Composition of Schedulability Analyses for Real-Time Multiprocessor Systems

Jinkyu Lee, *Member, IEEE*, Kang G. Shin, *Life Fellow, IEEE*, Insik Shin, *Member, IEEE*, and Arvind Easwaran

Abstract—With increasing popularity and deployment of multi-core chips in embedded systems, a number of real-time multiprocessor scheduling algorithms have been proposed along with their schedulability analyses (or tests), which verify temporal correctness under a specific algorithm. Each of these algorithms often comes with several different schedulability tests, especially when it is difficult to find exact schedulability tests for the algorithm. Such tests usually find *different* task sets deemed schedulable even under the same scheduling algorithm. While these different tests have been compared with each other in terms of schedulability performance, little has been done on how to combine such different tests to improve the overall schedulability of a given scheduling algorithm beyond a simple union of their individual schedulability. Motivated by this, we propose a *composition theory* for schedulability tests with two new methods. The first method composes task-level timing guarantees derived from different schedulability tests, and the second one derives system-level schedulability results from a single schedulability test. The unified composition theory with these two methods then utilizes existing schedulability tests effectively so as to cover additional schedulable task sets. The proposed composition theory is shown to be applicable to most existing preemptive/non-preemptive scheduling algorithms. We also present three case-studies, demonstrating how and by how much the theory can improve schedulability by composing existing schedulability tests. Our evaluation results also show that the composition theory makes it possible to cover up to 181.7 percent additional schedulable task sets for preemptive fpEDF, preemptive EDF and non-preemptive EDF scheduling algorithms beyond their existing tests.

Index Terms—Composition of schedulability analyses, real-time scheduling, real-time systems, multiprocessor systems

1 Introduction

To keep pace with the growing deployment of multi-core chips in embedded real-time systems, researchers and practitioners have been paying considerable attention to multiprocessor scheduling subject to timing constraints. The multiprocessor scheduling algorithms proposed thus far can be broadly classified as partitioned or global. While partitioned scheduling restricts each task to run only on a designated processor, global scheduling allows a task to migrate from one processor to another. While researchers have studied both types of scheduling to exploit their advantages (see [1] for a survey), we will in this paper focus on global scheduling, especially in view of its potential for full utilization of processors. Numerous global scheduling algorithms have been proposed for higher processor utilization, less preemption and migration overheads, and broader accommodation of other non-functional task requirements (e.g., preemptive or non-preemptive).

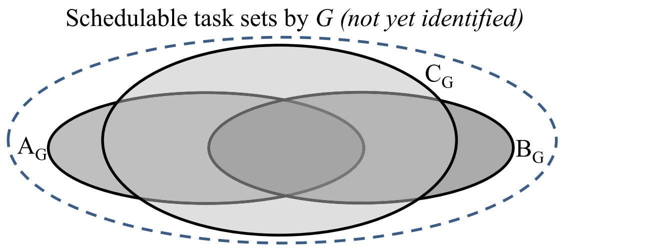
- *J. Lee is with Department of Computer Science and Engineering, Sungkyunkwan University, Suwon, Gyeonggi-Do, South Korea, E-mail: jinkyu.lee@skku.edu.*
- *K.G. Shin is with Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2121. E-mail: kgshin@eecs.umich.edu.*
- *I. Shin is with the Department of Computer Science, KAIST, Daejeon, South Korea. E-mail: insik.shin@cs.kaist.ac.kr.*
- *A. Easwaran is with School of Computer Engineering, Nanyang Technological University, Singapore. E-mail: arvinde@ntu.edu.sg.*

Manuscript received 20 Jan. 2013; revised 30 Dec. 2013; accepted 10 Jan. 2014. Date of publication 24 Feb. 2014; date of current version 13 Mar. 2015. Recommended for acceptance by A. Zomaya.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2014.2308183

Schedulability analyses (or tests) have been developed to guarantee the timing requirements of task sets under each scheduling algorithm. Since it is in general difficult to find exact deadline-miss conditions, each schedulability test determines, in a sufficient way, whether or not a task set is schedulable (i.e., guaranteeing all deadlines). The only known necessary and sufficient schedulability test in global multiprocessor scheduling is intended for a class of optimal scheduling algorithms with some restrictions, which are preemptive scheduling (allowing a higher-priority task to preempt a currently-executing lower-priority task) and the implicit deadline task model (in which the relative deadline of each task is the same as the period of the task) [2], [3], [4]. As a result, different sufficient schedulability tests have been developed for various scheduling algorithms such as global preemptive EDF [5]. A new schedulability test is expected to discover additional schedulable task sets which are deemed unschedulable by the existing tests.

However, very few of them have explored how to effectively utilize existing tests for better schedulability. The only way known thus far to *compose* the tests is to apply them *sequentially*; a task set is schedulable under a scheduling algorithm if at least one of its schedulability tests deems the task set schedulable. For example, suppose that there are three schedulability tests A_G, B_G and C_G for a scheduling algorithm G as shown in Fig. 1a. To date, we can guarantee the schedulability of only those task sets that are covered by A_G, B_G or C_G.

Our goal of this paper is to develop a new theory that effectively composes existing schedulability tests to cover more schedulable task sets. For example, we would like to



(a) Schedulable task sets proven by different tests A_G, B_G and C_G

	τ_1	τ_2	τ_3
A_G	✗	✗	✓
B_G	✓	✓	✗
A_G+B_G	?	?	?

(b) Task-level composition of A_G and B_G for $\{\tau_1,\tau_2,\tau_3\}$

m=1	τ_4,τ_5	✓
m=1	τ_5,τ_6	✓
m=2	**τ_4,τ_5,τ_6**	?

(c) System-level composition of C_G for $\{\tau_4,\tau_5,\tau_6\}$

Fig. 1. Three schedulability tests A_G, B_G and C_G of a scheduling algorithm G on a two-processor platform, and their compositions.

find additional schedulable task sets *not* covered by any of A_G, B_G and C_G, by composing the three schedulability tests in Fig. 1. For this, we must pay attention to *task* (instead of set) level schedulability. Although a test cannot guarantee the schedulability of an entire task set, we can find from the test that some of the tasks cannot trigger the first deadline-miss. Such task-level schedulability results can then be composed towards the schedulability guarantee of the entire task set. This way, we develop two composition methods. The first method provides a theoretical basis for composing task-level schedulability guarantees derived from different schedulability tests. For example, suppose that a task set $\{\tau_1, \tau_2, \tau_3\}$ on a two-processor platform scheduled by G is not deemed schedulable by any of A_G, B_G and C_G, but A_G and B_G can provide information on task-level schedulability as shown in Fig. 1b. If we compose the task-level schedulability results from A_G and B_G, no task can trigger the first deadline-miss, and the composition can therefore guarantee the schedulability of the task set $\{\tau_1, \tau_2, \tau_3\}$.

The second method introduces *system-level* composition by applying a single schedulability test to a subset of tasks on a subset of processors. For example, suppose that the schedulability test C_G itself cannot guarantee the schedulability of a task set $\{\tau_4, \tau_5, \tau_6\}$ on a two-processor platform scheduled by G, but it deems both $\{\tau_4, \tau_5\}$ and $\{\tau_5, \tau_6\}$ schedulable on a one-processor platform scheduled by G, as shown in Fig. 1c. Then, we can guarantee that none of $\{\tau_4, \tau_5, \tau_6\}$ triggers the first deadline miss when the task set is executed on a two-processor platform scheduled by G. Our composition theory proves that if a task set is schedulable on an m-processor platform by G, the task set with any x additional tasks is also schedulable on an $(m+x)$-processor platform by G, which will be detailed in Section 3. Finally, we develop a unified *composition theory* by combining these two composition methods.

The main advantage of our composition theory is its wide applicability; it can be applied to schedulability tests of *any* global work-conserving regular scheduling algorithm[1] regardless whether it allows preemption or not. In addition to deriving the best schedulability results by applying the theory to all the existing schedulability tests of a target scheduling algorithm, we may apply the theory to only some of the tests, improving some schedulability results without imposing additional time-complexity. This is very useful for online schedulability guarantees, in which time-complexity matters much.

One may wonder how and by how much the composition theory improves the schedulability of individual scheduling algorithms. Thus, we consider three prevalent scheduling algorithms (global preemptive EDF [5], global preemptive fpEDF [6], global non-preemptive EDF [7]), and then show how the composition theory derives new task- and set-level schedulability guarantees efficiently from existing schedulability tests of the target algorithms. We also demonstrate via simulation that the new schedulability guarantees derived by the composition theory find additional schedulable task sets that are not covered by the existing tests of the target algorithms.

In summary, this paper makes the following two key contributions.

- Development of a widely applicable composition theory for schedulability analysis in real-time multiprocessor systems, which efficiently exploits existing tests for better schedulability; and
- Demonstration of the composition theory's improvement of schedulability of popular scheduling algorithms.

The rest of this paper is organized as follows. Section 2 presents our system model. Section 3 provides an overview and classification of existing schedulability tests, and develops the composition theory for schedulability tests. Sections 4, 5 and 6 show how and by how much the composition theory improves the schedulability of three popular scheduling algorithms using their existing schedulability tests. Finally, Section 7 concludes the paper.

2 System Model, Assumptions, and Notations

In this paper we assume a sporadic task model [8], in which task τ_i in a task set Φ is specified as (T_i, C_i, D_i), where T_i is the minimum separation, C_i the worst-case execution time, and D_i the relative deadline. We focus on implicit (i.e., $C_i \le D_i = T_i$) and constrained (i.e., $C_i \le D_i \le T_i$) deadline tasks. A task τ_i invokes a series of jobs, each separated from its predecessor/successor by at least T_i time units. We assume that a job cannot be executed in parallel.

In this paper, we focus on (i) a multiprocessor platform consisting of m identical processors; (ii) global work-conserving scheduling algorithms under which a job can be executed on any processor, and processors are not left idle

1. The terms, *work-conserving* and *regular*, will be defined and illustrated in Section 2; most existing global scheduling algorithms are work-conserving and regular.

if there is a ready job; and (iii) *regular* scheduling algorithms, defined as follows.

