects the abstraction overhead of periodic interface. Under EDF scheduling, both the workload utilization and the number of tasks slightly affect the abstraction overhead. Under RM scheduling, however, they have little impact on the abstraction overhead. In summary, our approach incurs less abstraction overheads when we have a smaller interface period under EDF scheduling.

8. DISCUSSION

So far, we have addressed the component-abstraction problem under the assumptions that (1) the task model is the pure periodic task model and (2) tasks are independent. This section discusses the issues of extending our framework in relaxing these assumptions.

8.1 Supporting Aperiodic Tasks

In addition to the periodic tasks that execute repeatedly at regular time intervals, real-time systems may consist of tasks whose release times are not known a priori, since they are to respond to asynchronous external events. We consider two task models, *sporadic* and *aperiodic* task models, to characterize the workload generated in response to these events. In this section, we mainly distinguish them by the property of their deadlines: hard or soft.[3] Sporadic tasks are released with hard deadlines at random time instants with the minimum interarrival separation time between consecutive jobs. Aperiodic tasks can be released with soft deadlines at any arbitrary time instant without any restriction. One approach for handling sporadic and aperiodic tasks is to reject some of their jobs if they cannot complete in time. The other approach is to accept all sporadic and aperiodic tasks and to allow all sporadic tasks to complete prior to their hard deadlines and some aperiodic tasks to complete later than their soft deadlines. Here, we discuss how to extend our framework for supporting the sporadic and aperiodic task models with the latter approach.

A sporadic task τ_i can be defined by a triple (e_i, d_i, s_i), where e_i is a worst-case execution time requirement, d_i is a relative deadline, and s_i is a minimum interarrival separation between any two jobs of τ_i. Baruah et al. [1990b] have shown that the cumulative resource demands of jobs of τ_i over an interval $[t_0, t_0 + t)$ is maximized if the first arrives at the start of the interval (i.e., at time instant t_0) and subsequent jobs arrive as rapidly as permitted (i.e., at instants $t_0 + k \cdot s_i, k = 1, 2, 3, \ldots$). They presented the demand-bound function that computes the total resource demand of a sporadic task set W under EDF scheduling. This demand-bound function $\mathtt{dbf}^*_{\mathsf{EDF}}(W, t)$ calculates the maximum possible resource demands of W for every interval length t as follows:

$$\mathtt{dbf}^*_{\mathsf{EDF}}(W, t) = \sum_{\tau_i \in W} \max\left(0, \left(\left\lfloor \frac{t - d_i}{p_i} + 1 \right\rfloor \times e_i\right)\right). \tag{30}$$

[3]We here treat a sporadic task with a soft deadline as an aperiodic task.

This approach is pessimistic, but can guarantee the schedulability of sporadic tasks at design time. Some dynamic resource reclaiming approaches [Lehoczky and Ramos-Thuel 1992] can be used to complement the pessimism of this approach, dynamically reclaiming unused resources reserved for sporadic tasks when the sporadic tasks are released with longer interarrival separation than their minimum separation.

For an aperiodic task, we do not make any assumption on its interarrival separation and execution time. A desirable property in supporting aperiodic tasks is to minimize their response times without violating the real-time guarantees of periodic and sporadic tasks. One approach for this is to make the execution of aperiodic tasks interrupt-driven according to the *bandwidth-preserving server* [Lehoczky et al. 1987; Sprunt et al. 1989; Strosnider et al. 1995; Spuri and Buttazzo 1994; Abeni and Buttazzo 1998]. When an aperiodic task arrives, the execution of periodic or sporadic tasks are interrupted, and the aperiodic job is executed as long as its bandwidth-preserving server has a capacity enough to execute it. Our issue here is to discuss if and how such a bandwidth-preserving server can be used to support aperiodic tasks in a hierarchical scheduling framework.

For simplicity, we consider a *deferrable server* (p_s,c_s) [Strosnider et al. 1995], which is the simplest of bandwidth-preserving servers, that emulates a periodic task with a period p_s and an execution budget e_s. Whenever there is an aperiodic task to execute, the deferrable server is scheduled and it executes the aperiodic task until its budget is expired; it consumes an execution budget whenever executing an aperiodic task. Its budget is replenished to c_s every period p_s. Given that the deferrable server (p_s,e_s) has the highest priority, the demand-bound function of a periodic task T_i is defined under fixed-priority scheduling as follows [Strosnider et al. 1995]:

$$\mathtt{dbf}^{+}_{\mathsf{RM}}(W,t,i) = e_i + e_s + \left\lceil \frac{t-e_s}{p_s} \right\rceil e_s + \sum_{k=1}^{i-1} \left\lceil \frac{t}{p_k} \right\rceil e_k. \tag{31}$$

Our framework can then be extended with aperiodic tasks through the use of the deferrable server, by plugging in the demand bound function $\mathtt{dbf}^{+}_{\mathsf{RM}}(W,t,i)$ into Theorem 4.2.

8.2 Supporting Interacting Tasks with Data Dependency

Until now, we have addressed the component-abstraction problem under the assumption that the tasks of a component are independent, i.e., they can execute in any order. In many real-time systems, data and control dependencies among tasks may constrain the order in which they can execute. For example, in a radar surveillance system, the signal-processing task is the producer of track records, while the tracker task is the consumer. There are two approaches for supporting data dependencies between the producer and consumer tasks.

One approach is to place the precedence constraint between the producer and consumer tasks so that a consumer job synchronize with the corresponding producer job(s) and wait until the latter completes in order to execute.

