Non-Preemptive Scheduling for Mixed-Criticality Real-Time Multiprocessor Systems

Hyeongboo Baek, Namyong Jung, Hoon Sung Chwa, *Member, IEEE*, Insik Shin, *Member, IEEE*, and Jinkyu Lee, *Member, IEEE*

Abstract—Real-time scheduling for Mixed-Criticality (MC) systems has received a growing attention as real-time embedded systems accommodate various tasks with different levels of criticality. While many studies have addressed how to guarantee timing requirements for MC systems with uniprocessor and multiprocessors, most of them have focused on supporting preemptive tasks. On the other hand, there have been few studies to address *non-preemptive scheduling* especially for MC multiprocessor platforms, in which the jobs under execution cannot be preempted by other jobs. In this paper, we develop schedulability tests for non-preemptive scheduling, which is the first attempt for MC multiprocessor systems. To this end, we first generalize an existing NP-EDF (Non-Preemptive Earliest Deadline First) schedulability test developed for single-criticality multiprocessor systems, towards for MC multiprocessor systems. For the generalization, we introduce new timing guarantee techniques for the system transition between two different criticalities, which is one of the key features in MC systems. We next extend the proposed NP-EDF schedulability test towards NP-EDFVD (NP-EDF with Virtual Deadlines) that is specialized for MC systems, and pose a virtual deadline assignment problem. We develop an optimal virtual deadline assignment policy using a control knob of the system-level deadline-reduction parameter and then a suboptimal one for the task-level parameter. Our simulation results demonstrate that the NP-EDFVD schedulability test with the proposed virtual deadline assignment policies finds a number of additional schedulable task sets, which are not schedulable by the NP-EDF schedulability test.

Index Terms—Real-time scheduling, schedulability analysis, mixed-criticality, non-preemptive tasks, real-time multiprocessor systems

1 Introduction

As real-time embedded systems become more complex, they need various functionalities often with different levels of criticality. This necessitates timing guarantees of mixed-criticality (MC) systems at different levels of assurance. Starting from Vestal [1], many studies have introduced scheduling algorithms and analysis tailored to MC systems, developing techniques that give timing guarantees in the presence of the system transition between different criticalities, with a focus on uniprocessor platforms [2], [3], [4], [5], [6], [7], [8], [9], [10]. To follow the popularity of the multi-core architecture, some of those studies have been extended to a multiprocessor platform [11], [12], [13], [14], [15], [16], [17]. However, such progress has been biased to *preemptive scheduling*, in which a higher-priority job can preempt a lower-priority job at any time.

On the other hand, in *non-preemptive scheduling*, a running job is executed till completion without being preempted. It must be used when the job is of inherently non-preemptive nature and/or it is subject to extremely large or unpredictable preemption/migration overhead (e.g., interrupts and transactional operations) [18]. Despite its necessity, there exist only a few studies that have addressed non-preemptive scheduling on MC systems [19], [20], [21], [22], [23], which target uniprocessor or distributed platforms. In particular, few studies have addressed timing guarantees for non-preemptive scheduling on MC multiprocessor systems.

In this paper, we aim at developing the first schedulability test for non-preemptive scheduling on MC multiprocessor systems. We target NP-EDF (Non-Preemptive Earliest Deadline First), one of the most fundamental, popular non-preemptive scheduling algorithms, and then extend the results to NP-EDFVD (NP-EDF with Virtual Deadlines) that is designed for MC systems. To this end, we investigate an existing schedulability test of NP-EDF for Single-Criticality (SC) systems (denoted by Bar [24]), addressing the following issues.

- Q1. How does Bar give timing guarantees under NP-EDF for SC systems?
- Q2. How can we generalize the timing guarantee techniques towards MC systems, dealing with the most important feature of MC systems, which is the system transition from the low to high criticality?

To address Q1 and Q2, we first carefully investigate how Bar operates. Bar considers two execution environments: 1)

- *H. Baek, N. Jung, and J. Lee is with the Department of Computer Science and Engineering, Sungkyunkwan University (SKKU), Suwon-si, Gyeonggi-do 16419, Republic of Korea. E-mail: {hbbaek359, goswlsqkqh1}@gmail.com, jinkyu.lee@skku.edu.*
- *H. S. Chwa is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109. E-mail: hchwa@umich.edu.*
- *I. Shin is with the School of Computing, KAIST, Daejeon 34051, Republic of Korea. E-mail: insik.shin@cs.kaist.ac.kr.*

Manuscript received 10 July 2017; revised 4 Jan. 2018; accepted 8 Feb. 2018. Date of publication 15 Feb. 2018; date of current version 13 July 2018. (Corresponding author: Jinkyu Lee.) Recommended for acceptance by B. Ravindran. 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/TPDS.2018.2806443

n tasks executing under NP-EDF scheduling on m processors and 2) n tasks executing on n imaginary fractional processors individually such that each task τ_i runs on a fractional processor n_i with a given constant rate V_i and finishes its execution no later than Y amount of time ahead of its deadline. Then, it calculates the total amount of execution of an entire task set until t for the former and until $t - X$ for the latter, respectively. Once we find a condition where the former has no smaller amount than the latter does, we can use the condition to establish timing guarantees. Building upon the investigation on Bar, we propose a method to determine the execution rate of V_i and two shifting values X and Y so as to accommodate the system transition between two different criticalities, yielding an NP-EDF schedulability test for MC multiprocessor systems.

Then, the next step is to extend the above results to an MC-aware scheduling algorithm, NP-EDFVD. EDFVD [4], [5] (originally developed for preemptive scheduling) is introduced as an extension of EDF that improves schedulability significantly by the use of different deadlines (i.e., virtual deadlines) in different criticalities. Hence, we seek to develop techniques for NP-EDFVD by answering the following issues.

Q3. How can we adapt the proposed NP-EDF schedulability test to NP-EDFVD?

Q4. Given that determining right virtual deadlines is critical to improving the schedulability, how can we assign the virtual deadline of every task so as to give timing guarantees?

As to Q3, we observe that the use of virtual deadlines introduces new properties. For example, under NP-EDF scheduling even for MC systems, the execution rate assigned to a task in the high-criticality mode is always no smaller than that to the same task in the low-criticality mode; this is because they share the same relative deadline but the worst-case execution time of a task in the high-criticality mode is no smaller than that in the low-criticality mode. However, the same cannot hold if we assign virtual deadlines to tasks in the low criticality mode. Therefore, we incorporate this property to the schedulability of NP-EDFVD when we determine the execution rate. When it comes to Q4, it is critical to schedulability to determine right virtual deadlines. We first consider a virtual deadline assignment problem using a control knob of the system-level deadline-reduction parameter, and develop an optimal assignment policy. Then, we pose a more general problem of assigning virtual deadlines using a task-level deadline-reduction parameter, and propose a suboptimal assignment policy.

We evaluate the schedulability performance of the proposed schedulability tests with/without our virtual deadline assignment policies. Our simulation results demonstrate that our schedulability test for NP-EDFVD with an optimal virtual deadline assignment policy using the system-level parameter significantly improves the schedulability over that for NP-EDF. Also, if we additionally apply our suboptimal virtual deadline assignment policy using the task-level parameter, we can find a number of additional schedulable task sets.

In summary, this paper makes the following contributions.

- We generalize an existing NP-EDF schedulability test originally developed for SC systems towards MC systems, which yields the first schedulability test for non-preemptive scheduling on MC multiprocessor systems;
- We extend the proposed NP-EDF schedulability test towards NP-EDFVD, and pose a virtual deadline assignment problem;
- We develop an optimal virtual deadline assignment policy using the system-level deadline-reduction parameter, and a suboptimal one using the task-level parameter; and
- We demonstrate the significant schedulability improvement of NP-EDFVD with the proposed virtual deadline assignment, over NP-EDF.

The rest of this paper is structured as follows. Section 2 explains our system model. Section 3 develops a new schedulability test of NP-EDF for MC multiprocessor systems. Section 4 extends the proposed schedulability test to NP-EDFVD, and develops (sub)optimal virtual deadline assignment policies. Section 5 evaluates the schedulability performance of the proposed schedulability tests with our virtual deadline assignment policies. Section 6 discusses related work, and Section 7 finally concludes this paper.

2 System Model

We consider an MC multiprocessor system with m identical processors. We consider a sporadic task model, and specify a task τ_i in a task set τ by 5-tuple, $\tau_i = (L_i, T_i, C_i^{\mathsf{LO}}, C_i^{\mathsf{HI}}, D_i)$, each of which is described as follows.

- $L_i \in \{\mathsf{HI}, \mathsf{LO}\}$ denotes the criticality level of a task τ_i; HI and LO present high- and low-criticality levels, respectively. A task τ_i with $L_i = \mathsf{HI}$, referred to as a HI-criticality task, should be certified by the certification authorities (CAs) while a task τ_i with $L_i = \mathsf{LO}$, referred to as a LO-criticality task, is not assumed to be certified by CAs.
- T_i is the minimum separation between release times of two consecutive jobs invoked by τ_i.
- C_i^{LO} and C_i^{HI} denote the worst-case execution times (WCETs) of τ_i required for the high- and low-criticality levels, respectively.
- D_i denotes the relative deadline of τ_i.