Definition 1. *Suppose X is an m-processor platform, and X' is a platform on which the number of available processors is always at least m. A scheduling algorithm G is said to be* regular *if every job J invoked by tasks in a task set Φ satisfies $R_J \geq R'_J$, where R_J is the duration between the release and the completion of J when Φ is scheduled by G on a platform X and R'_J is the corresponding duration of the same job when Φ is scheduled by G on another platform X'.*

Most, if not all, well-known global real-time scheduling algorithms are regular, including preemptive/non-preemptive Rate Monotonic (RM) [5], Deadline Monotonic (DM) [9] and EDF [5], and preemptive fpEDF [6], EDF-US [10], Earliest Deadline first until Zero-Laxity (EDZL) [11], Fixed Priority until Zero-Laxity (FPZL) [12] and Least Laxity First (LLF) [13]. Note that one can synthetically construct irregular global work-conserving algorithms by employing some degree of randomness in prioritizing jobs, which are, in general, unuseful. For completeness, the following lemma shows why the above-mentioned algorithms are regular.

Lemma 1. *The preemptive and non-preemptive EDF algorithms are regular.*

Proof. We prove the lemma by contradiction. We use the same definitions of X and X' as those in Definition 1.

Suppose preemptive EDF is not regular and let t be the first time instant such that a particular execution of a job J^* is finished at t when Φ is scheduled by a scheduling algorithm G on X, while the corresponding execution is not finished until t on X'. This means that J^* on X executes in $[t-1, t)$, while J^* on X' does not.

Under preemptive EDF, the priorities of jobs are determined only by their deadlines, and the number of available processors of X' is always no smaller than that of X. Therefore, the only way for t to exist is that the number of jobs with higher priority than J^* on X' is strictly larger than that on X in $[t-1, t)$, contradicting the definition of t, i.e., every execution before t on X' is finished no later than the corresponding execution on X. This proves the lemma for preemptive EDF.

When it comes to non-preemptive EDF, the main difference is that we should count not only the number of jobs with higher priority than J^*, but also the number of jobs whose execution starts before t and has not been finished until t. Such non-preemptive jobs can block the execution of J^* regardless of their priorities. This is because, once a non-preemptive job starts its execution, it does not stop. So, the definition of t also contradicts for the same reason. This proves the lemma for non-preemptive EDF. □

Similar to preemptive/non-preemptive EDF, we can prove the regularity of other algorithms.

In this paper, we restrict our attention to global work-conserving regular scheduling algorithms, and therefore, we will use the term "scheduling algorithm G" to mean "global work-conserving regular scheduling algorithm G." For simplicity of presentation, let $|A|$ be the cardinality of A, and define $\delta_i = C_i/D_i$ (called task density), $u_i = C_i/T_i$ (called task utilization), $V_i = C_i/\max\big(0, D_i - \max_{\tau_j \in \tau} C_j\big)$, and $U_{sys} = \sum_{\tau_i \in \tau} u_i$ (called task set utilization). Also, we define three types of subsets of Φ as follows:

$$\Phi_y(\tau_k) \triangleq \Phi - \{\tau_i | y \text{ largest } \delta_i \text{ in } \Phi - \{\tau_k\}\}, \tag{1}$$

$$\Phi'_y(\tau_k) \triangleq \Phi - \{\tau_i | y \text{ largest } u_i \text{ in } \Phi - \{\tau_k\}\}, \text{ and} \tag{2}$$

$$\Phi''_y(\tau_k) \triangleq \Phi - \{\tau_i | y \text{ largest } V_i \text{ in } \Phi - \{\tau_k\}\}. \tag{3}$$

A schedulability test X_G of a scheduling algorithm G is said to *dominate* another schedulability test Y_G of G when every task set deemed schedulable by Y_G is also deemed schedulable by X_G, but the converse does not hold.

3 Composition Theory for Schedulability Tests

This section details the development of our composition theory for schedulability tests. We first overview and then classify existing schedulability tests. Next, we develop two independent methods for composing schedulability tests, and finally utilize them to develop a new unified composition theory.

3.1 Overview and Classification of Existing Schedulability Tests

Before overviewing existing schedulability tests, we formally define the schedulability of a task set as follows.

Definition 2. *A task set Φ is said to be* schedulable *by a scheduling algorithm* G *on an m-processor platform, if no job invoked by tasks in Φ misses its deadline when Φ is scheduled by* G *on the m-processor platform.*

Schedulability tests determine if a given task set is schedulable by a given algorithm on a given platform. To make such a decision, most schedulability tests use the following equivalent definition of schedulability instead of Definition 2.

Definition 3. *A task set Φ is said to be* schedulable *by a scheduling algorithm* G *on an m-processor platform, if for every task $\tau_k \in \Phi$, no job invoked by τ_k causes the* first deadline miss *when Φ is scheduled by* G *on the m-processor platform.*

Using the above definition, schedulability tests find necessary conditions for a job invoked by a task to trigger the *first deadline miss*, subject to no deadline miss for any other job. Then, a schedulability test regards the task set schedulable if the test guarantees that all individual jobs fail to satisfy the necessary conditions.

Depending on whether the identity of the task whose job can trigger the first deadline miss is known or not, we classify existing schedulability tests as per-task/utilization-based. The per-task schedulability test checks for each individual task whether any of its jobs can trigger the first deadline miss or not. Therefore, although the per-task schedulability test may deem a task set to be unschedulable, it is still possible for the test to identify tasks in the set that cannot trigger the first deadline miss. These are those tasks for which the test was successful.

TABLE 1
Examples of Per-Task-Based and Utilization-Based Schedulability Tests

Scheduling algorithm	Per-task-based schedulability tests	Utilization-based schedulability tests
preemptive EDF [5]	[14–18]	[19–21]
non-preemptive EDF [7]	[22–24]	[25]
preemptive EDF-US [10]		[10]
preemptive fpEDF [6]		[6]
preemptive FP [5]	[16, 18, 26]	[27]
non-preemptive FP [7]	[22, 28]	
preemptive DM-DS [29]		[27, 29]
preemptive EDZL [11]	[30, 31]	[32]
preemptive LLF [13]	[31, 33, 34]	

On the other hand, utilization-based schedulability tests only indicate the schedulability of a given task set; they do not distinguish each task's capability of triggering the first deadline miss. Typical examples of per-task- and utilization-based schedulability tests are shown in Table 1. For ease of understanding, we show two schedulability tests of the preemptive EDF scheduling algorithm, starting with a per-task-based schedulability test as follows:

Lemma 2 (Theorem 7 in [18]). *A task set Φ is schedulable by preemptive EDF on an m-processor platform if the following inequality holds for every task $\tau_k \in \Phi$:*

$$\sum_{\tau_i \in \Phi - \{\tau_k\}} \min\left(I_{k\leftarrow i}^{\mathsf{EDF}}, D_k - C_k + 1\right) < m \cdot (D_k - C_k + 1), \quad (4)$$

where $I_{k\leftarrow i}^{\mathsf{EDF}}$ is an upper-bound of the duration during which jobs of τ_i block a job of τ_k in an interval between the release time and deadline of the job of τ_k under preemptive EDF; see [18] for detail.

If Eq. (4) is satisfied for a given task τ_k, the schedulability test guarantees that no job invoked by τ_k triggers the first deadline miss when Φ is scheduled by preemptive EDF on an m-processor platform. This per-task behavior holds even if the schedulability test in Lemma 2 deems Φ (the entire task set) unschedulable.

The following lemma shows an example of utilization-based schedulability tests.

Lemma 3 (Theorem 3 in [19]). *A task set Φ is schedulable by preemptive EDF on an m-processor platform if the following inequality holds:*

$$\sum_{\tau_i \in \Phi} \delta_i \le m - (m-1) \cdot \max_{\tau_i \in \Phi} \delta_i. \quad (5)$$

Unlike the schedulability test in Lemma 2, the utilization-based test in Lemma 3 can only judge whether Φ (the entire task set) is schedulable or not, without indicating each task's capability of triggering the first deadline miss.

3.2 Composition Theory

While many real-time scheduling algorithms have been developed for multiprocessor platforms, their schedulability tests are usually only sufficient, but not necessary. To date, there are only a few multiprocessor scheduling algorithms with exact schedulability tests (i.e., a class of optimal algorithms for preemptive scheduling with the implicit deadline task model [2], [3], [4]), and their exact schedulability tests simply check if the system utilization (U_{sys}) is no greater than the number of processors (m).

Hence, various schedulability tests for each scheduling algorithm have been developed to find additional schedulable task sets that have not been covered by existing schedulability tests. However, little has been done on the effective utilization of existing schedulability tests of a given algorithm for a tighter analysis.[2] To date, the only way to utilize different schedulability tests of a given scheduling algorithm is to simply aggregate all schedulability results, i.e., a task set is schedulable by a given algorithm on a given platform, if at least one of the existing schedulability tests guarantees the task set to be schedulable.

Focusing on task-level schedulability, we develop two methods for *composing* schedulability tests of a given scheduling algorithm, which effectively utilize individual schedulability tests towards a tighter analysis. While the first method addresses how to compose task-level schedulability results of a given task set from different schedulability tests, the second method addresses how to derive new task-level schedulability results from a single schedulability test. We then unify the two methods, yielding a final composition theory.

The following lemma introduces the first method.

Lemma 4 (Task-level composition). *A task set Φ is schedulable by a scheduling algorithm* G *on an m-processor platform, if for every $\tau_k \in \Phi$, there exists a schedulability test X_G of* G *such that X_G guarantees that no job invoked by τ_k triggers the first deadline miss when Φ is scheduled by* G *on an m-processor platform.*

Proof. For every $\tau_k \in \Phi$, there exists a schedulability test which guarantees no job of τ_k will trigger the first deadline miss. Therefore, no job of tasks in Φ triggers the first deadline miss, thus making Φ schedulable. □

The lemma is straightforward, but can improve schedulability. The following example shows how the lemma improves the schedulability by utilizing existing tests.

Example 1. Consider a task set $\Phi = \{\tau_1(T_1 = 2, C_1 = 1, D_1 = 2), \tau_2(5, 2, 5), \tau_3(5, 3, 5)\}$. All existing schedulability tests of preemptive EDF [14], [15], [16], [17], [18], [19], [20], [21] deem the entire task set unschedulable by preemptive EDF on a two-processor platform. However, Bar [17] (likewise BeCi [16]) guarantees that no job invoked by τ_1 (likewise τ_2 and τ_3) triggers the first deadline miss, when Φ is scheduled by preemptive EDF on a two-processor platform.

Then, Lemma 4 with Bar and BeCi guarantees Φ to be schedulable by preemptive EDF on a two-processor platform, which is not guaranteed by any of the existing preemptive EDF schedulability tests (including Bar and BeCi) without this lemma.

Lemma 4 can be exploited only when per-task schedulability tests are involved. However, the lemma will be used even for utilization-based schedulability tests once we incorporate it into the second method as presented next.