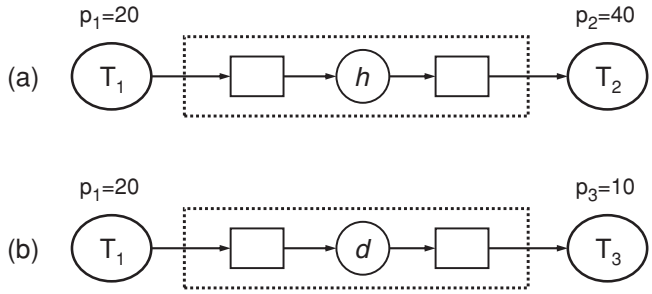

Fig. 13. Data producer and consumer tasks in the RTW model: (a) with a hold block and (b) with a delay block.

The other approach is not to synchronize producer and consumer tasks. Rather, each producer places the data generated by it in a shared address space to be used by the consumer at any time. In this case, the producer and consumer are independent, because they are not explicitly constrained to execute in turn. In this approach, a problem of data integrity can happen between data producer and consumer tasks running at different rates. The data integrity problem exists when the input to a data consumer task changes during the execution of that task. For example, a faster producer task supplies the input to a slower consumer task. The slower consumer reads an input value v_1 from the faster producer and begins computations using that value. The computations are preempted by another execution of the faster producer, which computes a new output value v_2. A data integrity problem now arises: when the slower consumer resumes execution, it continues its computations, now using the "new" input value v_2.

Here, we briefly examine an approach, namely, the *real-time workshop (RTW)* approach, that can resolve the data integrity problem in real-time data transfer between the producer and consumer tasks with different periods.

RTW is a tool for automatic code generation that can be used in the MATLAB/Simulink environment [Mathworks 2005]. The tool addresses the data integrity problem by placing *race transition* blocks, *hold* or *delay* blocks, between the data producer and consumer tasks of different periods, under the restriction that the periods of producer and consumer tasks should be harmonic. The tool assumes that a priority-based preemption mechanism will be used for task execution. Figure 13 shows an example of adding hold and delay blocks.

A hold block is inserted between a faster producer task T_1 and a slower consumer task T_2, for guaranteeing that the input value of T_2 from T_1 does not change during the execution of T_2. The hold block has the same period of T_2 but is a higher-priority task than T_2, so that it reads the latest output value of T_1 before T_2 executes and holds it until T_2 finishes executing.

A delay block is used for the inverse case, being inserted between a slower producer task T_1 and a faster consumer task T_3. The delay block has the same period of T_1 but is a higher-priority task than T_1, for ensuring that no matter when T_1 finishes, the delay block does not overwrite the input value of T_3 during the execution of T_3.

Resolving the data-integrity problem between the producer and consumer tasks using the hold and delay blocks, the RTW approach inherently allow these producer and consumer tasks to be treated independently. That is, even though a component C consists of interacting tasks with data dependency, the component C can be treated as consisting of only independent tasks. Assuming the negligible execution times of the hold and delay blocks, Matic and Henzinger [2005] showed that our proposed framework can be used for abstracting the component C. In addition to the RTW approach, they also showed that another approach, namely, the logical execution time (LET) approach [Henzinger et al. 2003], can be used for supporting interacting tasks with data dependencies and that our proposed framework can be used with LET for abstracting components with the interacting tasks.

8.3 Supporting Mutually Exclusive Resource Sharing

In many practical applications, there is a need for synchronization in the mutually exclusive access to shared resources. We here discuss how to support local (i.e., within-component) and global (i.e., between-components) resource sharing in the proposed compositional scheduling framework. In this section, for simplicity, we only consider fixed-priority preemptive scheduling within each component and a task is assumed to access one local and one global shared resources. A task T_i is characterized by (p_i, e_i, x_i^L, x_i^G), where x_i^L and x_i^G are the worst-case execution times of T_i within local and global critical sections (WCET-CS), respectively, and e_i includes x_i^L and x_i^G.

8.3.1 *Supporting Local Resource Sharing.* We consider an issue of supporting local resource sharing in a hierarchical scheduling framework. When a task tries to enter a critical section (for an access to a mutually exclusive resource), it can be blocked if there is another task inside the critical section at that time. Let B_i denote the maximum blocking time of a task T_i that T_i can experience in trying to enter a critical section.

The resource (processor) demand of a task T_i should then be extended with blocking B_i as follows:

$$\mathtt{dbf}_{\mathsf{FP}}^{S}(t, i) = \sum_{k=1}^{i} \left\lceil \frac{t}{p_k} \right\rceil e_k + B_i. \tag{32}$$

Several protocols [Sha et al. 1987; Rajkumar et al. 1988] were proposed to bound the maximum blocking time B_i of a task T_i under fixed-priority scheduling. For example, under priority ceiling protocol (PCP) [Rajkumar et al. 1988], a task can be blocked by, at most, one of its lower-priority tasks. Therefore, B_i is defined as the maximum WCET-CS of lower-priority tasks that can block T_i under PCP.

It has been shown in Almeida and Pedreiras [2004] that traditional synchronization protocols, such as priority inheritance protocol (PIP) [Sha et al. 1987] and priority ceiling protocol [Rajkumar et al. 1988], can be employed in a hierarchical scheduling framework to support local resource sharing, when Equation (32) is used as the demand-bound function of each task T_i; the

computation of B_i depends on which synchronization protocol is used. This is mainly because that all properties of any synchronization protocol established in a dedicated resource (processor) remains the same when applied to local resource sharing in a hierarchical scheduling framework. That is, the blocking and critical section entrance/leave within a component does not get affected by the other components. The only difference is that the blocking time and the duration of a critical section are inflated in a hierarchical scheduling framework, unlike over the dedicated resource (processor). Since we can take care of this time inflation with the supply bound function $\mathtt{sbf}_\Gamma(t)$ of a periodic resource model Γ, independent of B_i and $\mathtt{dbf}^S_{\mathtt{FP}}(t, i)$, our framework can be extended to support local resource sharing under the PIP or PCP protocol.