We assume that $C_i^{\mathsf{LO}} \leq C_i^{\mathsf{HI}} \leq D_i \leq T_i$ holds for every $\tau_i \in \tau$. Let n denote the number of tasks in τ. Using the above task parameters, each task τ_i invokes a series of jobs.

Scheduling algorithm. When it comes to scheduling algorithms, we consider global non-preemptive and work-conserving scheduling, in which a job of a task can be executed on a different core from another job of the task, a job cannot be preempted by any job during its execution, and a processor cannot be left idle if there is at least one job ready to execute.

System behaviors. We assume two different modes of a system run, each referred to as LO-mode and HI-mode. Every job of a task τ_i in LO-mode executes for no more than its C_i^{LO} while the job in HI-mode executes up to its C_i^{HI}; also, all LO-criticality tasks are not considered to be scheduled in HI-mode. The system starts at $t = 0$ and initially runs in LO-mode, and the system transition from LO-mode to HI-mode is triggered when any job's execution performed for more than C_i^{LO} (but not larger than C_i^{HI}) is observed. We

let t^{TR} denote the time instant when the system transition occurs. We assume that all jobs invoked by LO-criticality tasks are dropped immediately at t^{TR} even if they are executing; since we focus on non-preemptive scheduling, we may consider another model where the currently-executing jobs invoked by LO-criticality tasks are not dropped even in the presence of the system transition, which we can address if we extend this paper in the future. Note that both models make sense as discussed in [21].

Schedulability. We are interested in judging if a task set τ is schedulable by a scheduling algorithm on an MC multiprocessor platform, meaning that every job of every task $\tau_i \in \tau$ in LO-mode completes its execution (amounting to at most C_i^{LO}) within D_i time units, and every job of every HI-criticality task $\tau_i \in \tau | L_i = \mathsf{HI}$ in HI-mode completes its execution (amounting to at most C_i^{HI}) within D_i time units.

We now introduce some notations to be used in the later sections.

$$C_{max}^{\mathsf{LO}} \stackrel{\text{def.}}{=} \max_{\tau_i \in \tau} C_i^{\mathsf{LO}}, \qquad C_{max}^{\mathsf{HI}} \stackrel{\text{def.}}{=} \max_{\tau_i \in \tau | L_i = \mathsf{HI}} C_i^{\mathsf{HI}},$$

$$C_{max} \stackrel{\text{def.}}{=} \max(C_{max}^{\mathsf{LO}}, C_{max}^{\mathsf{HI}}),$$

$$V_i^{\mathsf{LO}} \stackrel{\text{def.}}{=} \lim_{\epsilon \to 0} \frac{C_i^{\mathsf{LO}}}{\max(\epsilon, D_i - C_{max}^{\mathsf{LO}})}, V_i^{\mathsf{HI}} \stackrel{\text{def.}}{=} \lim_{\epsilon \to 0} \frac{C_i^{\mathsf{HI}}}{\max(\epsilon, D_i - C_{max}^{\mathsf{HI}})},$$

$$V_{max}^{\mathsf{LO}} \stackrel{\text{def.}}{=} \max_{\tau_i \in \tau} V_i^{\mathsf{LO}}, \qquad V_{max}^{\mathsf{HI}} \stackrel{\text{def.}}{=} \max_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{HI}},$$

$$V_{sum}^{\mathsf{LO}} \stackrel{\text{def}}{=} \sum_{\tau_i \in \tau} V_i^{\mathsf{LO}}, \qquad V_{sum}^{\mathsf{HI}} \stackrel{\text{def.}}{=} \sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{HI}}.$$

3 Schedulability Analysis for NP-EDF on MC Multiprocessor Platforms

In this section, we develop an NP-EDF schedulability test for MC multiprocessor systems. To this end, we first present a schedulability condition for a case where the system is under LO-mode, using the existing NP-EDF schedulability test for SC multiprocessor systems. We then pose questions how to design schedulability conditions for MC systems, and derive important parameters for the conditions. Based on the design, we propose an NP-EDF schedulability test for MC multiprocessor systems, and prove its correctness.

3.1 Schedulability Condition Under LO-Mode by Recapitulation of Bar

As a first step towards an NP-EDF schedulability test for MC multiprocessor systems, we investigate Bar [24], an existing NP-EDF schedulability test for SC multiprocessor systems, and present how Bar can be used to derive a schedulability condition before the system transition.

Let $\mathsf{PS}^{\mathsf{LO}}(\tau)$ denote a process-sharing schedule such that a job of $\tau_i \in \tau$ executes with V_i^{LO} rate, thereby finishing its execution at C_{max}^{LO} amount of time ahead of its deadline (if $V_i^{\mathsf{LO}} \leq 1$). Also, let $\mathsf{NP\text{-}EDF}(\tau)$ denote a schedule such that jobs invoked by tasks in τ are scheduled by NP-EDF on an m-processor platform.

Let $W\big(\mathsf{A}(\tau), \delta, [t_a, t_b)\big)$ denote the amount of execution of δ in $[t_a, t_b)$, under a schedule $\mathsf{A}(\tau)$; the second parameter δ can be a single task, a single job, a set of tasks, or a set of jobs as long as it belongs to or is invoked by τ. Then, there is a relationship between the amount of execution by $\mathsf{NP\text{-}EDF}(\tau)$ and $\mathsf{PS}^{\mathsf{LO}}(\tau)$ as follows.

Lemma 1 (from Bar [24]). *Suppose τ satisfies Eq. (1).*

$$V_{sum}^{\mathsf{LO}} + (m-1) \cdot V_{max}^{\mathsf{LO}} \leq m. \tag{1}$$

Then, the following condition holds.

$$\begin{aligned} &W\big(\mathsf{NP\text{-}EDF}(\tau), J(\tau, q), [0, t + C_{max}^{\mathsf{LO}})\big) \\ &\geq W\big(\mathsf{PS}^{\mathsf{LO}}(\tau), J(\tau, q), [0, t)\big), \end{aligned} \tag{2}$$

where $J(\tau, q)$ denotes a set of q jobs invoked by tasks in τ with the q earliest absolute deadlines.

Proof. We prove it by contradiction, and the proof idea is similar to [24]. Suppose that t_0 denotes the first time instant in which Eq. (2) is violated. By the definition of t_0, there should be at least one job J_i of τ_i satisfying the following condition.

$$W\big(\mathsf{NP\text{-}EDF}(\tau), J_i, [0, t_0 + C_{max}^{\mathsf{LO}})\big) < W\big(\mathsf{PS}^{\mathsf{LO}}(\tau), J_i, [0, t_0)\big). \tag{3}$$

Let r_i be the release time of J_i. Since $r_i < t_0$ holds and t_0 is the first time instant violating Eq. (2), the following condition holds.

$$\begin{aligned} &W\big(\mathsf{NP\text{-}EDF}(\tau), J(\tau, q), [0, r_i + C_{max}^{\mathsf{LO}})\big) \\ &\geq W\big(\mathsf{PS}^{\mathsf{LO}}(\tau), J(\tau, q), [0, r_i)\big). \end{aligned} \tag{4}$$

Now, for J_i and $J(\tau, q)$, we compare the amount of execution under $\mathsf{NP\text{-}EDF}(\tau)$ in $[r_i + C_{max}^{\mathsf{LO}}, t_0 + C_{max}^{\mathsf{LO}})$, with that under $\mathsf{PS}^{\mathsf{LO}}$ in $[r_i, t_0)$. Let x and y respectively denote the cumulative length of intervals in $[r_i + C_{max}^{\mathsf{LO}}, t_0 + C_{max}^{\mathsf{LO}})$ where all processors are busy under $\mathsf{NP\text{-}EDF}(\tau)$, and that where at least one processor is idle under the schedule; therefore y equals to $t_0 - r_i - x$. Note that in $[r_i + C_{max}^{\mathsf{LO}}, t_0 + C_{max}^{\mathsf{LO}})$, no job in $J(\tau, q)$ under $\mathsf{NP\text{-}EDF}(\tau)$ is blocked or interfered by other jobs than $J(\tau, q)$ because every lower-priority job which starts its execution before r_i, finishes its execution before $r_i + C_{max}^{\mathsf{LO}}$ and jobs in $J(\tau, q)$ have a higher priority than other jobs by the definition of $J(\tau, q)$; this means that we only focus on $J(\tau, q)$ in $[r_i + C_{max}^{\mathsf{LO}}, t_0 + C_{max}^{\mathsf{LO}})$ under $\mathsf{NP\text{-}EDF}(\tau)$. We now present two properties.

- J_i under $\mathsf{NP\text{-}EDF}(\tau)$ does not complete its execution until $t_0 + C_{max}^{\mathsf{LO}}$ (otherwise it violates Eq. (3)), and thus executes for at least y amount of time since NP-EDF is work-conserving. Since J_i cannot execute for more than $(x+y) \cdot V_{max}^{\mathsf{LO}}$ under $\mathsf{PS}^{\mathsf{LO}}(\tau)$, Eqs. (3) and (4) for J_i yield the following condition:

$$y < (x+y) \cdot V_{max}^{\mathsf{LO}}. \tag{5}$$

- The amount of total execution of $J(\tau, q)$ under $\mathsf{NP\text{-}EDF}(\tau)$ is at least $m \cdot x + y$, and that under $\mathsf{PS}^{\mathsf{LO}}$ is at most $(x+y) \cdot \sum_{\tau_i \in \tau} V_i^{\mathsf{LO}}$. Eq. (3) for $J(\tau, q)$ and Eq. (4) derive the following condition:

$$m \cdot x + y < (x + y) \cdot V_{sum}^{\mathsf{LO}}. \quad (6)$$

By adding $(m-1)$ multiplied by Eqs. (5) to (6), we have

$$V_{sum}^{\mathsf{LO}} + (m-1) \cdot V_{max}^{\mathsf{LO}} > m, \quad (7)$$

which contradicts the supposition of this lemma. □

Using Lemma 1, we can guarantee schedulability by NP-EDF under MC multiprocessor systems as follows.