2. One implicit composition of existing schedulability tests has been introduced in the evaluation section of [35] without any proof, which will be discussed briefly in Section 7.

Lemma 5 (System-level composition). *Consider two disjoint task sets Φ' and Φ''. If no job of a given task τ_k in Φ' triggers the first deadline miss when Φ' is scheduled by an algorithm G on an m'-processor platform, no job of the given task τ_k in Φ' triggers the first deadline miss when $\Phi' \cup \Phi''$ is scheduled by G on an $(m' + |\Phi''|)$-processor platform.*

Proof. Suppose that no job of a given task τ_k in Φ' triggers the first deadline miss when Φ' is scheduled by G on an m'-processor platform, but the first deadline miss is triggered by a job of τ_k in Φ' at t when $\Phi' \cup \Phi''$ is scheduled by G on an $(m' + |\Phi''|)$-processor platform.

Since we focus on tasks with $D_i \leq T_i$, each task has at most one unfinished job at any time before t. Therefore, jobs of tasks in Φ'' do not occupy more than $|\Phi''|$ processors at any time before t, meaning that at least m' processors are available for jobs invoked by tasks in Φ'. Since we focus on global work-conserving *regular* algorithms, the supposition implies no job of the given task τ_k $(\in \Phi')$ triggers the first miss when $\Phi' \cup \Phi''$ is scheduled by G on an $(m' + |\Phi''|)$-processor platform. Therefore, the lemma holds. □

The lemma implies that if Φ' is schedulable by G on an m'-processor platform, no job invoked by tasks in Φ' triggers the first deadline miss when $\Phi' \cup \Phi''$ is scheduled by G on an $(m' + |\Phi''|)$-processor platform. Note that the lemma makes no guarantee on the first deadline miss for jobs invoked by tasks in Φ''.

Note that the second method presented in Lemma 5 appears similar to the technique used for schedulability tests of EDF-variant scheduling which gives the highest priority to at most $m-1$ designated tasks, such as EDF$^{(k)}$ [19], EDF-US [10], and fpEDF [6]. For the algorithms, since the schedulability of the highest-priority tasks is guaranteed regardless of the remaining tasks, their schedulability tests can virtually assign a processor to each highest-priority task and find whether the remaining tasks are schedulable on the remaining processors. Unlike the technique that is specialized for the EDF-variant algorithms with task-level priority changes, we focus on a scheduling algorithm as it is without changing the priority of the algorithm, and therefore Lemma 5 can be applied to any global work-conserving regular algorithm.

We also note that it is challenging to generalize the technique specialized for the EDF-variant algorithms to any work-conserving regular algorithm. In this paper, we address two major challenges: one is in Lemma 5, where we use the "work-conserving regular" property to show that system-level composition holds, and the other is the monotonicity with respect to certain parameters that allows us to make the composition test efficient, which will be detailed in Sections 4, 5 and 6.

One may wonder how the above lemma can lead to a tighter analysis using existing schedulability tests, and hence, we present a simple illustrative example.

Example 2. Consider a task set $\Phi = \{\tau_1(T_1 = 2, C_1 = 1, D_1 = 2), \tau_2(3, 2, 3), \tau_3(6, 2, 6)\}$. All existing schedulability tests of preemptive EDF [14], [15], [16], [17], [18], [19], [20], [21] deem the task set unschedulable by preemptive EDF on a two-processor platform. However, GFB [19] (i.e., Lemma 3) deems $\Phi_A \triangleq \{\tau_1, \tau_3\}$ and $\Phi_B \triangleq \{\tau_2, \tau_3\}$ schedulable by preemptive EDF on a one-processor platform, respectively.

Then, by setting $\Phi' = \Phi_A$ and $\Phi'' = \Phi - \Phi_A$ ($\Phi' = \Phi_B$ and $\Phi'' = \Phi - \Phi_B$), Lemma 5 guarantees that no job invoked by τ_1 and τ_3 (τ_2 and τ_3) triggers the first deadline miss when Φ is scheduled by preemptive EDF on a two-processor platform. Therefore, no task's job can trigger the first deadline miss, implying that Φ is schedulable by preemptive EDF on a two-processor platform.

By incorporating Lemma 5 into Lemma 4, we present a final composition theory, which can potentially achieve a tighter analysis than applying the lemmas separately.

Theorem 1 (Composition Theory). *A task set Φ is schedulable by a scheduling algorithm* G *on an m-processor platform, if for every $\tau_k \in \Phi$, there exists a task set Φ^* and a schedulability test X_G of* G *such that*

- *$\tau_k \in \Phi^*$ and $\Phi^* \subseteq \Phi$; and*
- *X_G guarantees that no job invoked by τ_k triggers the first deadline miss when Φ^* is scheduled by* G *on an $(m - |\Phi - \Phi^*|)$-processor platform.*

Proof. By setting $\Phi' = \Phi^*$ and $\Phi'' = \Phi - \Phi^*$, Lemma 5 guarantees that no job invoked by a given τ_k triggers the first deadline miss. Since this holds for every $\tau_k \in \Phi$, Lemma 4 guarantees that Φ is schedulable. □

We illustrate how the theorem results in better schedulability using the following example.

Example 3. Consider a task set $\Phi = \{\tau_1(T_1 = 10, C_1 = 5, D_1 = 10), \tau_2(3, 2, 3), \tau_3(8, 4, 8)\}$. All existing schedulability tests of preemptive EDF [14], [15], [16], [17], [18], [19], [20], [21] deem the task set unschedulable by preemptive EDF on a two-processor platform. However, Bar [17] guarantees that no job invoked by τ_2 triggers the first deadline miss when Φ is scheduled by preemptive EDF on a two-processor platform, and GFB [19] (i.e., Lemma 3) guarantees that no job invoked by $\{\tau_1, \tau_3\}$ triggers the first deadline miss when $\{\tau_1, \tau_3\}$ is scheduled by G on a single-processor platform.

Then, applying Lemma 4 or 5 does not guarantee the schedulability of Φ. However, τ_2 satisfies the condition of Theorem 1 by Bar with $\Phi^* = \Phi$, and τ_1 and τ_3 do by GFB with $\Phi^* = \{\tau_1, \tau_3\}$. Therefore, Theorem 1 guarantees that Φ is schedulable by preemptive EDF on a two-processor platform.

While the proposed composition theory can be applied to schedulability tests of any global work-conserving regular scheduling algorithm regardless of its preemption policy (e.g., non-preemptive or preemptive), we will show in the next three sections how and by how much the composition theory can find additional schedulable task sets by composing existing schedulability tests of a scheduling algorithm. In particular, we will explain how to derive better schedulability results without imposing additional time-complexity, which is related to the way to examine all possible Φ^* efficiently. Also, we demonstrate how to derive the best schedulability results using the composition theory, which is the main contribution of this paper.

4 Case Study I: Preemptive fpEDF

Using our composition theory (Theorem 1), we can improve the schedulability of any global work-conserving regular algorithm. As an example, we show how and by how much the composition theory can improve the schedulability of preemptive fpEDF scheduling [6] using the only existing schedulability test [6]. We first present how to efficiently apply the composition theory to the test. Then, we show via simulation average schedulability improvement by our composition theory.

4.1 Composition Theory to the Preemptive fpEDF Schedulability Test

The preemptive fpEDF scheduling algorithm [6] gives the highest priority to the jobs invoked by tasks with $m-1$ largest δ_i (>0.5), and prioritizes the remaining jobs according to the preemptive EDF policy [5]. Then, the following lemma presents the only existing schedulability test of preemptive fpEDF.

Lemma 6 (Theorem 6 in [6]). *A task set Φ is schedulable by preemptive fpEDF on an m-processor platform if either Eq. (6) or (7) holds:*

$$\sum_{\tau_i \in \Phi} \delta_i \le m - (m-1) \cdot \max_{\tau_i \in , \Phi} \delta_i, \tag{6}$$

$$\sum_{\tau_i \in \Phi} \delta_i \le \frac{m}{2} + \max_{\tau_i \in \Phi} \delta_i. \tag{7}$$

Note that Lemma 6 holds only when $m \ge 2$. If $m = 1$, the RHS of Eq. (7) should be equal to 1. The time-complexity of Lemma 6 is $O(|\Phi|)$.

To derive the maximum schedulability of a task set Φ from Theorem 1 using the above schedulability test, we may check all possible subsets Φ^* ($\subset \Phi$) for the test. However, such checks incur high time-complexity; for a given size of subset Φ^*, the number of subsets to be checked is $\binom{|\Phi|}{|\Phi^*|}$, and hence, the total number of task sets to be checked for Theorem 1 for each schedulability test is $\sum_{x=|\Phi|-m+1}^{|\Phi|}\binom{|\Phi|}{x}$, which is an exponential function of $|\Phi|$.

With a careful examination of the test, however, we can derive the best schedulability result of a task without checking all possible subsets, as stated in the following lemma.

Lemma 7. *Suppose that the scheduler is preemptive fpEDF and the underlying schedulability test is Lemma 6. Then, applying Theorem 1 for a given task τ_k with all possible Φ^* is equivalent to that only with every $\Phi_y(\tau_k)$ for $0 \le y \le m-1$, where $\Phi_y(\tau_k)$ is defined in Eq. (1).*

Proof. Suppose there exists a task set Φ^* that contains τ_k and satisfies (i) Eq. (6) or (ii) Eq. (7) on an $(m - |\Phi - \Phi^*|)$-processor platform. We prove that a corresponding $\Phi_y(\tau_k)$ satisfies the condition on the same platform.

For Case (i), if Φ^* contains a task with the largest δ_i (denoted by τ_x) among tasks in $\Phi - \{\tau_k\}$, we exchange the task in Φ^* with a task with the smallest δ_i (denoted by τ_y) among tasks in $\Phi - \Phi^*$. The exchange between τ_x and τ_y decreases the LHS of Eq. (6) and does not decrease the RHS of Eq. (6), implying that Φ^* with the exchange satisfies Eq. (6) on an $(m - |\Phi - \Phi^*|)$-processor platform. We repeat this exchange for the $2\text{nd}, 3\text{rd}, \ldots, |\Phi - \Phi^*|\text{th}$ largest δ_i among tasks in $\Phi - \{\tau_k\}$. Then, Φ^* with all the exchanges done is the same as $\Phi_y(\tau_k)$ for $y = |\Phi - \Phi^*|$. Note that we do not care for $|\Phi - \Phi^*| \ge m$; in such a case, there is no available processor (i.e., $m - |\Phi - \Phi^*| \le 0$).