8.3.2 *Supporting Global Resource Sharing.* We now explain how to support global resource sharing in a hierarchical scheduling framework. For simplicity, we assume that a single resource is accessed by a single task per component and there is no priority between components. Furthermore, we also assume that each task accesses, at most, one resource. We first present a basic global synchronization protocol under these assumptions and later discuss how to relax these assumptions. For the basic protocol, we assume that each global shared resource is guarded by a semaphore and the semaphore has a queue. Whenever a task tries to access the shared resource, the task is placed into the corresponding semaphore queue. For simplicity, our protocol assumes that the semaphore queue is managed by the FCFS (first-come-first-served) manner. We now consider schedulability analysis for this protocol.

Once the task T_i enters a global critical section, the task will be interfered by only its higher-priority tasks within the same component. We can then compute the maximum possible interference of higher-priority tasks to the task T_i within a critical section during an interval of length t as follows:

$$\mathtt{dbf\text{-}X}_{\mathtt{FP}}(t, i) = x_i^G + \sum_{k=1}^{i-1} \left\lceil \frac{t}{p_k} \right\rceil e_k, \tag{33}$$

As an approach to reduce $\mathtt{dbf\text{-}X}_{\mathtt{FP}}(t, i)$, we can assign the highest priority to the task T_i that accesses a global shared resource. Then, it is simply $\mathtt{dbf\text{-}X}_{\mathtt{FP}}(t, i) = x_i$.

We can compute the worst-case response time $rx_i(\Gamma)$ of a task T_i within a critical section over a periodic resource model Γ as follows:

$$rx_i(\Gamma) = \min\{t \mid \mathtt{dbf\text{-}X}_{\mathtt{FP}}(t, i) \le \mathtt{sbf}_\Gamma(t)\}.$$

When a task T_i tries to enter a global critical section, it can be placed into a corresponding semaphore queue. Let $Q(T_i)$ denote a set of tasks for which the task T_i should wait in the worst case in order to enter the critical section. We can compute the maximum blocking time B_i^G of T_i for a global critical section as follows:

$$B_i^G = \sum_{T_k \in Q(T_i)} rx_k(\Gamma_k), \tag{34}$$

where Γ_k denotes the periodic resource model associated with the component to which a task T_k belongs. Under the assumption that there is only one task within a component accessing a global resource, the task T_i can be blocked by $m-1$ tasks across components, where there are m components that would access the same global resource. We note that if this assumption is relaxed, the task T_i can be blocked by $n-1$ tasks, when there are n tasks accessing the same global resource across components.

Now, a task T_i that accesses a global shared resource can be viewed as an independent task to the other tasks within the same component, with the maximum blocking time of B_i^G. Therefore, we can compute the worst-case response time of T_i as follows:

$$r_i^G(\Gamma) = \min\{t \mid \ \mathtt{dbf}_{\mathsf{FP}}(t + B_i^G, i) \leq \mathtt{sbf}_{\Gamma}(t)\},$$

where the demand-bound function $\mathtt{dbf}_{\mathsf{FP}}(t, i)$ of T_i is given in Equation (9).

We can now see that all components subject to global resource sharing are schedulable, if, for each component $C(W, A = \mathsf{FP})$,

$$\forall T_i \in W \quad r_i^G \leq p_i. \tag{35}$$

In this section, we have described how our framework might be extended to support global resource sharing with the basic global synchronization protocol. We note that this protocol requires many extensions with relaxed assumptions to be practically useful. For example, when a task is allowed to access multiple global shared resources, it should prevent deadlocks involving global shared resources. Furthermore, when multiple tasks within a component are allowed to access the same global shared resource, it should handle the priority inversion problem between those tasks with the same component. When components have priorities, it should also handle the priority inversion problem between components. These issues, however, are beyond the scope of this paper as they require to extend the basic protocol, and we leave them as an interesting future work.

9. CONCLUSION

In this article, we defined the proposed compositional real-time scheduling framework. Problems need to be addressed and our approaches to the problems presented using the periodic interface model. We proposed the periodic resource model as a conceptual basis for the periodic interface. With the periodic resource model, we defined the scheduling unit and then developed its exact schedulability conditions and its schedulable workload utilization bound under EDF and RM scheduling. With the periodic interface model, we addressed the component-abstraction and composition problems and derived the schedulable component-abstraction bounds under EDF and RM scheduling. We also evaluated the overhead that the periodic interface incurs in terms of utilization increase through analytical bounds and simulation results.

In this article, we considered a compositional scheduling framework for component-based hard real-time systems. An interesting future work is to extend the framework for component-based soft real-time systems. This raises

the issues of developing soft real-time interface models. We believe that soft real-time task models, such as the (m, k)-firm deadline model [Hamdaoui and Ramanathan 1995] and the weakly hard task model [Bernat et al. 2001] can be used to develop the real-time interface model of a compositional soft real-time scheduling framework.

APPENDIX

In Section 5, we presented Theorem 5.2 to introduce a utilization bound for a scheduling unit $S(W, R, A)$ under RM scheduling. Here, we present the proof of Theorem 5.2.