Lemma 2 (from Bar [24]). *Suppose that τ is scheduled by NP-EDF on an m processor platform and satisfies Eq. (1). Then, there is no job deadline miss until the system transition t^{TR}.*

Proof. We first check that the sum of execution rates under $\mathsf{PS}^{\mathsf{LO}}(\tau)$ is always no larger than m, since $V_{sum}^{\mathsf{LO}} \leq m$ holds by Eq. (1). Then, the proof is by induction of jobs generated by τ, sorted in the order of their absolute deadlines. For the base case with a single job of τ_i, NP-EDF trivially schedules it without its deadline miss since $C_i^{\mathsf{LO}} \leq D_i$ holds. Then, we assume that $q-1$ jobs with the $q-1$ earliest absolute deadlines meet their deadlines and prove that a job with the q^{th} earliest absolute deadline (denoted by J_i) also meets its deadline, for $q > 1$.

J_i under $\mathsf{PS}^{\mathsf{LO}}(\tau)$ completes its execution no later than $d_i - C_{max}^{\mathsf{LO}}$, where d_i denotes the deadline of J_i. By Eq. (2) of Lemma 1, this implies that J_i under $\mathsf{NP\text{-}EDF}(\tau)$ completes its execution no later than d_i, which proves the lemma. □

3.2 Design of Schedulability Conditions for MC Multiprocessor Systems

In the previous section, there are two important parameters for $\mathsf{PS}^{\mathsf{LO}}(\tau)$: (i) each job of τ_i executes with V_i^{LO} rate, and (ii) each job of τ_i finishes its execution at C_{max}^{LO} amount of time ahead of its deadline. Determining right values of the two important parameters makes it possible to give timing guarantees to non-preemptive jobs under NP-EDF before the system transition. For timing guarantee after the system transition, we need to properly determine the two parameters for a new process-sharing schedule that generalizes $\mathsf{PS}^{\mathsf{LO}}(\tau)$, stated as follows.

- How can we determine the execution rate of τ_i for HI-mode? (i.e., V_i^{TR})
- How can we determine the finishing time of a job of τ_i in HI-mode? (i.e., Y^{TR} amount of time ahead of its deadline)

In addition, we need to determine a right time instant when the new processor-sharing schedule switches the execution rate of τ_i from V_i^{LO} to V_i^{TR}, stated as follows.

- How can we determine the right time instant at which the execution rate is changed from LO- to HI-mode? (i.e., $t^{\mathsf{TR}} - X^{\mathsf{TR}}$, meaning X^{TR} amount of time ahead of the system transition)

Here, we would like to emphasize that the execution rate under a process-sharing schedule (corresponding to $\mathsf{PS}(\tau)$ for LO-mode) is changed *before* t^{TR}. In this section, we will detail how to calculate the three important values V_i^{TR}, Y^{TR} and X^{TR}.

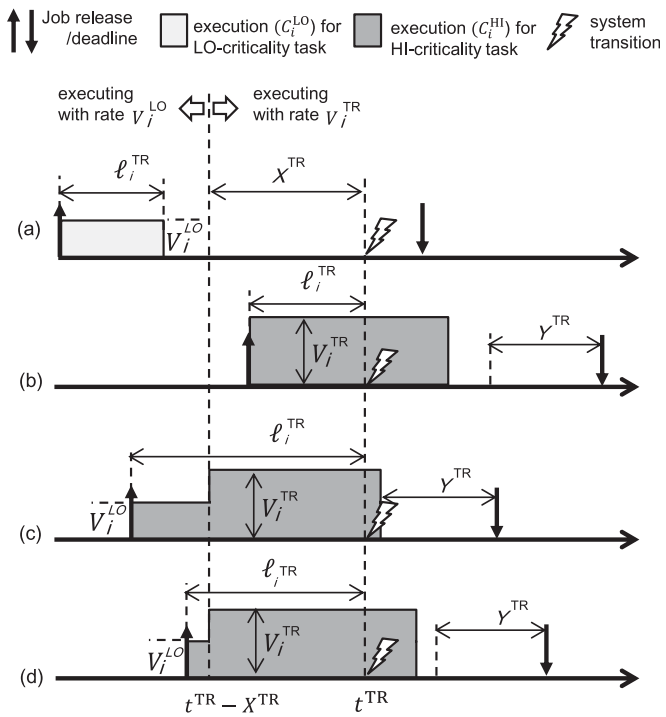


Fig. 1. How $\mathsf{PS}(\tau)$ works: a job of τ_i executing with (a) V_i^{LO} rate, (b) V_i^{TR} rate, and (c) and (d) V_i^{LO} and then V_i^{TR} rate.

With the three unknown variables V_i^{TR}, X^{TR} and Y^{TR} addressed in each issue, we now define a new process-sharing schedule. Let $\mathsf{PS}(\tau)$ denote a process-sharing schedule such that a job of $\tau_i \in \tau$ executes with V_i^{LO} rate before $t^{\mathsf{TR}} - X^{\mathsf{TR}}$, and with V_i^{TR} and zero rate after $t^{\mathsf{TR}} - X^{\mathsf{TR}}$ if $L_i = \mathsf{HI}$ and $L_i = \mathsf{LO}$, respectively. We let the actual execution time of each job under $\mathsf{PS}(\tau)$ be the same as that under $\mathsf{NP\text{-}EDF}(\tau)$. Under $\mathsf{PS}(\tau)$, a job of $\tau_i \in \tau$ finishes its execution no later than C_{max}^{LO} amount of time ahead of its deadline, if it finishes its execution earlier than t^{TR} under $\mathsf{NP\text{-}EDF}(\tau)$. On the other hand, a job of $\tau_i \in \tau$ with $L_i = \mathsf{HI}$ finishes its execution no later than Y^{TR} amount of time ahead of its deadline under $\mathsf{PS}(\tau)$, if it finishes its execution no earlier than t^{TR} under $\mathsf{NP\text{-}EDF}(\tau)$.

Fig. 1 presents how $\mathsf{PS}(\tau)$ works for a job of τ_i, with four examples with different release times. A job can execute with (i) V_i^{LO} rate in Fig. 1a, (ii) V_i^{TR} rate in Fig. 1b, and (iii) V_i^{LO} and then V_i^{HI} rate in Figs. 1c and 1d.

Then, we would like to prove a condition for addressing the amount of execution of $\mathsf{PS}(\tau)$ compared to that of $\mathsf{NP\text{-}EDF}(\tau)$, which corresponds to Eq. (2), as follows:

$$W\big(\mathsf{NP\text{-}EDF}(\tau), J(\tau,q), [0, t+Y^{\mathsf{TR}})\big) \geq W\big(\mathsf{PS}(\tau), J(\tau,q), [0,t)\big). \quad (8)$$

Recall that $J(\tau,q)$ denotes a set of q jobs invoked by tasks in τ with the q earliest absolute deadlines.

For addressing the execution rate of $\mathsf{PS}(\tau)$ in LO- and HI-mode, we consider task sets satisfying Eq. (1) and the following condition, respectively:

$$V_{sum}^{\mathsf{TR}} + (m-1) \cdot V_{max}^{\mathsf{TR}} \leq m, \quad (9)$$

where $V_{sum}^{\mathsf{TR}} = \sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{TR}}$ and $V_{max}^{\mathsf{TR}} = \max_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{TR}}$.

Then, the remaining issues are to determine the unknown variables V_i^{TR}, X^{TR} and Y^{TR} such that we can prove that if Eqs. (1) and (9) hold for τ, τ is schedulable by NP-EDF on MC multiprocessor systems. We now discuss qualification of the unknown variables, starting from V_i^{TR}.