For Case (ii), we perform the same exchange as Case (i). The exchange between τ_x and τ_y decreases the LHS of Eq. (7) by $\delta_x - \delta_y$, but decreases the RHS by at most $\delta_x - \delta_y$ since $\max_{\tau_i \in \Phi^*} \delta_i$ after the exchange is larger than, or equal to δ_y. Then, Φ^* with the exchange still satisfies Eq. (7) on an $(m - |\Phi - \Phi^*|)$-processor platform. Similar to Case (i), we repeat such an exchange for the 2nd, $3\text{rd}, \ldots, |\Phi - \Phi^*|\text{th}$ largest δ_i among tasks in $\Phi - \{\tau_k\}$, and finally Φ^* with all the exchanges is the same as $\Phi_y(\tau_k)$ for $y = |\Phi - \Phi^*|$.

By Cases (i) and (ii), the lemma holds. □

Thanks to Lemma 7, deriving a new schedulability condition of a given task from the test in Lemma 6 requires only m task sets (instead of all possible subsets) to be checked. Besides, it requires only $O(|\Phi|)$ time-complexity to test the m task sets. That is, while a naive approach requires $O(|\Phi| \cdot m)$ time-complexity because the test in Lemma 6 itself needs $O(|\Phi|)$ time-complexity, we reduce it by $O(|\Phi|)$ if we store $\sum_{\tau_i \in \Phi_y(\tau_k)} \delta_i$ for the next set ($\sum_{\tau_i \in \Phi_{y+1}(\tau_k)} \delta_i$). Then, the composition theory derives a new closed-form preemptive fpEDF schedulability test from Lemma 6 without imposing additional time-complexity, as stated in the following theorem.

Theorem 2 (New Preemptive fpEDF Schedulability Test). *A task set Φ is schedulable by preemptive fpEDF on an m-processor platform if either Eq. (8) or (9) holds:*

$$\sum_{\tau_i \in \Phi} \delta'_i \le m - (m-1) \cdot \max_{\tau_i \in \Phi} \delta_i, \tag{8}$$

$$\sum_{\tau_i \in \Phi} \delta''_i \le \frac{m}{2} + \max_{\tau_i \in \Phi} \delta_i, \tag{9}$$

where δ'_k

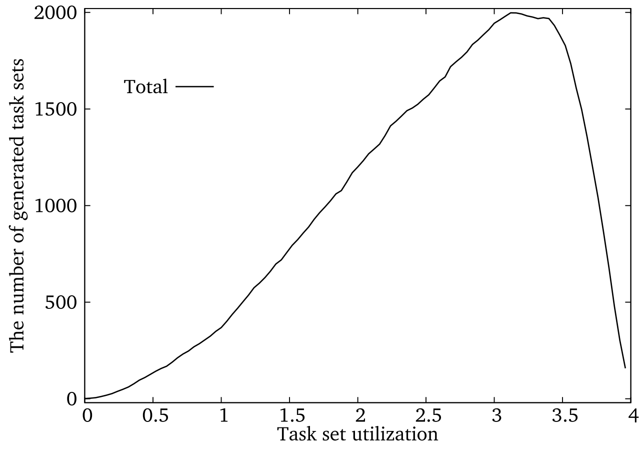
$$= \begin{cases} \min(\delta_k, 1 - \max_{\tau_i \in \Phi} \delta_i), & \text{if } \tau_k \text{ belongs to tasks with} \\ & \text{m-1 largest } \delta_i \text{ in } \Phi - \{\tau_{max}\}, \\ \delta_k, & \text{otherwise,} \end{cases} \tag{10}$$

δ''_k

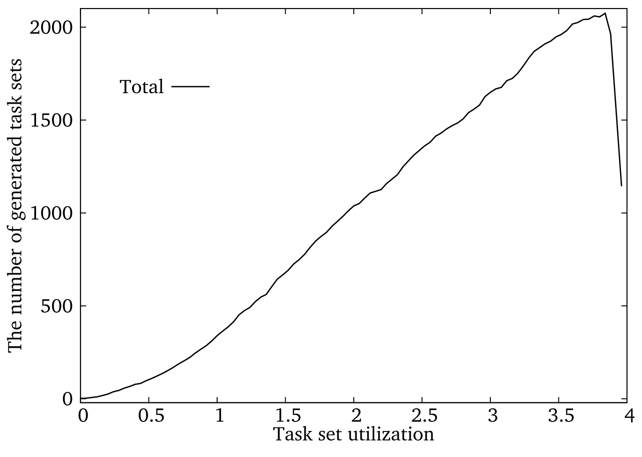
$$= \begin{cases} \min(\delta_k, 0.5), & \text{if } \tau_k \text{ belongs to tasks with} \\ & \text{m-2 largest } \delta_i \text{ in } \Phi - \{\tau_{max}\}, \\ \delta_k, & \text{otherwise,} \end{cases} \tag{11}$$

and τ_{max} is a task with the largest δ_i in Φ.

Proof. Case (i): We first show that if Eq. (8) is satisfied, then for every task τ_k in Φ, there exists a task set Φ^* such that $\tau_k \in \Phi^*$ and $\Phi^* \subseteq \Phi$ hold and the test in Theorem 2 guarantees that no job of τ_k triggers the first deadline miss when Φ^* is scheduled by preemptive fpEDF on an $(m - |\Phi - \Phi^*|)$-processor platform.



(a) $m = 4$, constrained deadline task sets



(b) $m = 4$, implicit deadline task sets

Fig. 2. Generated task sets.

Suppose that Eq. (8) holds. Let Φ^* denote $\{\tau_{max}\} \cup \{\tau_k \in \Phi - \{\tau_{max}\} | \min(\delta_k, 1 - \max_{\tau_i \in \Phi} \delta_i) = \delta_k\}$. Then, $\min(\delta_k, 1 - \max_{\tau_i \in \Phi} \delta_i) = 1 - \max_{\tau_i \in \Phi} \delta_i$ holds for every task τ_k in $\Phi - \Phi^*$. Therefore, if we deduct $1 - \max_{\tau_i \in \Phi} \delta_i$ for every $\tau_k \in \Phi - \Phi^*$ in both the LHS and RHS in Eq. (8), the final condition is the same as Eq. (6) for Φ^* and $(m - |\Phi - \Phi^*|)$-processors as follows:

$$\begin{aligned} &\sum_{\tau_i \in \Phi} \delta'_i \le m - (m-1) \cdot \max_{\tau_i \in \Phi} \delta_i \\ \Rightarrow &\sum_{\tau_i \in \Phi^*} \delta_i \le (m - |\Phi - \Phi^*|) - (m - |\Phi - \Phi^*| - 1) \cdot \max_{\tau_i \in \Phi} \delta_i \\ \Leftrightarrow &\sum_{\tau_i \in \Phi^*} \delta_i \le (m - |\Phi - \Phi^*|) - (m - |\Phi - \Phi^*| - 1) \cdot \max_{\tau_i \in \Phi^*} \delta_i, \end{aligned} \tag{12}$$

which implies a success in meeting the schedulability of τ_{max}.

The schedulability for other tasks can be met by exchanging a task in Φ^* with a task in $\Phi - \Phi^*$. That is, if $\Phi - \Phi^*$ contains a task with the xth largest δ_i among tasks in Φ (denoted by $\tau_{max:x}$), we exchange τ_{max} in Φ^* with $\tau_{max:x}$ in $\Phi - \Phi^*$. Such an exchange decreases the LHS of the final condition of Eq. (12) and increases the RHS, implying the final condition of Eq. (12) holds for the new Φ^*. Therefore, we satisfy the schedulability for every task $\tau_i \in \Phi$ is guaranteed by Lemma 6.

Case (ii): Suppose Eq. (9) holds. The proof is similar to Case (i), and therefore, we only give a high-level account of Eq. (9) using Eq. (8). If we focus on the RHS of Eq. (6) and that of Eq. (7), (i.e., $m \cdot (1 - \max_{\tau_i \in \Phi} \delta_i) + \max_{\tau_i \in \Phi} \delta_i$ versus $m \cdot 0.5 + \max_{\tau_i \in \Phi} \delta_i$), they are respectively reduced by $1 - \max_{\tau_i \in \Phi} \delta_i$ and 0.5 whenever m is reduced by 1. Therefore, while we benefit by excluding a processor and a task with $\delta_i > 1 - \max_{\tau_i \in \Phi} \delta_i$ repeatedly until there is only one processor (resulting in Eq. (8) from Eq. (6)), we do the same thing for a task with $\delta_i > 0.5$ (resulting in Eq. (9) from Eq. (7)). □

Note that the time-complexity of Theorem 2 is $O(|\Phi|)$, which is the same as that of Lemma 6.

4.2 Task Set Generation

To evaluate the schedulability improvement by our composition theory, we generate task sets based on a technique proposed earlier [36], which has also been used in many previous studies (e.g., see [18], [34], [37]). We have three input parameters: (a) the number of processors m $(2, 4$ or $8)$, (b) the task system (constrained or implicit deadline), and (c) individual task utilization (C_i/T_i) distribution (bimodal with parameter: 0.1, 0.3, 0.5, 0.7, or 0.9, or exponential with parameter: 0.1, 0.3, 0.5, 0.7, or 0.9). For a given bimodal parameter p, a value for C_i/T_i is uniformly chosen in $[0, 0.5)$ with probability p, and in $[0.5, 1)$ with probability $1 - p$. For a given exponential parameter $1/\lambda$, a value for C_i/T_i is chosen according to the exponential distribution whose probability density function is $\lambda \cdot \exp(-\lambda \cdot x)$.

For each task, T_i is uniformly chosen in $[1, T_{max} = 1{,}000]$, C_i is chosen based on the bimodal or exponential parameter, and D_i is uniformly chosen in $[C_i, T_i]$ for constrained deadline task systems or D_i is equal to T_i for implicit deadline task systems.

For each combination of (a), (b) and (c), we repeat the following procedure and generate 10,000 task sets, thus resulting in 100,000 task sets for any given m and the type of task sets.

1) Initially, we generate a set of $m + 1$ tasks.
2) In order to exclude unschedulable sets, we check whether the generated task set can pass a necessary feasibility condition [38].
3) If it fails to pass the feasibility test, we discard the generated task set and return to Step 1. Otherwise, we include this set for evaluation. Then, this task set serves as a basis for the next new set; we create a new set by adding a new task into an already created and tested set, and return to Step 2.

Then, Figs. 2a and 2b illustrate the distribution of the generated task sets according to the task set utilization (U_{sys}) for $m = 4$. The trend for the distribution of other m is similar to the figures.