We consider a scheduling unit $S(W, R, A)$ such that $W = \{T_i = T(p_i, e_i)\}$, $R = \Gamma(\Pi, \Theta)$, and $A = \mathsf{RM}$. A workload set W is said to *fully utilize* a resource model R if a scheduling unit S is schedulable, but S becomes no longer schedulable if the execution time e_i of any task T_i is increased. If a scheduling unit S is schedulable, the workload utilization U_W of a workload set W is said to be a *schedulable workload utilization*. The least upper bound to the schedulable workload utilization is said to be a utilization bound.

Our proof is organized as follows: We first consider a task period restriction that the largest ratio between task periods is less than 2. Lemma 9.1 and 9.2 show the properties of a workload set W, when its workload utilization U_W is the least upper bound to the schedulable workload utilization. Using the results of the two lemmas, Theorem 9.1 derives a utilization bound and Theorem 9.2 removes the task period restriction.

We first start our discussion under the assumption that the ratio between any two task period is less than 2. This assumption is later removed in Theorem 9.2. We present the following lemma to introduce a property of a workload set W in terms of the execution time requirement e_i of each task T_i, when the workload utilization U_W is the the least upper bound to the schedulable workload utilization.

LEMMA 9.1. *For scheduling unit $S(W, R, A)$, where $W = \{T_i = T(p_i, e_i)\}$, $R = \Gamma(\Pi, \Theta)$, and $A = \mathsf{RM}$, under the restriction that the ratio between any two task periods of W is less than 2, if the workload set W fully utilizes the resource Γ under RM scheduling with the smallest possible workload utilization, then it follows that*

$$\sum_{T_i \in W} e_i = \mathtt{sbf}_\Gamma(P_{min}),$$

where P_{min} is the smallest task period of the workload set W.

PROOF. We mainly adapt the proof of Liu and Layland [1973] to prove this lemma. For the workload set W, we assume that $p_1 < p_2 < \cdots < p_{n-1} < p_n$. We now assume that the workload set W fully utilizes the resource Γ under RM scheduling with the smallest possible workload utilization U^*_W. Let $e^*_1, e^*_2, \ldots, e^*_n$ be the execution times of the tasks $T_1, T_2, \ldots, T_n$ that determine U^*_W. Then, we first need to show that

$$e^*_1 = \mathtt{sbf}_\Gamma(p_1, p_2),$$

where we define $\mathtt{sbf}_\Gamma(t_1, t_2)$ as $\mathtt{sbf}_\Gamma(t_2) - \mathtt{sbf}_\Gamma(t_1)$ for notational simplicity. We now show this by contradiction.

Suppose that

$$e_1^* = \mathtt{sbf}_\Gamma(p_1, p_2) + \Delta, \quad \Delta > 0. \tag{36}$$

Let

$$e_1' = \mathtt{sbf}_\Gamma(p_1, p_2), \quad e_2' = e_2^* + \Delta, \quad e_3' = e_3^*, \quad \ldots \quad e_n' = e_n^*.$$

Given that $e_1^*, e_2^*, \ldots, e_n^*$ guarantee the schedulability of the scheduling unit S and that any increase in e_i^* will make S unschedulable, it is clear that a workload set with $e_1', e_2', \ldots, e_n'$ is schedulable over Γ and that any increase in e_i' will violate the schedulability of the task set over Γ. Let U_W' denote the corresponding utilization. We have

$$U_W^* - U_W' = (\Delta/p_1) - (\Delta/p_2) > 0.$$

Hence, this assumption given in Eq. (36) is false, when $\Delta > 0$.

Alternatively, suppose that

$$e_1^* = \mathtt{sbf}_\Gamma(p_1, p_2) - \Delta, \quad \Delta > 0. \tag{37}$$

Let

$$e_1'' = \mathtt{sbf}_\Gamma(p_1, p_2), \quad e_2'' = e_2^* - 2\Delta, \quad e_3'' = e_3^*, \quad \ldots \quad e_n'' = e_n^*.$$

Again, a workload set with $e_1'', e_2'', \cdots, e_{n-1}'', e_n''$ is also schedulable over Γ and any increase in e_i'' will violate the schedulibility of the task set. Let U'' denote the corresponding utilization. We have

$$U_W^* - U_W'' = -(\Delta/p_1) + (2\Delta/p_2) > 0.$$

Again, this assumption given in Equation (37) is also false, when $\Delta > 0$.

Therefore, if indeed U_W^* is the least upper bound of the workload utilization, then

$$e_1^* = \mathtt{sbf}_\Gamma(p_1, p_2).$$

In a similar way, we can show that

$$e_2^* = \mathtt{sbf}_\Gamma(p_2, p_3), \quad e_3^* = \mathtt{sbf}_\Gamma(p_3, p_4), \quad \ldots \quad e_{n-1}^* = \mathtt{sbf}_\Gamma(p_{n-1}, p_n).$$

Consequently,

$$\begin{aligned} e_n^* &= \mathtt{sbf}_\Gamma(0, p_n) - 2(e_1^* + e_2^* + \cdots + e_{n-1}^*) \\ &= \mathtt{sbf}_\Gamma(0, p_n) - 2\mathtt{sbf}_\Gamma(p_1, p_n) \\ &= \mathtt{sbf}_\Gamma(0, p_1) - \mathtt{sbf}_\Gamma(p_1, p_n) \end{aligned}$$