We consider a job of interest J_i invoked by τ_i, such that the system transition occurs in the middle of execution of J_i or before J_i's execution (therefore J_i executes C_i^{HI} amount). Let ℓ_i^{TR} denote an interval length from its release time to the earlier time instant between t^{TR} and the time instant when J_i finishes its execution under $\mathsf{PS}(\tau)$; if J_i's release time is after t^{TR}, ℓ_i^{TR} is zero. Then, the amount of execution of J_i under $\mathsf{PS}(\tau)$ is calculated by

$$V_i^{\mathsf{LO}} \cdot \max(0, \ell_i^{\mathsf{TR}} - X^{\mathsf{TR}}) + V_i^{\mathsf{TR}} \cdot \max\big(0, D_i - Y^{\mathsf{TR}} - \max(0, \ell_i^{\mathsf{TR}} - X^{\mathsf{TR}})\big), \tag{10}$$

which should be equal to C_i^{HI} for J_i to finish its execution. That is, as shown in Figs. 1c and 1d, under $\mathsf{PS}(\tau)$, a job of τ_i executes with V_i^{LO} rate from its release time to $t^{\mathsf{TR}} - X^{\mathsf{TR}}$ (whose length is $\max(0, \ell_i^{\mathsf{TR}} - X^{\mathsf{TR}})$), while the job executes with V_i^{TR} rate from $t^{\mathsf{TR}} - X^{\mathsf{TR}}$ to Y^{TR} amount of time ahead of its deadline (whose length is $D_i - Y^{\mathsf{TR}} - \max(0, \ell_i^{\mathsf{TR}} - X^{\mathsf{TR}})$). This derives V_i^{TR} and its properties as follows:

Lemma 3. *V_i^{TR} is calculated by Eq. (11), and V_i^{TR} decreases (or stays) as X^{TR} increases and Y^{TR} and ℓ_i^{TR} decrease.*

$$V_i^{\mathsf{TR}} = \lim_{\epsilon \to 0} \frac{C_i^{\mathsf{HI}} - V_i^{\mathsf{LO}} \cdot \max(\epsilon, \ell_i^{\mathsf{TR}} - X^{\mathsf{TR}})}{\max\big(\epsilon, D_i - Y^{\mathsf{TR}} - \max(\epsilon, \ell_i^{\mathsf{TR}} - X^{\mathsf{TR}})\big)}. \tag{11}$$

Proof. From the condition where Eq. (10) is equal to C_i^{HI}, Eq. (11) is immediately derived.

In Eq. (11), it is straightforward that as Y^{TR} increases, V_i^{TR} increases. Also, if we increase X^{TR} or decrease ℓ_i^{TR}, we can reduce (or do not change) the time interval when a job of τ_i executes with V_i^{LO} rate, yielding a decrease in V_i^{TR}, e.g., compare Figs. 1c and 1d. □

Next, the following lemma discusses the qualification of X^{TR}.

Lemma 4. $X^{\mathsf{TR}} \leq C_{max}^{\mathsf{LO}}$.

Proof. Suppose that X^{TR} is strictly larger than C_{max}^{LO}. Then, we cannot guarantee the schedulability of a job of τ_i with $L_i = \mathsf{LO}$ whose deadline is no later than t^{TR}. This is because the amount of execution of a job of τ_i whose deadline is later than $t^{\mathsf{TR}} - X^{\mathsf{TR}} + C_{max}^{\mathsf{LO}}$ (but no later than t^{TR}) under $\mathsf{PS}(\tau)$ is strictly less than C_i^{LO}, which makes it impossible to say that if Eq. (2) holds, all the jobs are schedulable (as we do that in Lemma 2). □

When it comes to Y^{TR}, we need to address the blocking by lower-priority jobs in both LO- and HI-modes. For the former and the latter, we need at least C_{max}^{LO} and at least C_{max}^{HI}, respectively; therefore, Y^{TR} should be at least C_{max}. Considering Lemma 3, we set Y^{TR} to the smallest possible value C_{max}.

As to ℓ_i^{TR}, the value depends on each job's release pattern and the system transition instant, implying that we cannot control it. Therefore, we need to calculate an upper-bound of the value in order to upper-bound V_i^{TR}. Once we target a task set satisfying Eq. (1), we can use the property of Eq. (2) and calculate an upper bound of the amount of execution of jobs whose priority is higher than the job of interest. This allows to calculate the maximum interval between the release and finishing time of the job of interest, which is also an upper-bound of ℓ_i^{TR}. The following lemma records this.

Lemma 5. *Suppose that τ satisfies Eq. (1). Then, ℓ_i^{TR} is upper-bounded by R_i^{LO} where*

$$R_i^{\mathsf{LO}} = C_i^{\mathsf{LO}} + C_{max}^{\mathsf{LO}} + \lim_{\epsilon \to 0} \max\big(\epsilon, D_i - C_{max}^{\mathsf{LO}}\big) \cdot \frac{\sum_{\tau_j \in \tau \setminus \{\tau_i\}} V_j^{\mathsf{LO}}}{m}. \tag{12}$$

Proof. We now prove $r_i + \ell_i^{\mathsf{TR}} \leq f_i \leq r_i + R_i^{\mathsf{LO}}$, where r_i and f_i respectively denote the release time of J_i and the time instant when J_i finishes its execution of C_i^{LO} under $\mathsf{NP\text{-}EDF}(\tau)$. Recall that we focus on a job of interest of J_i invoked by τ_i, such that the system transition occurs in the middle of execution of J_i or before J_i's execution (therefore J_i executes C_i^{HI} amount). We also recall that ℓ_i^{TR} denotes an interval length from r_i to the earlier time instant between t^{TR} and the time instant when J_i finishes its execution under $\mathsf{PS}(\tau)$. Once we prove the two inequalities, we can derive $\ell_i^{\mathsf{TR}} \leq R_i^{\mathsf{LO}}$. From now on, we prove the two inequalities separately.

First, we prove the first inequality $r_i + \ell_i^{\mathsf{TR}} \leq f_i$. Suppose that the inequality does not hold. By definition of ℓ_i^{TR}, the system transition occurs no earlier than $r_i + \ell_i^{\mathsf{TR}}$. Therefore, $r_i + \ell_i^{\mathsf{TR}} > f_i$ implies that the system transition occurs after J_i finishes C_i^{LO} amount of execution, which contradicts the fact that the system transition occurs in the middle of execution of J_i or before J_i's execution (which is the definition of J_i).

Second, we prove the second inequality $f_i \leq r_i + R_i^{\mathsf{LO}}$. We consider two cases: (i) J_i under $\mathsf{NP\text{-}EDF}(\tau)$ starts its execution before $r_i + C_{max}^{\mathsf{LO}}$, and (ii) otherwise. Case (i) implies that f_i is no later than $r_i + C_{max}^{\mathsf{LO}} + C_i^{\mathsf{LO}}$. Since the latter is no later than $r_i + R_i^{\mathsf{LO}}$ by Eq. (12), the inequality holds for Case (i). When it comes to Case (ii), since every job released before r_i finishes its execution before $r_i + C_{max}^{\mathsf{LO}}$, we now calculate the amount of execution of every job whose priority is higher than J_i in $[r_i + C_{max}^{\mathsf{LO}}, r_i + D_i]$ in which any lower-priority job cannot block J_i. If the amount is no larger than $\lim_{\epsilon \to 0} \max\big(\epsilon, D_i - C_{max}^{\mathsf{LO}}\big) \cdot \sum_{\tau_j \in \tau \setminus \{\tau_i\}} V_j^{\mathsf{LO}}$, the lemma holds, which is true as follows.

Let $r_i + x_j$ denote the earliest deadline of a job of τ_j ($\neq \tau_i$), after r_i. If $x_j > D_i$, no job of τ_j has a higher priority than J_i in $[r_i + C_{max}^{\mathsf{LO}}, r_i + D_i)$. Otherwise, we consider two intervals: $[r_i + C_{max}^{\mathsf{LO}}, r_i + x_j)$ and $[r_i + x_j, r_i + D_i)$; note that the first interval may not exist if $x_j \leq C_{max}^{\mathsf{LO}}$. In $[r_i + C_{max}^{\mathsf{LO}}, r_i + x_j)$, a job of τ_j has a higher priority than J_i, and the amount of its execution is $W\big(\mathsf{NP\text{-}EDF}(\tau), \tau_j, [0, r_i + \ell_i^{\mathsf{TR}})\big) - W\big(\mathsf{NP\text{-}EDF}(\tau), \tau_j, [0, r_i + C_{max}^{\mathsf{LO}})\big)$, which is the same as $W\big(\mathsf{PS}(\tau), \tau_j, [0, r_i + \ell_i^{\mathsf{TR}} - C_{max}^{\mathsf{LO}})\big) - W\big(\mathsf{NP\text{-}EDF}(\tau), \tau_j, [0, r_i + C_{max}^{\mathsf{LO}})\big)$, because a job under

$\mathsf{PS}(\tau)$ finishes its execution no later than C_{max}^{LO} amount of time ahead of its deadline. The amount of execution of jobs of τ_j with a higher priority than J_i is at most $\lfloor \frac{D_i - x_j}{T_j} \rfloor \cdot C_j$. Then, the amount of execution of jobs whose priority is higher than J_i in $[r_i + C_{max}^{\mathsf{LO}}, r_i + D_i)$ is calculated by

$$\begin{aligned} &\sum_{\tau_j \in \tau \setminus \{\tau_i\}} W\big(\mathsf{PS}(\tau), \tau_j, [0, r_i + x_j - C_{max}^{\mathsf{LO}})\big) \\ &\quad - \sum_{\tau_j \in \tau \setminus \{\tau_i\}} W\big(\mathsf{NP\text{-}EDF}(\tau), \tau_j, [0, r_i + C_{max}^{\mathsf{LO}})\big) \\ &\quad + \sum_{\tau_j \in \tau \setminus \{\tau_i\}} \Big\lfloor \frac{D_i - x_j}{T_j} \Big\rfloor \cdot C_j. \end{aligned} \tag{13}$$

Since J_i cannot execute until $r_i + C_{max}^{\mathsf{LO}}$, $W\big(\mathsf{NP\text{-}EDF}(\tau), \tau, [0, r_i + C_{max}^{\mathsf{LO}})\big) \geq W\big(\mathsf{PS}(\tau), \tau, [0, r_i)\big)$ implies that $W\big(\mathsf{NP\text{-}EDF}(\tau), \tau \setminus \{\tau_i\}, [0, r_i + C_{max}^{\mathsf{LO}})\big) \geq W\big(\mathsf{PS}(\tau), \tau \setminus \{\tau_i\}, [0, r_i)\big)$, which derives the following condition from Eq. (13).