These generated sets will be used to evaluate the schedulability improvement by the composition theory for preemptive fpEDF, preemptive EDF and non-preemptive EDF, respectively in Sections 4.3, 5.2 and 6.2.

TABLE 2
The Number of Schedulable Task Sets by, and Time-Complexity of fpEDF and fpEDF-Comp

Implicit deadline task sets		
	fpEDF [6]	fpEDF-comp
# of sched. sets: $m = 4$	44871	56074
# of sched. sets: $m = 8$	31609	45940
Time-complexity	$O(\vert\Phi\vert)$	$O(\vert\Phi\vert)$

Constrained deadline task sets		
	fpEDF [6]	fpEDF-comp
# of sched. sets: $m = 4$	17942	32102
# of sched. sets: $m = 8$	8952	25217
Time-complexity	$O(\vert\Phi\vert)$	$O(\vert\Phi\vert)$

4.3 Evaluation of Schedulability Improvement

Now, we evaluate how much the composition theory in Theorem 1 improves the schedulability of preemptive fpEDF. Since Lemma 6 is the only existing schedulability test, we compare the average schedulability results by the test with those by the composition theory with the test (i.e., Theorem 2). The former and latter are annotated as fpEDF and fpEDF-comp, respectively. For evaluation, we test 100,000 implicit and constrained task sets for each of m, as described in Section 4.2.

Table 2 summarizes the time-complexity and the number of schedulable task sets of fpEDF and fpEDF-comp for both implicit and constrained deadline task sets. In particular, Fig. 3 exhibits more detailed schedulability results for constrained deadline task sets. Each figure consists of several plots, each showing the number of task sets proven schedulable by each schedulability test, with task set utilization (U_{sys}) in the interval $[U_{sys} - 0.01 \cdot m, U_{sys} + 0.01 \cdot m)$.

As shown in Table 2, and Figs. 3a and 3b, fpEDF-comp finds a large number of additional schedulable task sets which are deemed unschedulable by fpEDF. In particular, the schedulability improvements are 78.9 and 181.7 percent for constrained deadline task sets when $m = 4$ and 8, respectively. Since fpEDF is the only existing schedulability test of fpEDF, our composition theory significantly improves the schedulability of the algorithm. Also, such improvement does not increase time-complexity in that the time-complexity of both tests is $O(|\Phi|)$, as shown in the table.

5 Case Study II: Preemptive EDF

While Section 4 was concerned with how and by how much the composition theory improves the schedulability of preemptive fpEDF, this section shows them for preemptive EDF [5]. Unlike preemptive fpEDF, there are many existing schedulability tests for preemptive EDF, and hence, the composition theory can fully utilize Lemma 4 (composition of task-level schedulability results from different tests) as well as Lemma 5 (composition of system-level schedulability results from a single test).

5.1 Application of Composition Theory to the Preemptive EDF Schedulability Tests

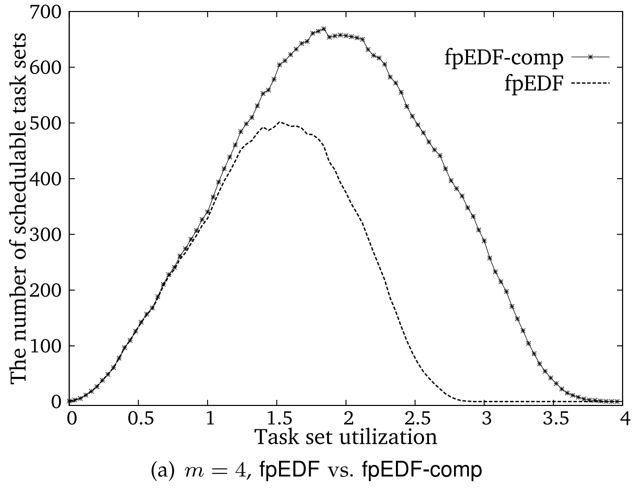
The preemptive EDF scheduling algorithm [5] prioritizes jobs according to their deadlines, and schedules them preemptively. There have been many schedulability tests of preemptive EDF developed thus far [14], [15], [16], [17], [18], [19], [20], [21]. We choose two schedulability tests [19], [20], and show how the composition theory in Theorem 1 efficiently derives new task-level schedulability results from the chosen tests.

We have already presented a schedulability test of [19] in Lemma 3. Similar to Lemma 7 for the schedulability test in Lemma 6, we can derive the best new task-level schedulability results without checking all possible subsets as follows.

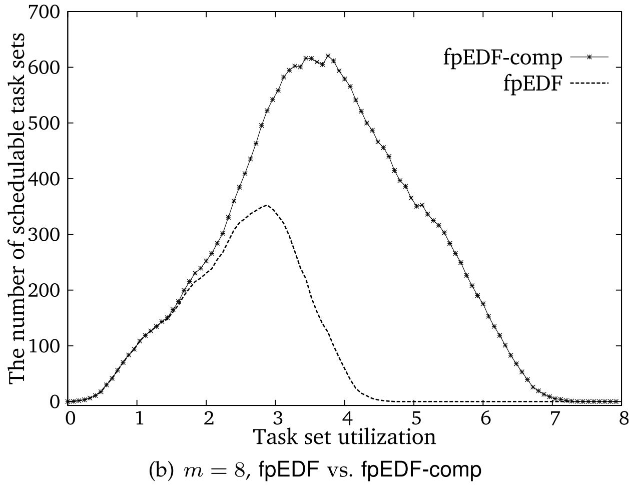
Lemma 8. *Suppose the scheduler is preemptive EDF and the underlying schedulability test is the one in Lemma 3. Then, applying Theorem 1 for a given task τ_k with all possible Φ^* for the test is equivalent to that only with every $\Phi_y(\tau_k)$ for $0 \le y \le m-1$, where $\Phi_y(\tau_k)$ is defined in Eq. (1).*

Proof. The proof is the same as Case (i) in Lemma 7. □

Lemma 8 helps efficiently derive a new schedulability condition of a given task from the test in Lemma 3; we need to investigate only m task sets in order to derive the best task-level schedulability results for a task. Then, the naive approach for the investigation requires $O(m \cdot |\Phi|)$, but we can reduce the time-complexity, as stated in the following theorem.



(a) $m = 4$, fpEDF vs. fpEDF-comp



(b) $m = 8$, fpEDF vs. fpEDF-comp

Fig. 3. Schedulability results of preemptive fpEDF schedulability tests for constrained deadline task sets.

Theorem 3 (new preemptive EDF schedulability test). *A task set Φ is schedulable by preemptive EDF on an m-processor platform if Eq. (8) holds.*

Proof. The lemma holds by Case (i) of the proof of Theorem 2. □

Note that the time-complexity of the theorem is only $O(|\Phi|)$, which is the same as that of Lemma 3.

We now show how to select subsets to be checked for Theorem 1 when we apply another schedulability test of preemptive EDF. The test has been derived based on the force-forward demand bound function (ffdbf) as stated in the following lemma.

Lemma 9 ([20], Summarized in Theorem 6 in [35]). *A task set Φ is schedulable by preemptive EDF on an m-processor platform if there exists σ such that $\max_{\tau_i\in\Phi}\delta_i \le \sigma < \frac{m-\sum_{\tau_i\in\Phi}u_i}{m-1} - \epsilon$ (with an arbitrarily small ϵ) holds along with the following condition:*

$$\frac{\mathsf{ffdbf}(t,\sigma)}{t} \le (m-(m-1)\cdot\sigma), \forall t \ge 0, \tag{13}$$

$$\text{where } \mathsf{ffdbf}(t,\sigma) \triangleq \left(\left\lfloor \frac{t-D_i}{T_i}\right\rfloor + 1\right)\cdot C_i + \sigma\cdot\max\left(0, (t-D_i) \bmod T_i - T_i + \frac{T_i}{\sigma}\right). \tag{14}$$

The authors of [20], [35] described how to reduce the search space of t and σ. The overall time-complexity of the test in Lemma 9 is proven to be pseudo-polynomial. Note that the schedulability test in Lemma 9 is a generalization of that in Lemma 3; they are equivalent for implicit deadline task sets, and the former dominates the latter for constrained deadline task sets at the expense of high time-complexity.

Unlike Lemma 8, we cannot find an efficient way to search all possible Φ^*'s. This is because, while the test in Lemma 3 is simply a function of every δ_i, that in Lemma 9 needs to investigate Eq. (13) for all possible t and σ. Therefore, without an exhaustive search, it is difficult to identify which task should be removed to minimize the LHS of Eq. (13).

Fortunately, we observe that $\sum_{\tau_i\in\Phi}\delta_i$ and $\sum_{\tau_i\in\Phi}u_i$ are, respectively, an upper-bound and a lower-bound of the LHS of Eq. (13). Using this observation, we use the following heuristics to identify "good" subsets to investigate for testing: (i) every $\Phi_y(\tau_k)$ in Eq. (1) for $0 \le y \le m-1$ and (ii) every $\Phi'_y(\tau_k)$ in Eq. (2) for $0 \le y \le m-1$. Then, we only check $2\cdot m$ task sets, improving tractability but missing some potential task-level schedulability conditions that can be obtained by the composition theory. The following theorem records the independent schedulability test derived from Lemma 9 using the composition theory.

Theorem 4 (New Preemptive EDF Schedulability Test). *A task set Φ is schedulable by preemptive EDF on an m-processor platform if for each task τ_k, there exists $\Phi_y(\tau_k)$ or $\Phi'_y(\tau_k)$, which satisfies Lemma 9 on an $(m-y)$-processor platform $(0 \le y \le m-1)$.*

Proof. By Theorem 1, the theorem holds. □

5.2 Evaluation of Schedulability Improvement

We now evaluate the quantitative improvement of preemptive EDF schedulability by the composition theory. Unlike preemptive fpEDF, there are many existing schedulability tests for preemptive EDF. We annotate them as GFB [19] (i.e., Lemma 3), BCL [18] (i.e., Lemma 2), Bak [14], BeCi [16], Bar [17] and BBM [20] (i.e., Lemma 9). To represent the aggregate result of the existing tests, let Sum denote a test which deems a given task set schedulable if at least one of the existing tests guarantees the test set to be schedulable. Note that Sum is the best schedulability result without our composition method. To show the schedulability improvement of individual schedulability tests by our composition theory, let GFB-comp and BBM-comp respectively denote Theorems 3 and 4, derived from Lemmas 3 (GFB) and 9 (BBM) using the composition theory. Finally, let Comp denote a test that composes all the existing schedulability tests by the composition theory in Theorem 1. Since there are exponential ways to apply the composition theory, we test only $\Phi_y(\tau_k)$ and $\Phi'_y(\tau_k)$ for $0 \le y \le m-1$, for each task τ_k. We will discuss the time-complexity issue in Section 7.