Finally, we have

$$\sum_{T_i \in W} e_i^* = \mathtt{sbf}_\Gamma(p_1).$$

□

We present the following lemma to show another property of a workload set W in terms of P_{min}, where P_{min} is the smallest task period in W, when

W fully utilizes the resource $\Gamma(\Pi, \Theta)$ with the smallest workload utilization. More specifically, the following lemma shows that the least upper bound to the schedulable workload utilization becomes smallest when $P_{min} = (k+2)\Pi - 2\Theta$ for all $P_{min} \in [(k+1)\Pi - \Theta, (k+2)\Pi - \Theta)$, where $k = 1, 2, 3, \cdots, .$

LEMMA 9.2. *Consider a scheduling unit $S(W, R, A)$, where $W = \{T_i = T(p_i, e_i)\}$, $R = \Gamma(\Pi, \Theta)$, and $A =$ RM. Let P_{min} denote the smallest task period of W, and it is assumed that $2(\Pi - \Theta) < P_{min}$. A range of P_{min} can be categorized as $P_{min} \in [(k+1)\Pi - \Theta, (k+2)\Pi - \Theta)$ with a positive integer $k = 1, 2, 3, \cdots, .$ If the workload set W fully utilizes the resource Γ under RM scheduling with the smallest possible workload utilization, U_W is minimized when $P_{min} = (k+2)\Pi - 2\Theta$ for all $P_{min} \in [(k+1)\Pi - \Theta, (k+2)\Pi - \Theta)$, where $k = 1, 2, 3, \cdots, .$*

PROOF. Suppose that the workload set W fully utilizes the resource Γ with the smallest possible workload utilization under RM scheduling. Then, according to Lemma 9.1, the following holds:

$$\sum_{T_i \in W} e_i = \mathtt{sbf}_\Gamma(P_{min}).$$

Let P_k^* denote $(k+2)\Pi - 2\Theta$. We consider two cases in terms of P_{min}: (1) $P_{min} \in \left[P_k^* - (\Pi - \Theta), P_k^*\right)$ and (2) $P_{min} \in \left(P_k^*, P_k^* + \Theta\right)$.

1. For the first case where $P_{min} \in \left[P_k^* - (\Pi - \Theta), P_k^*\right)$, it is clear that $\mathtt{sbf}_\Gamma(P_{min}) = k\Theta$, since there is no resource supply during the interval $\left[P_k^* - (\Pi - \Theta), P_k^*\right)$ at the worst-case resource supply. We transform $W = \{T(p_i, e_i)\}$ to $W' = \{T(p_i', e_i')\}$ such that

$$T(p_i', e_i') = \begin{cases} T(p_i, e_i) & \text{if } (p_i \geq P_k^*), \\ T\left(P_k^*, e_i\right) & \text{otherwise,} \end{cases}$$

   With this transformation, W' has the smallest task period of P_k^* and still fully utilizes the resource $\Gamma(\Pi, \Theta)$ while its workload utilization decreases, i.e., $U_{W'} < U_W$.

2. For the second case, where $P_{min} \in \left(P_k^*, P_k^* + \Theta\right)$, there is a resource supply of Θ during the interval $\left[P_k^*, P_k^* + \Theta\right)$ at the worst-case resource supply. Let us consider $P_{min} = P_k^* + \delta$, where $0 < \delta < \Theta$. In this case, $\mathtt{sbf}_\Gamma(P_{min}) = k\Theta + \delta$. We wish to show that

$$\mathtt{sbf}_\Gamma(P_k^*) < \frac{P_k^*}{P_{min}}\, \mathtt{sbf}_\Gamma(P_{min}). \tag{38}$$

   It follows that

$$\frac{\mathtt{sbf}_\Gamma(P_k^*)}{P_k^*} - \frac{\mathtt{sbf}_\Gamma(P_{min})}{P_{min}} = \frac{k\Theta}{P_k^*} - \frac{k\Theta + \delta}{P_k^* + \delta} < 0.$$

   We transform $W = \{T(p_i, e_i)\}$ to $W'' = \{T(p_i'', e_i'')\}$ such that

$$T(p_i'', e_i'') = T(q \cdot p_i, q \cdot e_i), \qquad \text{where } q = P_k^*/P_{min}.$$

With this transformation, W'' has the smallest task period of P_k^*, and $U_{W''} = U_W$. According to Lemma 9.1, if W'' fully utilizes the resource Γ with the smallest possible workload utilization under RM scheduling , it should satisfy

$$\sum_{T(p_i'',e_i'')\in W''} e_i'' = \mathtt{sbf}_\Gamma(P_k^*). \tag{39}$$

However, it follows

$$\sum_{T(p_i'',e_i'')\in W''} e_i'' \;=\; q \cdot \sum_{T(p_i,e_i)\in W} e_i \;=\; q \cdot \mathtt{sbf}_\Gamma(P_{min}) \;>\; \mathtt{sbf}_\Gamma(P_k^*).$$

Thus, we may need to decrease some e_i'' to make $S(W'', R, A)$ schedulable, and this consequently decreases the workload utilization $U_{W''}$, which leads to $U_{W''} < U_W$.

□

Using the properties shown in Lemma 9.1 and 9.2, we present the following theorem to introduce a utilization bound $\mathsf{UB}_{\Gamma,\mathsf{RM}}(n, P_{min})$ for a scheduling component $S(W, R, A)$, where $W = \{T_1(p_1, e_1), \ldots, T_n(p_n, e_n)\}$, $R = \Gamma(\Pi, \Theta)$, and $A = \mathsf{RM}$.