$$\begin{aligned} \text{Eq. (13)} \leq &\sum_{\tau_j \in \tau \setminus \{\tau_i\}} W\big(\mathsf{PS}(\tau), \tau_j, [0, r_i + x_j - C_{max}^{\mathsf{LO}})\big) \\ &- \sum_{\tau_j \in \tau \setminus \{\tau_i\}} W\big(\mathsf{PS}(\tau), \tau_j, [0, r_i)\big) \\ &+ \sum_{\tau_j \in \tau \setminus \{\tau_i\}} \Big\lfloor \frac{D_i - x_j}{T_j} \Big\rfloor \cdot C_j^{\mathsf{LO}} \\ \leq &\sum_{\tau_j \in \tau \setminus \{\tau_i\}} W\big(\mathsf{PS}(\tau), \tau_j, [r_i, r_i + x_j - C_{max}^{\mathsf{LO}})\big) \\ &+ \sum_{\tau_j \in \tau \setminus \{\tau_i\}} \Big(\frac{D_i - x_j}{D_j - C_{max}^{\mathsf{LO}}} \Big) \cdot C_j^{\mathsf{LO}} \\ \leq &\sum_{\tau_j \in \tau \setminus \{\tau_i\}} (x_j - C_{max}^{\mathsf{LO}}) \cdot \Big(\frac{C_j^{\mathsf{LO}}}{D_j - C_{max}^{\mathsf{LO}}} \Big) \\ &+ \sum_{\tau_j \in \tau \setminus \{\tau_i\}} \Big(\frac{D_i - x_j}{D_j - C_{max}^{\mathsf{LO}}} \Big) \cdot C_j^{\mathsf{LO}} \\ = &(D_i - C_{max}^{\mathsf{LO}}) \cdot \sum_{\tau_j \in \tau \setminus \{\tau_i\}} V_i^{\mathsf{LO}}. \end{aligned}$$

Therefore, $f_i \leq r_i + R_i^{\mathsf{LO}}$ for Case (ii) holds.

Finally, $r_i + \ell_i^{\mathsf{TR}} \leq f_i \leq r_i + R_i^{\mathsf{LO}}$ holds, implying $\ell_i^{\mathsf{TR}} \leq R_i^{\mathsf{LO}}$. This proves the lemma. □

Finally, if we combine all the results from Lemmas 3, 4 and 5 and apply $X^{\mathsf{TR}} = C_{max}^{\mathsf{LO}}$, $Y^{\mathsf{TR}} = C_{max}$, and $\ell_i^{\mathsf{TR}} = R_i^{\mathsf{LO}}$, we have the following V_i^{TR} that yields a tighter schedulability condition.

$$V_i^{\mathsf{TR}} = \lim_{\epsilon \to 0} \frac{C_i^{\mathsf{HI}} - V_i^{\mathsf{LO}} \cdot \max(\epsilon, R_i^{\mathsf{LO}} - C_{max}^{\mathsf{LO}})}{\max\big(\epsilon, D_i - C_{max} - \max(\epsilon, R_i^{\mathsf{LO}} - C_{max}^{\mathsf{LO}})\big)}. \tag{14}$$

3.3 Final Schedulability Test for NP-EDF

So far, we determined the unknown variables V_i^{TR}, X^{TR} and Y^{TR} so as to make a correct and tight schedulability condition in the presence of the system transition. In this section, we formally present and prove our NP-EDF schedulability test for MC multiprocessor platforms, starting from the following lemma that corresponds to Lemma 1.

Lemma 6. *Suppose that τ satisfies Eq. (1) as well as Eq. (9) with V_i^{TR} in Eq. (14). Then, Eq. (8) with $Y^{\mathsf{TR}} = C_{max}$ holds for all $t > 0$.*

Proof. We follow the basic proof idea of Lemma 1, using the contradiction. Suppose that t_0 denotes the first time instant in which Eq. (8) is violated. By the definition of t_0, there should be at least one job J_i of τ_i satisfying the following condition.

$$W\big(\mathsf{NP\text{-}EDF}(\tau), J_i, [0, t_0 + C_{max})\big) < W\big(\mathsf{PS}(\tau), J_i, [0, t_0)\big). \tag{15}$$

Let r_i be the release time of J_i. Since $r_i < t_0$ holds and t_0 is the first time instant violating Eq. (8), the following condition holds.

$$\begin{aligned} &W\big(\mathsf{NP\text{-}EDF}(\tau), J(\tau, q), [0, r_i + C_{max})\big) \\ &\geq W\big(\mathsf{PS}(\tau), J(\tau, q), [0, r_i)\big). \end{aligned} \tag{16}$$

Depending on the time instants t^{TR}, r_i and t_0, we consider three cases: (i) $t^{\mathsf{TR}} - C_{max}^{\mathsf{LO}} \geq t_0$, (ii) $t^{\mathsf{TR}} - C_{max}^{\mathsf{LO}} \in [r_i, t_0)$, and (iii) $t^{\mathsf{TR}} - C_{max}^{\mathsf{LO}} < r_i$. While the execution rate of τ_i under $\mathsf{PS}(\tau)$ in $[r_i, t_0)$ is V_i^{LO} and V_i^{TR}, respectively for (i) and (iii), (ii) exhibits both execution rates.

Now, for J_i and $J(\tau, q)$, we compare the amount of execution under $\mathsf{NP\text{-}EDF}(\tau)$ in $[r_i + C_{max}, t_0 + C_{max})$, with that under PS in $[r_i, t_0)$. Let x and y denote the cumulative length of intervals in $[r_i + C_{max}, t_0 + C_{max})$ where all processors are busy under $\mathsf{NP\text{-}EDF}(\tau)$, and that where at least one processor is idle under the schedule; therefore y equals to $t_0 - r_i - x$. Note that in $[r_i + C_{max}, t_0 + C_{max})$, no job in $J(\tau, q)$ under $\mathsf{NP\text{-}EDF}(\tau)$ is blocked or interfered by other jobs than $J(\tau, q)$ because every lower-priority job which starts its execution before r_i finishes its execution before $r_i + C_{max}$ and jobs in $J(\tau, q)$ have a higher priority than other jobs by the definition of $J(\tau, q)$; this means that we only focus on $J(\tau, q)$ in $[r_i + C_{max}, t_0 + C_{max})$ under $\mathsf{NP\text{-}EDF}(\tau)$. We now present two properties.

- J_i under $\mathsf{NP\text{-}EDF}(\tau)$ does not complete its execution until $t_0 + C_{max}$ (otherwise it violates Eq. (15)), and thus executes for at least y time units since NP-EDF is work-conserving. Since J_i cannot execute for more than $\max(0, \min(t^{\mathsf{TR}} - C_{max}^{\mathsf{LO}}, t_0) - r_i) \cdot V_{max}^{\mathsf{LO}} + \max(0, t_0 - \max(t^{\mathsf{TR}} - C_{max}^{\mathsf{LO}}, r_i)) \cdot V_{max}^{\mathsf{TR}}$ under $\mathsf{PS}(\tau)$, Eqs. (15) and (16) for J_i yield the following condition:[1]

$$\begin{aligned} y < &\max(0, \min(t^{\mathsf{TR}} - C_{max}^{\mathsf{LO}}, t_0) - r_i) \cdot V_{max}^{\mathsf{LO}} \\ &+ \max(0, t_0 - \max(t^{\mathsf{TR}} - C_{max}^{\mathsf{LO}}, r_i)) \cdot V_{max}^{\mathsf{TR}}. \end{aligned} \tag{17}$$

- The amount of total execution of $J(\tau, q)$ under $\mathsf{NP\text{-}EDF}(\tau)$ is at least $m \cdot x + y$, and that under

1. The max and min functions are needed to address Cases (i), (ii) and (iii).

$\mathsf{PS}^{\mathsf{LO}}$ is at most $\max(0, \min(t^{\mathsf{TR}} - C^{\mathsf{LO}}_{max}, t_0) - r_i) \cdot V^{\mathsf{LO}}_{sum} + \max(0, t_0 - \max(t^{\mathsf{TR}} - C^{\mathsf{LO}}_{max}, r_i)) \cdot V^{\mathsf{TR}}_{sum}$ under $\mathsf{PS}(\tau)$. Eq. (15) for $J(\tau, q)$ and Eq. (16) derive the following condition:

$$m \cdot x + y < \max(0, \min(t^{\mathsf{TR}} - C^{\mathsf{LO}}_{max}, t_0) - r_i) \cdot V^{\mathsf{LO}}_{sum} + \max(0, t_0 - \max(t^{\mathsf{TR}} - C^{\mathsf{LO}}_{max}, r_i)) \cdot V^{\mathsf{TR}}_{sum}. \quad (18)$$

By adding $(m-1)$ multiplied by Eqs. (17) to (18), we can show the contradiction of Eq. (1) for Case (i) and the contradiction of Eq. (9) with V^{TR}_i in Eq. (14) for Case (iii). Also, if we further remove Eq. (1) multiplied by $\max(0, \min(t^{\mathsf{TR}} - C^{\mathsf{LO}}_{max}, t_0) - r_i)$ from adding $(m-1)$ multiplied by Eqs. (17) to (18), we can also show the contradiction of Eq. (9) with V^{TR}_i for Case (ii). Since all the three cases yield contradiction, the lemma holds. □

Using the lemma, we finally present our schedulability test.