For evaluation, we use the same 100,000 implicit and constrained task sets for each m, presented in Section 4.2. Table 3 shows time-complexity and the overall schedulability performance—the number of schedulable task sets by each schedulability test. Fig. 4 shows average schedulability results for constrained deadline task sets for $m = 2$ and 4. Each figure consists of several plots, each showing the number of task sets proven schedulable by each schedulability test, with task set utilization (U_{sys}) in the interval $[U_{sys} - 0.01\cdot m, U_{sys} + 0.01\cdot m)$.

First, we show how much the composition theory can improve individual schedulability tests by deriving new schedulability tests. As shown in Table 3, and Figs. 4a and 4b, there is a significant difference between the number of schedulable task sets by an existing schedulability test GFB and its corresponding schedulability test GFB-comp derived using the composition theory. That is, GFB-comp finds 48.5 and 122.9 percent additional schedulable constrained deadline task sets which are deemed unschedulable by GFB for $m = 2$ and 4, respectively. In particular, GFB-comp, which has the lowest time-complexity among all schedulability tests of preemptive EDF as shown in Table 3, outperforms all low time-complexity tests GFB, BCL and Bak. Therefore, GFB-comp is useful when the time-complexity matters, e.g., adaptive systems with admission control.

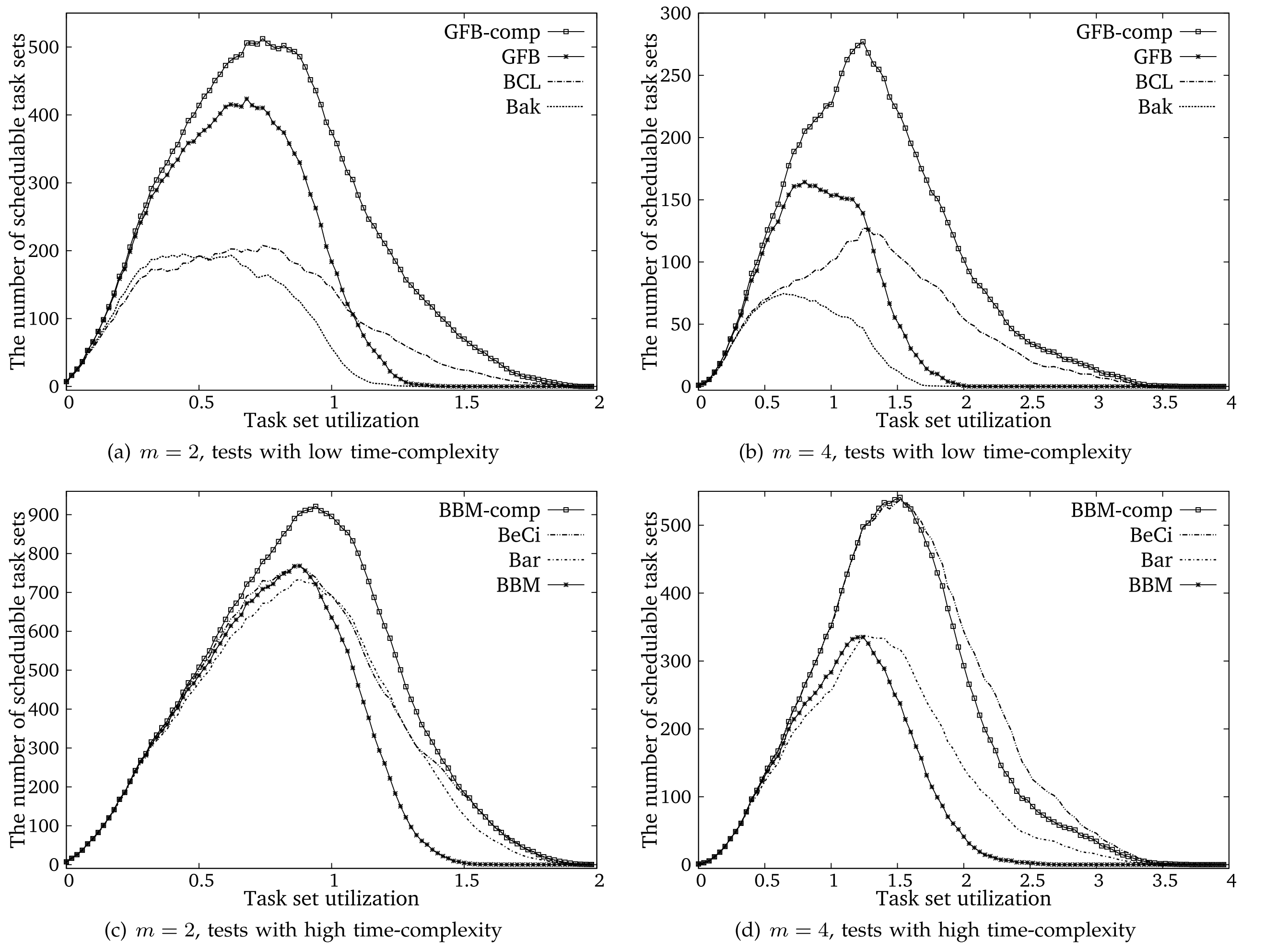
Such improvements are also seen under BBM-comp, which is derived from a schedulability test BBM using the composition theory. As shown in Table 3 and Figs. 4c and 4d, the improvements of BBM-comp over BBM are 40.4 and 104.1 percent, respectively, for $m = 2$ and 4 with constrained deadline task sets. Also, BBM-comp becomes the best ($m = 2$) or one of the best ($m = 4$) single schedulability tests as shown in the same figures and Table 3.

So far, we have confirmed that if the composition theory is applied to a given schedulability test, the newly-derived schedulability tests significantly improve average schedulability of task sets. For preemptive EDF, there are many existing schedulability tests as shown in Table 3. Therefore, we can improve schedulability more by applying the composition theory across existing schedulability tests (i.e.,

TABLE 3
The Number of Schedulable Task Sets by, Time-Complexity of, and Average Running Time of Schedulability Tests for Preemptive EDF (PP Means "Pseudo-Polynomial")

Implicit deadline task sets							
	GFB = Bak = BBM	GFB-comp = BBM-comp	BCL	BeCi	Bar	Sum	Comp
# of sched. sets							
$m = 2$	43944	52538	20999	47043	55278	59116	60661
$m = 4$	21938	30237	11528	32784	28945	36743	37359
$m = 8$	11703	18614	6261	23807	16243	25401	25636
Time-complexity	$O(\|\Phi\|)$	$O(\|\Phi\|)$	$O(\|\Phi\|^2)$	PP	PP	PP	PP
running time (ms)							
$m = 2$	0.02	0.02	0.02	0.03	0.44	0.36	0.33
$m = 4$	0.02	0.02	0.02	0.05	0.36	0.27	0.30
$m = 8$	0.03	0.03	0.03	0.14	0.41	0.28	0.50

Constrained deadline task sets										
	GFB	GFB-comp	BCL	Bak	BeCi	Bar	BBM	BBM-comp	Sum	Comp
# of sched. sets										
$m = 2$	15052	22359	9705	8741	34450	32583	28246	39510	40757	43781
$m = 4$	4153	9255	4633	2011	19674	11420	8900	18113	20559	22066
$m = 8$	1095	3878	2177	405	11948	3995	2744	8832	12108	12614
Time-complexity	$O(\|\Phi\|)$	$O(\|\Phi\|)$	$O(\|\Phi\|^2)$	$O(\|\Phi\|^3)$	PP	PP	PP	PP	PP	PP
running time (ms)										
$m = 2$	0.01	0.01	0.01	0.02	0.02	0.08	1.21	1.28	0.35	0.17
$m = 4$	0.02	0.02	0.02	0.02	0.04	0.06	1.23	1.37	0.60	0.25
$m = 8$	0.03	0.03	0.03	0.03	0.11	0.06	1.29	1.57	0.83	0.56



(a) $m = 2$, tests with low time-complexity (b) $m = 4$, tests with low time-complexity

(c) $m = 2$, tests with high time-complexity (d) $m = 4$, tests with high time-complexity

Fig. 4. Schedulability results of preemptive EDF schedulability tests for constrained deadline task sets.

Lemma 4), and such an improvement can be seen via Comp as stated below.

As shown in Table 3, BeCi is the best single schedulability test of existing schedulability tests (note that BBM-comp and GFB-comp are derived by the composition theory). If we simply aggregate the schedulability results from the existing tests, Sum finds 18.3 and 4.5 percent additional constrained deadline task sets, which are deemed unschedulable by the best single schedulability test for $m = 2$ and 4, respectively. When the composition theory is applied, the schedulability improvement of Comp over the best single schedulability test is more pronounced: 27.1 and 12.2 percent improvements with constrained deadline task sets for $m = 2$ and 4, respectively.

We also measured the average running time of each schedulability test on Intel Xeon CPU E31230 with 8.00 GB RAM, and summarized the results in Table 3 where Sum and Comp are shown not to take much time. That is, the average running time of Sum is strictly smaller than the simple sum of the running times of existing schedulability tests, because, if a task set is deemed schedulable by a schedulability test with low time-complexity, we need not run other schedulability tests with high time-complexity, for the test set. The same holds for Comp, when it checks per-task schedulability. Therefore, our compositional theory requires reasonable running times as shown in the table.

In summary, the composition theory developed in this paper effectively utilizes existing schedulability tests to not only derive new schedulability tests that outperform the corresponding existing tests without incurring additional time-complexity, but also find additional schedulable task sets that are not deemed schedulable by existing tests. Therefore, for both situations where time-complexity matters and does not matter, the composition theory improves schedulability guarantees.

6 Case Study III: Non-Preemptive EDF

While Sections 4 and 5 demonstrate the effectiveness of the composition theory for preemptive scheduling algorithms, this section deals with a popular non-preemptive scheduling algorithm, i.e., non-preemptive EDF.

6.1 Composition Theory to Non-Preemptive EDF Schedulability Tests

The non-preemptive EDF scheduling algorithm prioritizes jobs by their deadlines, and once a job starts its execution, it will run to completion without getting preempted by any other job. The following lemma presents a popular utilization-based schedulability test for non-preemptive EDF.