THEOREM 9.1. *Scheduling unit $S(W, R, A)$ is schedulable, where $W = \{T(p_1, e_1), \ldots, T(p_n, e_n)\}$, $R = \Gamma(\Pi, \Theta)$, and $A = \mathsf{RM}$, under the restrictions that the ratio between any two task periods of W is less than 2, if*

$$U_W \le U_\Gamma \cdot n\Big[\Big(\frac{(2k+2) - 2 \cdot U_\Gamma}{(k+2) - 2 \cdot U_\Gamma}\Big)^{1/n} - 1\Big],$$

where $k = K_{\mathsf{RM}}(P_{min}, \Gamma(\Pi, \Theta))$, defined in Equation (11), and P_{min} is the smallest task period in W.

PROOF. For a workload set $W = \{T_1 = T(p_1, e_1), \ldots, T_n = T(p_n, e_n)\}$, we assume that $p_n > p_{n-1} > \cdots > p_2 > p_1$. Let $e_1^*, e_2^*, \cdots, e_n^*$ be the execution times of the tasks $T_1, T_2, \cdots, T_n$ that determine the least upper bound of U_W subject to the schedulability guarantee of the scheduling unit S. Let U_W^* denote the least schedulable utilization bound for S. To achieve U_W^*, Lemma 9.1 shows that the execution times $e_1^*, e_2^*, \cdots, e_n^*$ should be determined as follows:

$$e_1^* = \mathtt{sbf}_\Gamma(p_1, p_2), \quad \ldots, \quad e_{n-1}^* = \mathtt{sbf}_\Gamma(p_{n-1}, p_n), \quad e_n^* = \mathtt{sbf}_\Gamma(0, p_1) - \mathtt{sbf}_\Gamma(p_1, p_n).$$

Then, we can derive U_W^* as follows:

$$\begin{aligned} U_W^* &= \frac{e_1^*}{p_1} + \cdots + \frac{e_{n-1}^*}{p_{n-1}} + \frac{e_n^*}{p_n} \\ &= \frac{\mathtt{sbf}_\Gamma(p_1, p_2)}{p_1} + \cdots + \frac{\mathtt{sbf}_\Gamma(p_{n-1}, p_n)}{p_{n-1}} + \frac{\mathtt{sbf}_\Gamma(0, p_1) - \mathtt{sbf}_\Gamma(p_1, p_n)}{p_n}. \end{aligned} \tag{40}$$

Now, we want to find the the minimum value of U_W^*. According to Lemma 9.2, the smallest task period p_1 should be P_k^*, where $P_k^* = (k+2)\Pi - 2\Theta$, to minimize U_W^*. Thus, we rewrite Equation (40) as follows:

$$\begin{aligned} U_W^* &= \frac{\mathtt{sbf}_\Gamma(P_k^*, p_2)}{P_k^*} + \cdots + \frac{\mathtt{sbf}_\Gamma(p_{n-1}, p_n)}{p_{n-1}} + \frac{\mathtt{sbf}_\Gamma(0, P_k^*) - \mathtt{sbf}_\Gamma(P_k^*, p_n)}{p_n} \\ &= \frac{\mathtt{sbf}_\Gamma(P_k^*, p_2)}{P_k^*} + \cdots + \frac{\mathtt{sbf}_\Gamma(p_{n-1}, p_n)}{p_{n-1}} + \frac{k\Theta - \mathtt{sbf}_\Gamma(P_k^*, p_n)}{p_n}. \end{aligned} \tag{41}$$

To find the minimum value of U_W^*, Equation (41) must be minimized over the p_i's. Since the supply bound function $\mathtt{sbf}_\Gamma(t_1, t_2)$ is a discrete function. However, it is difficult to obtain the minimum value of U_W^* through a numerical analysis. Thus, we replace $\mathtt{sbf}_\Gamma(t_1, t_2)$ with its linear lower-bound function $\mathtt{lsbf}_\Gamma(t_1, t_2)$ to obtain the minimum value of U_W^* through the numerical analysis. Now, we rewrite Equation (41) as follows:

$$\begin{aligned} U_W^* &= \frac{\mathtt{lsbf}_\Gamma(P_k^*, p_2)}{P_k^*} + \cdots + \frac{\mathtt{lsbf}_\Gamma(p_{n-1}, p_n)}{p_{n-1}} + \frac{k\Theta - \mathtt{lsbf}_\Gamma(P_k^*, p_n)}{p_n} \\ &= \frac{U_\Gamma(p_2 - P_k^*)}{P_k^*} + \cdots + \frac{U_\Gamma(p_n - p_{n-1})}{p_{n-1}} + \frac{k\Theta - U_\Gamma(p_n - P_k^*)}{p_n} \\ &= U_\Gamma\Big(\frac{p_2}{P_k^*} + \cdots + \frac{p_n}{p_{n-1}} + \frac{k\Pi + P_k^*}{p_n} - n\Big). \end{aligned} \tag{42}$$

We can now find the minimum value of U_W^* by minimizing Equation (40) over the p_i's. This can be done by setting the first derivative of U_W^*, with respect to each of the p_i's equal to zero and solving the resultant difference equations:

$$\partial U_W^* / \partial p_i = \frac{p_i^2 - p_{i-1} \cdot p_{i+1}}{p_{i-1} \cdot p_i^2} = 0, \quad i = 2, 3, \cdots, n. \tag{43}$$

The definition $p_{n+1} = (k\Pi + P_k^*)$ has been adopted for convenience.