Theorem 1. *Suppose that τ satisfies Eq. (1) as well as Eq. (9) with V^{TR}_i in Eq. (14). Then, τ is schedulable by NP-EDF under MC multiprocessor systems.*

Proof. Once Lemma 6 derives that Eq. (8) with $Y^{\mathsf{TR}} = C_{max}$ holds for all $t > 0$, the proof of this theorem is the same as that of Lemma 2. □

We can simply calculate the time-complexity of the proposed NP-EDF schedulability test for MC multiprocessor systems in Theorem 1, which is $O(n)$.

4 Virtual Deadline Assignment for NP-EDFVD

In this section, we adapt the proposed schedulability test for NP-EDF to NP-EDFVD, and pose a virtual deadline assignment problem. We then present an optimal virtual deadline assignment policy for the system-level deadline reduction parameter, and a suboptimal one for the task-level parameter. Note that NP-EDFVD can improve schedulability of NP-EDF by adjusting the deadline of each task τ_i with $L_i = \mathsf{HI}$ in LO-mode as the same concept works in preemptive scheduling [4], [5], [14], [15]. This requires development of principles how execution demand for HI-mode (including the system transition) and LO-mode vary with virtual deadlines, which is a matter of this section.

4.1 Schedulability Analysis for NP-EDFVD on MC Multiprocessor Platforms

In this paper, we consider NP-EDFVD, and shorten the relative deadline of τ_i with $L_i = \mathsf{HI}$ in LO-mode, from D_i to a new relative deadline D^{LO}_i. Since the new relative deadline of τ_i (i.e., D^{LO}_i) can be in $[C^{\mathsf{LO}}_{max}, D_i]$ as seen in the denominator of V_i, we let α_i for τ_i to adjust the value of D^{LO}_i from C_{max} (by $\alpha_i = 0$) to D_i (by $\alpha_i = 1$). That is, we replace the term of $\max(0, D_i - C^{\mathsf{LO}}_{max})$ with $\max(0, D_i - C^{\mathsf{LO}}_{max}) \cdot \alpha_i$. Then, once we determine α_i, D^{LO}_i can be calculated by $C^{\mathsf{LO}}_{max} + (D_i - C^{\mathsf{LO}}_{max}) \cdot \alpha_i$; therefore, this section discusses how to determine α_i for every $\tau_i \in \tau | L_i = \mathsf{HI}$, assuming α_i is set to 1 and never changed for every $\tau_i \in \tau | L_i = \mathsf{LO}$.

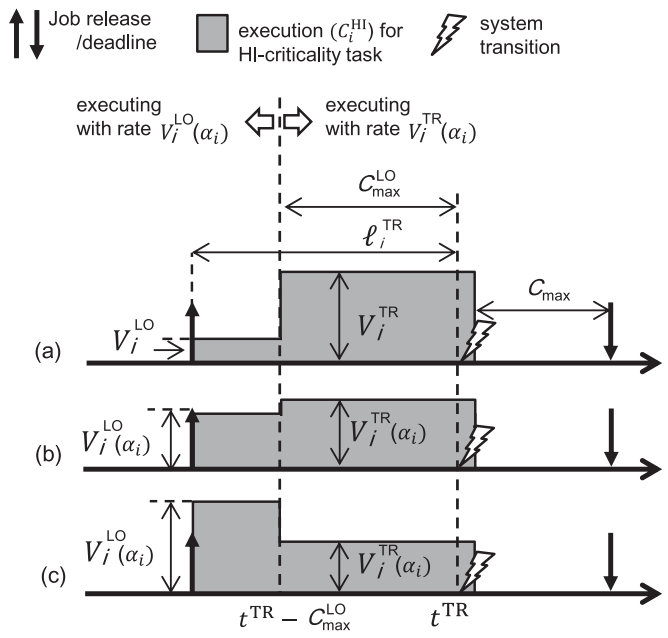


Fig. 2. How $\mathsf{PS}(\tau)$ works with and without virtual deadlines: (a) $\alpha_i = 1$ (no virtual deadline); (b) small α_i; and (c) smaller α_i.

Using α_i, V^{LO}_i and R^{LO}_i for NP-EDF can be expressed by $V^{\mathsf{LO}}_i(\alpha_i)$ and $R^{\mathsf{LO}}_i(\alpha_i)$ for NP-EDFVD, as follows.

$$V^{\mathsf{LO}}_i(\alpha_i) = \lim_{\epsilon \to 0} \frac{C^{\mathsf{LO}}_i}{\max(\epsilon, D_i - C^{\mathsf{LO}}_{max}) \cdot \alpha_i} = V^{\mathsf{LO}}_i \cdot \frac{1}{\alpha_i}, \quad (19)$$

$$R^{\mathsf{LO}}_i(\alpha_i) = C^{\mathsf{LO}}_i + C^{\mathsf{LO}}_{max} + \max(D_i - C^{\mathsf{LO}}_{max}) \cdot \alpha_i \cdot \frac{\sum_{\tau_j \in \tau \setminus \{\tau_i\}} V^{\mathsf{LO}}_j(\alpha_j)}{m}. \quad (20)$$

Differently from $V^{\mathsf{LO}}_i(\alpha_i)$ and $R^{\mathsf{LO}}_i(\alpha_i)$, we need to carefully construct $V^{\mathsf{TR}}_i(\alpha_i)$, because of the following observation.

Observation 1. Under NP-EDF (without the use of deadline scaling), V^{TR}_i increases as ℓ^{TR}_i increases as discussed in Lemma 3. However, under NP-EDFVD (with the use of the proposed deadline scaling), the same does not always hold.

As shown in Fig. 2a, without virtual deadlines, $V^{\mathsf{LO}}_i \leq V^{\mathsf{TR}}_i$ holds. However, if we decrease α_i from 1 to a particular value less than 1, $V^{\mathsf{LO}}_i(\alpha_i)$ increases while $V^{\mathsf{TR}}_i(\alpha_i)$ decreases as shown in Fig. 2b. If we further decrease α_i, it is possible to satisfy $V^{\mathsf{LO}}_i(\alpha_i) > V^{\mathsf{TR}}_i(\alpha_i)$ as shown in Fig. 2c. In the last case shown in Fig. 2c, if ℓ^{TR} increases, $V^{\mathsf{TR}}_i(\alpha_i)$ does not increase.

That is, V^{LO}_i is always no larger than V^{TR}_i with $\ell^{\mathsf{TR}}_i = 0$ under NP-EDF, while it is possible for $V^{\mathsf{LO}}_i(\alpha_i)$ for some α_i to be larger than $V^{\mathsf{TR}}_i(\alpha_i)$ with $\ell^{\mathsf{TR}}_i = 0$ under NP-EDFVD. In this case, $V^{\mathsf{TR}}_i(\alpha_i)$ is maximized when $\ell^{\mathsf{TR}}_i = 0$. Therefore, we need to consider both cases where ℓ^{TR}_i is the smallest (i.e., 0) and largest (i.e., R^{LO}_i) for calculating $V^{\mathsf{TR}}_i(\alpha_i)$, which are addressed in the first and second terms, respectively in the max function of the RHS of the following condition.

$$V_i^{\mathsf{TR}}(\alpha_i) = \lim_{\epsilon \to 0} \max\Bigg(\frac{C_i^{\mathsf{HI}}}{\max(\epsilon, D_i - C_{max})}, \frac{C_i^{\mathsf{HI}} - V_i^{\mathsf{LO}}(\alpha_i) \cdot \max(\epsilon, R_i^{\mathsf{LO}}(\alpha_i) - C_{max}^{\mathsf{LO}})}{\max\big(\epsilon, D_i - C_{max} - \max(\epsilon, R_i^{\mathsf{LO}}(\alpha_i) - C_{max}^{\mathsf{LO}})\big)} \Bigg). \tag{21}$$

Then, once we substitute $V_i^{\mathsf{LO}}(\alpha_i)$ and $V_i^{\mathsf{TR}}(\alpha_i)$ for V_i^{LO} and V_i^{TR} in Eqs. (1) and (9), for every $\tau_i | L_i = \mathsf{HI} \in \tau$, we can develop the following NP-EDFVD schedulability test for MC multiprocessor systems, corresponding to Theorem 1.

Theorem 2. *Suppose that τ satisfies Eqs. (22) and (23).*

$$\sum_{\tau_i \in \tau} V_i^{\mathsf{LO}}(\alpha_i) + (m-1) \cdot \max_{\tau_i \in \tau} V_i^{\mathsf{LO}}(\alpha_i) \le m. \tag{22}$$

$$\sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{TR}}(\alpha_i) + (m-1) \cdot \max_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{TR}}(\alpha_i) \le m. \tag{23}$$

Then, τ is schedulable by NP-EDFVD under MC multiprocessor systems.

Proof. The proof is the same as that for NP-EDF in Lemma 6 and Theorem 1. □

4.2 Optimal Virtual Deadline Assignment for the System-Level Deadline Reduction Parameter α

Now, we consider a virtual-deadline assignment problem for the system-level deadline reduction parameter $\alpha_i = \alpha$ for every $\tau_i \in \tau | L_i = \mathsf{HI}$. We wish to find the value of α that satisfies the conditions Eqs. (22) and (23). To this end, we need to derive properties on how the LHS of the conditions varies with α.