Lemma 10 (Theorem 1 in [25]). *A task set Φ is schedulable by non-preemptive EDF on an m-processor platform if Eq. (15) holds:*

$$\sum_{\tau_i\in\Phi} V_i \le m - (m-1)\cdot \max_{\tau_i\in\Phi} V_i. \qquad (15)$$

Then, similar to Lemmas 7 and 8, we can apply the composition theory to Lemma 10, without incurring exponential time-complexity, as stated in the following lemma.

Lemma 11. *Suppose the scheduler is non-preemptive EDF and the underlying schedulability test is that in Lemma 10. Then, applying Theorem 1 for a given task τ_k with all possible Φ^* for the test is equivalent to that only with every $\Phi''_y(\tau_k)$ for $0 \le y \le m-1$, where $\Phi''_y(\tau_k)$ is defined in Eq. (3).*

Proof. The proof is the same as Case (i) of the proof of Lemma 7 if we replace δ_i with V_i. □

Besides the efficient way to derive task-level schedulability results, we can derive new schedulability results from Lemma 6 using the composition theory as follows.

Theorem 5 (new non-preemptive EDF schedulability test). *A task set Φ is schedulable by non-preemptive EDF on an m-processor platform if $\max_{\tau_i\in\Phi} V_i \le 1$ holds and Eq. (16) holds:*

$$\sum_{\tau_i\in\Phi} V'_i \le m - (m-1)\cdot \max_{\tau_i\in\Phi} V_i, \qquad (16)$$

where V'_k

$$= \begin{cases} \min\big(V_k, 1-\max_{\tau_i\in\Phi} V_i\big), & \text{if } \tau_k \text{ belongs to tasks with} \\ & \text{m-1 largest } V_i \text{ in } \Phi - \{\tau_{max}\}, \\ V_k, & \text{otherwise,} \end{cases} \qquad (17)$$

Proof. We do not consider a task set with $\max_{\tau_i\in\Phi} V_i > 1$ since the task set trivially violates Eq. (15). Then, by replacing δ_i with V_i in Case (i) of the proof of Theorem 2, this follows. □

Note that the time-complexity of Theorem 5 is only $O(|\Phi|)$, which is the same as Lemma 10.

6.2 Evaluation of Schedulability Improvement

To evaluate the schedulability improvement by the composition theory, we test the same 100,000 implicit and constrained task sets for each m in Section 4.2. We evaluate all the existing non-preemptive EDF schedulability tests: Bar06 (Lemma 10) [25], GYG [22], LeAn [23] and LeSh [24]. Also, let Sum denote a test which guarantees the schedulability of a given task set if at least one of Bar06, GYG, LeAn and LeSh deems the task set schedulable. We also annotate the new test derived from Bar06 using the composition theory, as Bar06-comp, and let Comp denote a test that applies the composition theory to the all the existing schedulability tests. To address the time-complexity issue, we test only $\Phi_y(\tau_k)$, $\Phi'_y(\tau_k)$ and $\Phi''_y(\tau_k)$ for $0 \le y \le m-1$, for each task τ_k.

We summarize the time-complexity and the number of schedulable task sets in Table 4 and Fig. 5. First, when Bar06-comp is compared with Bar06, the former significantly improves the schedulability guarantees, without imposing additional time-complexity. Since Bar06 is the only existing low-time-complexity non-preemptive EDF schedulability test, Bar06-comp is very useful when online timing guarantees are required. Also, as in Section 5.2, Table 4 demonstrates that our composition theory for non-preemptive EDF also exhibits reasonable running times.

In addition to the improvement of a single schedulability test, the composition theory enables coverage of additional schedulable task sets. That is, compared to Sum, Comp finds up to 3.0 percent additional schedulable task sets by non-preemptive EDF.

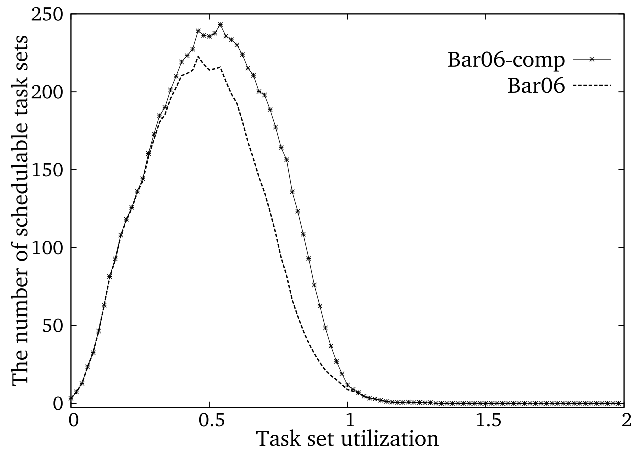
7 Conclusion and Discussion

In this paper, we have presented a novel composition theory for schedulability tests that consists of two composition methods. While we have demonstrated the application and impact of the theory with a limited number of examples, it can also be used for schedulability

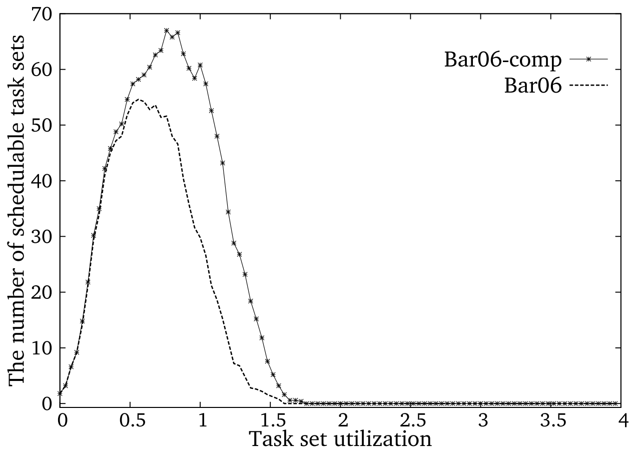
TABLE 4
The Number of Schedulable Task Sets by, Time-Complexity of, and Average Running Time of Schedulability Tests for Non-Preemptive EDF (PP Means 'Pseudo-Polynomial")

Implicit deadline task sets							
	Bar06	Bar06-comp	LeSh	GYG	LeAn	Sum	Comp
# of sched. sets							
$m = 2$	5970	7188	24281	19100	26226	27531	27680
$m = 4$	1080	1546	14524	8044	14041	15991	16281
$m = 8$	185	268	8457	4045	7613	9083	9353
Time-complexity	$O(\|\Phi\|)$	$O(\|\Phi\|)$	PP	PP	PP	PP	PP
running time (ms)							
$m = 2$	0.01	0.02	0.04	0.15	0.26	0.34	0.23
$m = 4$	0.02	0.02	0.08	0.23	0.84	1.06	0.71
$m = 8$	0.03	0.03	0.24	0.44	4.26	4.82	3.26

Constrained deadline task sets							
	Bar06	Bar06-comp	LeSh	GYG	LeAn	Sum	Comp
# of sched. sets							
$m = 2$	1253	1614	10852	9020	10766	11506	11570
$m = 4$	106	168	5653	4212	5117	5918	5987
$m = 8$	1	5	2929	1998	2343	3030	3076
Time-complexity	$O(\|\Phi\|)$	$O(\|\Phi\|)$	PP	PP	PP	PP	PP
running time (ms)							
$m = 2$	0.01	0.01	0.03	0.08	0.08	0.13	0.05
$m = 4$	0.02	0.02	0.06	0.12	0.18	0.30	0.12
$m = 8$	0.03	0.03	0.15	0.22	0.64	0.92	0.36



(a) $m = 2$, Bar06 vs. Bar06-comp



(b) $m = 4$, Bar06 vs. Bar06-comp

Fig. 5. Schedulability results of non-preemptive EDF schedulability tests for implicit deadline task sets.

tests of *any* global work-conserving regular multiprocessor scheduling algorithm.

Since a naive approach to applying Theorem 1 incurs exponential time-complexity, we addressed this issue by finding representative subsets that yield the same schedulability results as the exhaustive search (e.g., Lemmas 7, 8 and 11), or applying only some subsets at the expense of potential schedulability loss (e.g., Theorem 4). Our choice of subsets (i.e., $\Phi_y(\tau_k)$, $\Phi'_y(\tau_k)$ and $\Phi''_y(\tau_k)$) is shown to be effective in terms of finding additional schedulable task sets.

In future, we would like to develop a more general composition theory. We will consider re-use of the response time derived from different schedulability tests. This is implicitly used in the evaluation section of [35] without any proof. We would like to formalize the development of a response-time-based composition, and incorporate it in the two composition methods presented in this paper. We expect a unified composition theory with these three methods to be more effective than what we have presented in this paper.

ACKNOWLEDGMENTS

The work reported in this paper was supported in part by the US National Science Foundation under Grants CNS-0930813, CNS-1138200 and CNS-1329702. The work reported in this paper was also supported in part by National Research Foundation of Korea Grant funded by the Korean Government (Ministry of Education, Science and Technology) [NRF-2011-357-D00186]. This work was supported in part by BRL (2009-0086964), BSRP (2010-0006650, 2012-R1A1A1014930), NIPA (H0503-12-1041), KEIT (2011-10041313), DGIST CPS Global Center, and KIAT (M002300089) funded by the Korea Government (MEST/MKE). This work was also supported by Start-up Grant, NTU, Singapore. Jinkyu Lee is the corresponding author.

REFERENCES

[1] R.I. Davis and A. Burns, "A Survey of Hard Real-Time Scheduling for Multiprocessor Systems," *ACM Computing Surveys*, vol. 43, pp. 35:1-35:44, 2011.

[2] S. Baruah, N.K. Cohen, C.G. Plaxton, and D.A. Varvel, "Proportionate Progress: A Notion of Fairness in Resource Allocation," *Algorithmica*, vol. 15, no. 6, pp. 600-625, 1996.

[3] G. Levin, S. Funk, C. Sadowski, I. Pye, and S. Brandt, "DP-FAIR: A Simple Model for Understanding Optimal Multiprocessor Scheduling," *Proc. Euromicro Conf. Real-Time Systems (ECRTS)*, pp. 3-13, 2010.

[4] P. Regnier, G. Lima, E. Massa, G. Levin, and S. Brandt, "RUN: Optimal Multiprocessor Real-Time Scheduling via Reduction to Uniprocessor," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 104-115, 2011.

[5] C. Liu and J. Layland, "Scheduling Algorithms for Multi-Programming in a Hard-Real-Time Environment," *J. ACM*, vol. 20, no. 1, pp. 46-61, 1973.