The general solution to Equation (43) can be shown to be

$$p_i = ((k+2)\Pi - 2\Theta) \cdot \Big(\frac{(2k+2)\Pi - 2\Theta}{(k+2)\Pi - 2\Theta}\Big)^{(i-1)/n}, \quad i = 1, 2, \cdots, n. \tag{44}$$

It follows from Equations (42) and (44) that

$$\begin{aligned} U_W^* &= U_\Gamma \cdot n\Big[\Big(\frac{(2k+2)\Pi - 2\Theta}{(k+2)\Pi - 2\Theta}\Big)^{1/n} - 1\Big] \\ &= U_\Gamma \cdot n\Big[\Big(\frac{(2k+2) - 2 \cdot U_\Gamma}{(k+2) - 2 \cdot U_\Gamma}\Big)^{1/n} - 1\Big]. \end{aligned}$$

This completes the proof. □

The restriction that the largest ratio between task periods less than is 2 in Theorem 9.1 can actually be removed; we now present Theorem 9.2[4] as follows.

[4]Theorem 9.2 is identical to Theorem 5.2.

THEOREM 9.2. *Scheduling unit* $S(W, R, A)$ *is schedulable, where* $W = \{T(p_1, e_1), \ldots, T(p_n, e_n)\}$, $R = \Gamma(\Pi, \Theta)$, *and* $A = \mathsf{RM}$, *if*

$$U_W \leq U_\Gamma \cdot n\Big[\Big(\frac{(2k+2) - 2U_\Gamma}{(k+2) - 2U_\Gamma}\Big)^{1/n} - 1\Big],$$

where $k = K_{\mathsf{RM}}(P_{min}, \Gamma(\Pi, \Theta))$, *defined in Equation (11), and* P_{min} *is the smallest task period in* W.

PROOF. Consider a task set $W = \{T_1 = T(p_1, e_1), \cdots, T_n = T(p_n, e_n)\}$. Without loss of generality, we assume that $p_1 \leq p_2 \leq \cdots \leq p_n$. Assume that $e_1, e_2, \cdots, e_n$ guarantees the schedulability of W over Γ and that any increase in e_i for any task $T_i \in W$ will violate the schedulability of W over Γ. Let U_W denote the corresponding utilization.

Suppose that for some task $T_k \in W$, $\lfloor p_n/p_k \rfloor > 1$. To be specific, let $p_n = q \cdot p_k + r, q > 1$ and $0 \leq r < p_k$. Let us replace the task T_k by a task T'_k, such that $p'_k = q \cdot p_k$ and $e'_k = e_k$, and increase e_n by the amount needed to maximize the utilization subject to the schedulability guarantee over Γ, according to Theorem 4.2. This increase is, at most, $e_k(q-1)$, the time within the critical window of T_n occupied by T_k, but not by T'_k. Let $U_{W'}$ denote the utilization factor of such a set of tasks. We have

$$\begin{aligned} U_{W'} &= U_W + \frac{e_k(q-1)}{p_n} + \frac{e_k}{p'_k} - \frac{e_k}{p_k} \\ &= U_W + \frac{e_k(q-1)}{q \cdot p_k + r} + \Big(\frac{e_k}{q \cdot p_k} - \frac{q \cdot e_k}{q \cdot p_k}\Big) \\ &= U_W + e_k(q-1)\Big(\frac{1}{qp_k + r} - \frac{1}{qp_k}\Big). \end{aligned}$$

Since $q - 1 > 0$ and $[1/(qp_k + r)] - (1/q \cdot p_k) \leq 0, U_{W'} < U_W$. Therefore, we conclude that in determining the least upper bound of the processor utilization, we need only consider task sets in which the ratio between any two request periods is less than 2. The theorem thus follows directly from Theorem 9.1. □

ACKNOWLEDGMENTS

We thank Al Mok for introducing the problem to us. We also thank to Luis Almeida, Xiang "Alex" Feng, Slobodan Matik, and John Regehr for helpful discussions. We are grateful to the referees for many valuable comments that were essential in improving this article.

REFERENCES

ABENI, L. AND BUTTAZZO, G. 1998. Integrating multimedia applications in hard real-time systems. In *RTSS '98: Proceedings of the 19th IEEE Real-Time Systems Symposium*. IEEE Computer Society, Los Alamitos, CA. 4–13.

ALMEIDA, L. AND PEDREIRAS, P. 2004. Scheduling within temporal partitions: response-time analysis and server design. In *EMSOFT '04: Proceedings of the 4th ACM International Conference on Embedded Software*. ACM Press, New York. 95–103.

BARUAH, S., HOWELL, R., AND ROSIER, L. 1990a. Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor. *J. Real-Time Syst. 2*, 301–324.

BARUAH, S., MOK, A., AND ROSIER, L. 1990b. Preemptively scheduling hard-real-time sporadic tasks on one processor. In *RTSS '90: Proceedings of the 11th IEEE Real-Time Systems Symposium*. 182–190.

BERNAT, G., BURNS, A., AND LLAMOSI, A. 2001. Weakly hard real-time systems. *IEEE Trans. Comput. 50,* 4, 308–321.

DENG, Z. AND LIU, J. W.-S. 1997. Scheduling real-time applications in an open environment. In *RTSS '97: Proceedings of the 18th IEEE Real-Time Systems Symposium*. 308–319.

FENG, X. 2004. Design of real-time virtual resource architecture for large-scale embedded systems. Ph.D. thesis, University of Texas, Austin, TX.

FENG, X. A. AND MOK, A. K. 2002. A model of hierarchical real-time virtual resources. In *RTSS '02: Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS'02)*. IEEE Computer Society, Los Alamitos, CA. 26–35.