Observation 2. As α decreases, the LHS of Eq. (22) increases but that of Eq. (23) does not increase.

This is because, $V_i^{\mathsf{LO}}(\alpha)$ and $V_i^{\mathsf{TR}}(\alpha)$ are a decreasing and a non-decreasing function of α, respectively; the former is obvious, while the latter is not. Thus, let us explain it now.

In $V_i^{\mathsf{TR}}(\alpha)$ for $\tau_i \in \tau | L_i = \mathsf{HI}$, $V_i^{\mathsf{LO}}(\alpha)$ and $R_i^{\mathsf{LO}}(\alpha)$ are two terms that depend on α. Since we can conclude that $V_i^{\mathsf{TR}}(\alpha)$ is a non-decreasing function of α if we know that $V_i^{\mathsf{LO}}(\alpha)$ and $R_i^{\mathsf{LO}}(\alpha)$ are a non-increasing and non-decreasing function of α, respectively. And what remains to be explained is the latter, i.e., $R_i^{\mathsf{LO}}(\alpha)$.

The third term of $R_i^{\mathsf{LO}}(\alpha)$ for $\tau_i \in \tau | L_i = \mathsf{HI}$ in the RHS of Eq. (20) can be expressed as follows.

$$\begin{aligned} &\max\big(D_i - C_{max}^{\mathsf{LO}}\big) \cdot \alpha_i \cdot \frac{\sum_{\tau_j \in \tau \setminus \{\tau_i\}} V_j^{\mathsf{LO}}(\alpha_j)}{m} \\ &= \max\big(D_i - C_{max}^{\mathsf{LO}}\big) \cdot \frac{\sum_{\tau_j \in \tau \setminus \{\tau_i\} | L_j = \mathsf{HI}} V_j^{\mathsf{LO}}}{m} \\ &+ \max\big(D_i - C_{max}^{\mathsf{LO}}\big) \cdot \alpha \cdot \frac{\sum_{\tau_j \in \tau \setminus \{\tau_i\} | L_j = \mathsf{LO}} V_j^{\mathsf{LO}}}{m}. \end{aligned} \tag{24}$$

Therefore, $R_i^{\mathsf{LO}}(\alpha)$ is a non-decreasing function of α.

While we focus on a task set which violates at least one of Eqs. (22) and (23) (otherwise, Theorem 1 deems the task set schedulable by NP-EDF without virtual deadlines), Observation 2 indicates that we need to focus on a task which satisfies Eq. (22) but violates Eq. (23), and decrease α as much as possible until the LHS of Eq. (22) is equal to m, which reduces the LHS of Eq. (23) as much as possible.

We now propose an optimal virtual deadline assignment policy for the system-level parameter α in Algorithm 1. Due to the existence of the max function in the LHS of Eq. (1), we need to consider two cases depending on whether the task which exhibits the max value of V_i^{LO} is a HI-criticality task (the first case) or a LO-criticality task (the second case).

Algorithm 1. Optimal Virtual Deadline Assignment for the System-Level Parameter

1: $j \leftarrow \arg\max_{\tau_i \in \tau} V_i^{\mathsf{LO}}$;
2: **if** $L_j = \mathsf{HI}$ **then**
3: $\quad \alpha \leftarrow \frac{\sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} + (m-1) \cdot V_j^{\mathsf{LO}}}{m - \sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}}}$;
4: **else**
5: $\quad \alpha \leftarrow \frac{\sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}}}{m - \sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}} - (m-1) \cdot V_j^{\mathsf{LO}}}$;
6: $\quad j2 \leftarrow \arg\max_{\tau_i \in \tau} V_i^{\mathsf{LO}}(\alpha_j = \alpha)$;
7: $\quad$ **if** $j \neq j2$ **then**
8: $\quad\quad \alpha \leftarrow \frac{\sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} + (m-1) \cdot \max_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}}}{m - \sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}}}$.
9: $\quad$ **end if**
10: **end if**
11: **if** Eqs. (22) and (23) hold with $\alpha_i = \alpha$ for every $\tau_i \in \tau | L_i = \mathsf{HI}$ and $\alpha_i = 1$ for every $\tau_i \in \tau | L_i = \mathsf{LO}$ **then**
12: $\quad$ **return** Schedulable;
13: **end if**
14: **return** Unschedulable;

In the first case, since the task which exhibits the max value of V_i^{LO} (denoted by τ_j) still exhibits the max value of $V_i^{\mathsf{LO}}(\alpha)$ after changing α, we can simply solve the following equation (Lines 1–3):

$$\begin{aligned} &\sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}} + \frac{1}{\alpha} \cdot \sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} + \frac{1}{\alpha} \cdot (m-1) \cdot V_j^{\mathsf{LO}} = m \\ \Rightarrow \quad &\alpha = \frac{\sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} + (m-1) \cdot V_j^{\mathsf{LO}}}{m - \sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}}}. \end{aligned} \tag{25}$$

On the other hand, for the second case, it is possible to change the task which exhibits the max value of V_i^{LO} (from τ_j to τ_{j2}) if we reduce α. Therefore, we consider two subcases depending on whether the task which exhibits the max value of V_i^{LO} is (i) unchanged or (ii) changed by the change of α.

For (i), we can solve the similar equation as follows (Lines 4-5).

$$\begin{aligned} &\sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}} + \frac{1}{\alpha} \cdot \sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} + (m-1) \cdot V_j^{\mathsf{LO}} = m \\ \Rightarrow \quad &\alpha = \frac{\sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}}}{m - \sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}} - (m-1) \cdot V_j^{\mathsf{LO}}}. \end{aligned} \tag{26}$$

In order to know whether the current situation belongs to (i) or (ii), we first solve the above equation. If the task which exhibits the max value of V_i^{LO} still exhibits the max value of

$V_i^{\mathsf{LO}}(\alpha)$ after applying α calculated by the above equation, the resulting α is the final value; otherwise, we should consider (ii).

For (ii), the task which exhibits the max value of $V_i^{\mathsf{LO}}(\alpha)$ after applying a new α becomes the task with the largest V_i^{LO} among every task $\tau_i \in \tau | L_i = \mathsf{HI}$, yielding the following equation (Lines 6–9).

$$\begin{aligned} &\sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}} + \frac{1}{\alpha} \cdot \sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} \\ &+ \frac{1}{\alpha} \cdot (m-1) \cdot \max\nolimits_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} = m \\ \Rightarrow \quad &\alpha = \frac{\sum_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}} + (m-1) \cdot \max_{\tau_i \in \tau | L_i = \mathsf{HI}} V_i^{\mathsf{LO}}}{m - \sum_{\tau_i \in \tau | L_i = \mathsf{LO}} V_i^{\mathsf{LO}}}. \end{aligned} \tag{27}$$

Using the calculated α for the three cases, Algorithm 1 finally checks the schedulability (Lines 11–14). Then, the following theorem presents the optimality of the virtual-deadline assignment for α in Algorithm 1.

Theorem 3. *If Algorithm 1 returns unschedulable, there exists no α that makes Eqs. (22) and (23) hold with $\alpha_i = \alpha$ for every $\tau_i \in \tau | L_i = \mathsf{HI}$ and $\alpha_i = 1$ for every $\tau_i \in \tau | L_i = \mathsf{LO}$.*

Proof. Suppose that Algorithm 1 returns unschedulable (but it yields α denoted by α'), but there exists α'' that makes Eqs. (22) and (23) hold with $\alpha_i = \alpha''$ for every $\tau_i \in \tau | L_i = \mathsf{HI}$ and $\alpha_i = 1$ for every $\tau_i \in \tau | L_i = \mathsf{LO}$. If the LHS of Eq. (22) is exactly m, α'' should be the same as α'. If the LHS of Eq. (22) is strictly less than m, α'' should be larger than α'. Since $V_i^{\mathsf{TR}}(\alpha)$ is a non-decreasing function of α, it is impossible to meet Eq. (23) with α'' but not to do that with α'. □

We can easily calculate the time-complexity of Algorithm 1, which is $O(n)$.

4.3 Suboptimal Virtual Deadline Assignment for the Task-Level Deadline Reduction Parameter α_i

In this section, we solve a virtual deadline assignment problem for the task-level deadline reduction parameter α_i. To this end, we first discuss the valid range of α_i and analyze the effect of its change on schedulability.

While the definitional range of α_i is $[0, 1]$ as we discussed in Section 4.1, α_i should be larger than V_i^{LO} for schedulability. This is because, if $\alpha_i \leq V_i^{\mathsf{LO}}$, then $V_i^{\mathsf{LO}}(\alpha_i) \geq 1$, resulting in the LHS of Eq. (22) no smaller than m only with a single task τ_i. Therefore, the valid range of α_i is $[V_i^{\mathsf{LO}}, 1]$.

Observation 3. If a single α_i of τ_i decreases without changing α_j for every $\tau_j \in \tau \setminus \{\tau_i\}$, $V_i^{\mathsf{TR}}(\alpha_i)$ non-increases and $V_j^{\mathsf{TR}}(\alpha_j)$ for every $\tau_j \in \tau \setminus \{\tau_i\}$ non-decreases.