[6] S.K. Baruah, "Optimal Utilization Bounds for the Fixed-Priority Scheduling of Periodic Task Systems on Identical Multiprocessors," *IEEE Trans. Computers*, vol. 53, no. 6, pp. 781-784, June 2004.

[7] K.H. Kim and M. Naghibzadeh, "Prevention of Task Overruns in Real-Time Non-Preemptive Multiprogramming Systems," *Proc. Int'l Symp. Computer Performance Modelling, Measurement and Evaluation*, pp. 267-276, 1980.

[8] A. Mok, "Fundamental Design Problems of Distributed Systems for the Hard-Real-Time Environment," PhD dissertation, Massachusetts Inst. of Technology, 1983.

[9] J. Leung and J. Whitehead, "On the Complexity of Fixed-Priority Scheduling of Periodic Real-Time Tasks," *Performance Evaluation*, vol. 2, pp. 237-250, 1982.

[10] A.Srinivasan, S. Baruah, "Deadline-Based Scheduling of Periodic Task Systems on Multiprocessors," *Information Processing Letters*, vol. 84, no. 2, pp. 93-98, 2002.

[11] S.K. Lee, "On-Line Multiprocessor Scheduling Algorithms for Real-Time Tasks," *Proc. IEEE Region 10's Ninth Ann. Int'l Conf.*, pp. 607-611, 1994.

[12] R.I. Davis and A. Burns, "FPZL Schedulability Analysis," *Proc. IEEE Real-Time Technology and Applications Symp. (RTAS)*, pp. 245-256, 2011.

[13] J.Y.-T. Leung, "A New Algorithm for Scheduling Periodic, Real-Time Tasks," *Algorithmica*, vol. 4, pp. 209-219, 1989.

[14] T. Baker, "An Analysis of EDF Schedulability on a Multiprocessor," *IEEE Trans. Parallel and Distributed Systems*, vol. 16, no. 8, pp. 760-768, Aug. 2005.

[15] M. Bertogna, M. Cirinei, and G. Lipari, "Improved Schedulability Analysis of EDF on Multiprocessor Platforms," *Proc. Euromicro Conf. Real-Time Systems (ECRTS)*, pp. 209-218, 2005.

[16] M. Bertogna and M. Cirinei, "Response-Time Analysis for Globally Scheduled Symmetric Multiprocessor Platforms," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 149-160, 2007.

[17] S. Baruah, "Techniques for Multiprocessor Global Schedulability Analysis," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 119-128, 2007.

[18] M. Bertogna, M. Cirinei, and G. Lipari, "Schedulability Analysis of Global Scheduling Algorithms on Multiprocessor Platforms," *IEEE Trans. Parallel and Distributed Systems*, vol. 20, no. 4, pp. 553-566, Apr. 2009.

[19] J. Goossens, S. Funk, and S. Baruah, "Priority-Driven Scheduling of Periodic Task Systems on Multiprocessors," *Real-Time Systems*, vol. 25, no. 2/3, pp. 187-205, 2003.

[20] S. Baruah, V. Bonifaci, A. Marchetti-Spaccamela, and S. Stiller, "Implementation of a Speedup-Optimal Global EDF Schedulability Test," *Proc. Euromicro Conf. Real-Time Systems (ECRTS)*, pp. 259-268, 2009.

[21] T.P. Baker and S. Baruah, "An Analysis of EDF Schedulability for Arbitrary Sporadic Task Systems," *Real-Time Systems*, vol. 43, pp. 3-24, 2009.

[22] N. Guan, W. Yi, Z. Gu, Q. Deng, and G. Yu, "New Schedulability Test Conditions for Non-Preemptive Scheduling on Multiprocessor Platforms," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 137-146, 2008.

[23] H. Leontyev and J.H. Anderson, "A Unified Hard/Soft Real-Time Schedulability Test for Global EDF Multiprocessor Scheduling," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 375-384, 2008.

[24] J. Lee and K.G. Shin, "Controlling Preemption for Better Schedulability in Multi-Core Systems," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 29-38, 2012.

[25] S. Baruah, "The Non-Preemptive Scheduling of Periodic Tasks Upon Multiprocessors," *Real-Time Systems*, vol. 32, no. 1, pp. 9-20, 2006.

[26] N. Guan, M. Stigge, W. Yi, and G. Yu, "New Response Time Bounds for Fixed Priority Multiprocessor Scheduling," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 387-397, 2009.

[27] M. Bertogna, M. Cirinei, and G. Lipari, "New Schedulability Tests for Real-Time Task Sets Scheduled by Deadline Monotonic on Multiprocessors," *Proc. Int'l Conf. Principles of Distributed Systems*, pp. 306-321, 2005.

[28] N. Guan, W. Yi, Q. Deng, Z. Gu, and G. Yu, "Schedulability Analysis for Non-Preemptive Fixed-Priority Multiprocessor Scheduling," *J. Systems Architecture*, vol. 57, no. 5, pp. 536-546, 2011.

[29] B. Andersson, S. Baruah, and J. Jonsson, "Static-Priority Scheduling on Multiprocessors," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 193-202, 2001.

[30] T.P. Baker, M. Cirinei, and M. Bertogna, "EDZL Scheduling Analysis," *Real-Time Systems*, vol. 40, pp. 264-289, 2008.

[31] J. Lee and I. Shin, "Demand-Based Schedulability Analysis for Real-Time Multi-Core Scheduling," *J. Systems and Software*, vol. 89, pp. 99-108, 2014.

[32] J. Lee and I. Shin–, "EDZL Schedulability Analysis in Real-Time Multi-Core Scheduling," *IEEE Trans. Software Eng.*, vol. 39, no. 7, pp. 910-916, July 2013.

[33] J. Lee, A. Easwaran, and I. Shin, "LLF Schedulability Analysis on Multiprocessor Platforms," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 25-36, 2010.

[34] J. Lee, A. Easwaran, and I. Shin, "Laxity Dynamics and LLF Schedulability Analysis on Multiprocessor Platforms," *Real-Time Systems*, vol. 48, no. 6, pp. 716-749, 2012.

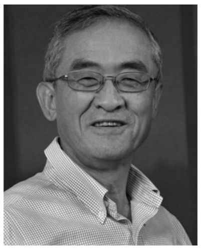
[35] M. Bertogna and S. Baruah, "Tests for Global EDF Schedulability Analysis," *J. Systems Architecture*, vol. 57, no. 5, pp. 487-497, 2011.

[36] T.P. Baker, "Comparison of Empirical Success Rates of Global versus Paritioned Fixed-Priority and EDF Scheduling for Hard Real-Time," Technical Report TR-050601, Florida State Univ., Dept. of CS, 2005.

[37] J. Lee, A. Easwaran, and I. Shin, "Maximizing Contention-Free Executions in Multiprocessor Scheduling," *Proc. IEEE Real-Time Technology and Applications Symp. (RTAS)*, pp. 235-244, 2011.

[38] T.P. Baker and M. Cirinei, "A Necessary and Sometimes Sufficient Condition for the Feasibility of Sets of Sporadic Hard-Deadline Tasks," *Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 178-190, 2006.

Jinkyu Lee is an assistant professor in Department of Computer Science and Engineering, Sungkyunkwan University, South Korea, where he joined in 2014. He received the BS, MS, and PhD degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2004, 2006, and 2011, respectively. He has been a research fellow/visiting scholar in the Department of Electrical Engineering and Computer Science, University of Michigan until 2014. His research interests include system design and analysis with timing guarantees, QoS support, and resource management in real-time embedded systems and cyber-physical systems. He won the best student paper award from the 17th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS) in 2011, and the Best Paper Award from the 33rd IEEE Real-Time Systems Symposium (RTSS) in 2012.



Kang G. Shin is the Kevin and Nancy O'Connor professor of computer science in the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor. His current research interests include QoS-sensitive computing and networking as well as on embedded real-time and cyber-physical systems. He has supervised the completion of 74 PhDs, and authored/coauthored more than 800 technical articles (more than 300 of these are in archival journals), a textbook and more than 20 patents or invention disclosures, and received numerous best paper awards, including the Best Paper Awards from the 2011 ACM International Conference on Mobile Computing and Networking (MobiCom11), the 2011 IEEE International Conference on Autonomic Computing, the 2010 and 2000 USENIX Annual Technical Conferences, as well as the 2003 IEEE Communications Society William R. Bennett Prize Paper Award, and the 1987 Outstanding IEEE Transactions of Automatic Control Paper Award. He has also received several institutional awards, including the Research Excellence Award in 1989, Outstanding Achievement Award in 1999, Distinguished Faculty Achievement Award in 2001, and Stephen Attwood Award in 2004 from The University of Michigan (the highest honor bestowed to Michigan Engineering faculty); a Distinguished Alumni Award of the College of Engineering, Seoul National University in 2002; 2003 IEEE RTC Technical Achievement Award; and 2006 Ho-Am Prize in Engineering (the highest honor bestowed to Korean-origin engineers). He was a cofounder of a couple of startups and also licensed some of his technologies to industry. He is a life fellow of the IEEE.

Insik Shin received the BS degree from Korea University, the MS degree from Stanford University, and the PhD degree from the University of Pennsylvania all in computer science in 1994, 1998, and 2006, respectively. He is currently an associate professor in the Department of Computer Science at KAIST, South Korea, where he joined in 2008. He has been a postdoctoral research fellow at Malardalen University, Sweden, and a visiting scholar at the University of Illinois, Urbana-Champaign until 2008. His research interests include cyber-physical systems and real-time embedded systems. He is currently a member of the Editorial Board of *Journal of Computing Science and Engineering*. He has been cochair of various workshops including satellite workshops of RTSS, CPSWeek, and RTCSA and has served various program committees in real-time embedded systems, including RTSS, RTAS, ECRTS, and EMSOFT. He received best paper awards, including Best Paper Awards from RTSS in 2003 and 2012, Best Student Paper Award from RTAS in 2011, and Best Paper runner-ups at ECRTS and RTSS in 2008. He is a member of the IEEE.

Arvind Easwaran is an assistant professor in the School of Computer Engineering at Nanyang Technological University (NTU), Singapore. He received MSc and PhD degrees from the University of Pennsylvania, USA, and a BE degree from University of Mumbai, India, all in Computer Science & Engineering. Prior to joining NTU in 2013, he has been an Invited Research Scientist at the Polytechnic Institute of Porto, Portugal, and an R & D Scientist at Honeywell Aerospace, USA. His research interests include Cyber-Physical Systems, Real-Time Systems, and Formal Methods. He has published in several leading conferences and journals in these areas, some of which are highly cited and have won awards.