GOYAL, P., GUO, X., AND VIN, H. M. 1996. A hierarchical CPU scheduler for multimedia operating systems. In *Usenix Association Second Symposium on Operating Systems Design and Implementation (OSDI)*. 107–121.

HAMDAOUI, M. AND RAMANATHAN, P. 1995. A dynamic priority assignment technique for streams with (m, k)-firm deadlines. *IEEE Trans. Comput. 44,* 12, 1443–1451.

HENZINGER, T. A., HOROWITZ, B., AND KIRSCH, C. M. 2003. Giotto: A time-triggered language for embedded programming. *Proc. IEEE 91*, 84–99.

KUO, T.-W. AND LI, C.-H. 1999. A fixed-priority-driven open environment for real-time applications. In *RTSS '99: Proceedings of the 20th IEEE Real-Time Systems Symposium*. IEEE Computer Society, Los Alamitos, CA. 256–267.

LEHOCZKY, J. AND RAMOS-THUEL, S. 1992. An optimal algorithm for scheduling soft-aperiodic tasks in fixed-priority preemptive systems. In *Proceedings of IEEE Real-Time Systems Symposium*. 110–123.

LEHOCZKY, J. P., SHA, L., AND STROSNIDER, J. K. 1987. Enhanced aperiodic responsiveness in hard real-time environments. In *RTSS '87: Proceedings of* 8^{th} *IEEE Real-Time Systems Symposium*. IEEE Computer Society, Los Alamitos, CA. 261–270.

LEHOCZKY, J., SHA, L., AND DING, Y. 1989. The rate monotonic scheduling algorithm: exact characterization and average case behavior. In *RTSS '89: Proceedings of the 10th IEEE Real-Time Systems Symposium*. 166–171.

LIPARI, G. AND BARUAH, S. 2000. Efficient scheduling of real-time multi-task applications in dynamic systems. In *Proceedings of the IEEE*. 166–175.

LIPARI, G. AND BINI, E. 2003. Resource partitioning among real-time applications. In *ECRTS '03: Proceedings of 15th Euromicro Conference on Real-Time Systems*. 151–158.

LIPARI, G., CARPENTER, J., AND BARUAH, S. 2000. A framework for achieving inter-application isolation in multiprogrammed hard-real-time environments. In *RTSS '00: Proceedings of the 21st IEEE Real-Time Systems Symposium*.

LIU, C. AND LAYLAND, J. 1973. Scheduling algorithms for multi-programming in a hard-real-time environment. *J. ACM 20,* 1, 46–61.

MATHWORKS. 2005. Models with multiple sample rates. In *Real-Time Workshop User Guide*. 1–34.

MATIC, S. AND HENZINGER, T. A. 2005. Trading end-to-end latency for composability. In *RTSS '05: Proceedings of the 26th IEEE Real-Time Systems Symposium*. 99–110.

MOK, A., FENG, X., AND CHEN, D. 2001. Resource partition for real-time systems. In *RTAS '01: Proceedings of the 7th IEEE Real-Time Technology and Applications Symposium*. 75–84.

RAJKUMAR, R., SHA, L., AND LEHOCZKY, J. P. 1988. Real-time synchronization protocols for multiprocessors. In *RTSS '88: Proceedings of the 9th IEEE Real-Time Systems Symposium*. IEEE Computer Society, Los Alamitos, CA. 259–269.

REGEHR, J. AND STANKOVIC, J. 2001. HLS: A framework for composing soft real-time schedulers. In *RTSS '01: Proceedings of the 22nd IEEE Real-Time Systems Symposium*. 3–14.

SAEWONG, S., RAJKUMAR, R. R., LEHOCZKY, J. P., AND KLEIN, M. H. 2002. Analysis of hierarhical fixed-priority scheduling. In *ECRTS '02: Proceedings of the 14th Euromicro Conference on Real-Time Systems*. IEEE Computer Society, Los Alamitos, CA. 173–181.

SHA, L., LEHOCZKY, J. P., AND RAJKUMAR, R. 1987. Task scheduling in distributed real-time systems. In *Proceedings of the International Conference on Industrial Electronics, Control, and Instrumentation*. Cambridge, MA. 909–916.

SHIN, I. AND LEE, I. 2003. Periodic resource model for compositional real-time guarantees. In *RTSS '03: Proceedings of the 24th IEEE Real-Time Systems Symposium*. IEEE Computer Society, Los Alamitos, CA. 2–13.

SHIN, I. AND LEE, I. 2004. Compositional real-time scheduling framework. In *RTSS '04: Proceedings of the 25th IEEE Real-Time Systems Symposium*. IEEE Computer Society, Los Alamitos, CA. 57–67.

SPRUNT, B., SHA, L., AND LEHOCZKY, J. P. 1989. Aperiodic task scheduling for hard real-time systems. *Real-Time Syst. 1,* 1 (June), 27–60.

SPURI, M. AND BUTTAZZO, G. C. 1994. Efficient aperiodic service under earliest deadline scheduling. In *RTSS '94: Proceedings of the* 15^{th} *IEEE Real-Time Systems Symposium*. IEEE Computer Society, Los Alamitos, CA. 2–11.

STROSNIDER, J. K., LEHOCZKY, J. P., AND SHA, L. 1995. The deferrable server algorithm for enhanced aperiodic responsiveness in hard real-time environments. *IEEE Trans. Comput. 44,* 1 (Jan.), 73–91.

Received September 2004; revised July 2006 and December 2006; accepted March 2007