This is because, if α_i of τ_i is reduced, the following properties hold: $V_i^{\mathsf{LO}}(\alpha_i)$ and $R_j^{\mathsf{LO}}(\alpha_j)$ for every $\tau_j \in \tau \setminus \{\tau_i\}$ increase; $R_i^{\mathsf{LO}}(\alpha_i)$ decreases; and $V_j^{\mathsf{LO}}(\alpha_j)$ for every $\tau_j \in \tau \setminus \{\tau_i\}$ stays. Hence, with decrement of α_i, while $V_i^{\mathsf{TR}}(\alpha_i)$ non-increases, $V_j^{\mathsf{TR}}(\alpha_j)$ for every $\tau_j \in \tau \setminus \{\tau_i\}$ non-decreases. This implies it is not true that the LHS of Eq. (23) is a non-decreasing function of α_i for every $\tau_i \in \tau | L_i = \mathsf{HI}$; therefore, we cannot simply find α_i for every $\tau_i \in \tau | L_i = \mathsf{HI}$ that satisfies Eqs. (22) and (23), without exhaustive search.

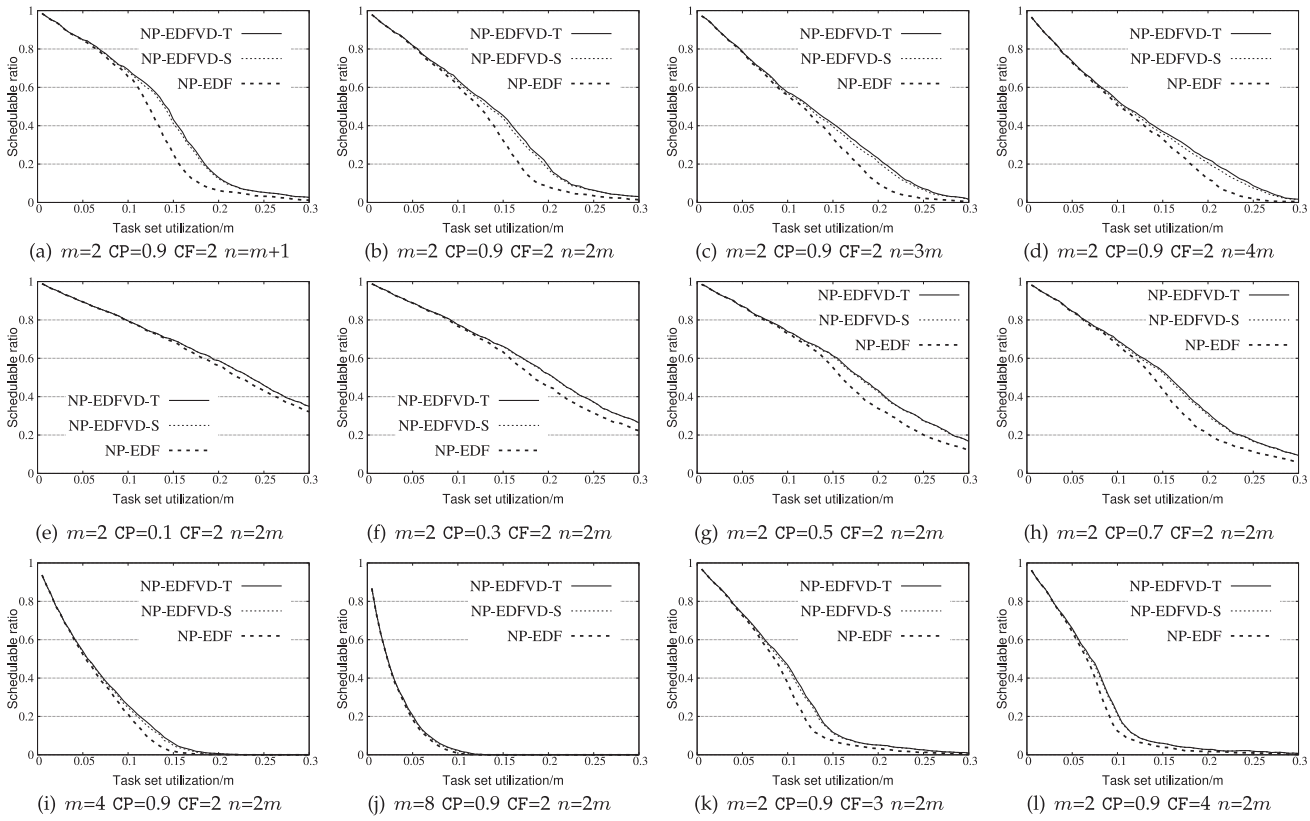
Therefore, we propose a suboptimal algorithm that finds a proper individual α_i for every $\tau_i \in \tau | L_i = \mathsf{HI}$ with a small number of trials. The algorithm targets task sets that are schedulable by neither NP-EDF (by Theorem 1) nor NP-EDFVD with the system-level deadline-reduction parameter α (by Algorithm 1). Staring from $\alpha_i = 1$ for every $\tau_i \in \tau | L_i = \mathsf{HI}$, we try to reduce each α_i by ϵ and compute the difference between the LHS of Eq. (22) and that of Eq. (23). If τ_j yields the largest decrement of the latter normalized by the increment of the former, we decide to reduce α_j by ϵ. We repeat this process until there is no task to reduce its α_i or the task set is schedulable with the current setting of α_i.

We present this suboptimal algorithm in Algorithm 2 for a given ϵ. We first set α_i of every task $\tau_i \in \tau$ to 1 (Lines 1–3). Then, for every $\tau_i \in \tau | L_i = \mathsf{HI}$ and $\alpha_i - \epsilon > V_i^{\mathsf{LO}}$ (i.e., every HI-criticality task τ_i that can reduce its α_i), we calculate the decrement of the LHS of Eq. (23) in case that α_i is reduced by ϵ, normalized by the increment of the LHS of Eq. (22) in case that α_i is reduced by ϵ (denoted by $\triangle_i$ in Line 8); we then select the largest $\triangle_i$ (Lines 9–12). If the value of $\triangle_i$ is not changed for every $\tau_i \in \tau | L_i = \mathsf{HI}$, we deem τ unschedulable (Lines 14–16). Otherwise, we decrease α_j by ϵ, for $\tau_j \in \tau | L_j = \mathsf{HI}$ with the greatest value of $\triangle_i$ (Line 17). Thereafter, we repeat to find an α_i to be reduced, and we stop the repetition if the LHS of Eq. (22) is larger than m (unschedulable, Line 4) or Eqs. (22) and (23) hold (schedulable, Lines 18–20).

Algorithm 2. Virtual Deadline Assignment for the Task-Level Parameter (ϵ)

```
1: for τ_i ∈ τ do
2:    α_i = 1;
3: end for
4: while the LHS of Eq. (22) < m do
5:    index = 1;
6:    △_max = 0;
7:    for τ_i ∈ τ | L_i = HI and α_i − ε > V_i^LO do
8:       Compute △_i = − (△the LHS of Eq. (23)) / (△the LHS of Eq. (22)) for ε.
9:       if △_i > △_max then
10:         index ← i;
11:         △_max = △_i
12:      end if
13:   end for
14:   if △_max = 0 then
15:      return Unschedulable;
16:   end if
17:   α_index ← α_index − ε;
18:   if Eqs. (22) and (23) hold then
19:      return Schedulable;
20:   end if
21: end while
22: return Unschedulable;
```

There are at most $(1/\epsilon)$ points to be checked for a single α_i, in total up to $O(n/\epsilon)$ points, and each point needs to calculated Eqs. (22) and (23). Therefore, the time-complexity of Algorithm 2 is $O(n^2/\epsilon)$.



(a) m=2 CP=0.9 CF=2 n=m+1 (b) m=2 CP=0.9 CF=2 n=2m (c) m=2 CP=0.9 CF=2 n=3m (d) m=2 CP=0.9 CF=2 n=4m

(e) m=2 CP=0.1 CF=2 n=2m (f) m=2 CP=0.3 CF=2 n=2m (g) m=2 CP=0.5 CF=2 n=2m (h) m=2 CP=0.7 CF=2 n=2m

(i) m=4 CP=0.9 CF=2 n=2m (j) m=8 CP=0.9 CF=2 n=2m (k) m=2 CP=0.9 CF=3 n=2m (l) m=2 CP=0.9 CF=4 n=2m

Fig. 3. Schedulable ratio of the three individual schedulability tests over varying n, CP, m and CF.

5 Evaluation

In this section, we evaluate the schedulability performance of the proposed schedulability tests for NP-EDF and NP-EDFVD on MC multiprocessor platforms. We first illustrate our simulation environments and then discuss various factors influencing schedulability of the proposed schedulability tests with empirical simulation results.

5.1 Task Set Generation

For task set generation, we employ UUnifast-discard [25], which is a popular task set generation method for multiprocessors, originated from UUnifast for uniprocessor platforms [26]. Under UUnifast-discard, we have three input parameters: (i) the number of processors m, (ii) the number of tasks n, and (iii) task set utilization $U = \sum_{\tau_i \in \tau} C_i^{\mathsf{LO}}/T_i$. In addition, we have two additional input parameters for MC scheduling [13]: (iv) the ratio between each task τ_i's WCETs for the high- and low-criticality level, i.e., $\frac{C_i^{\mathsf{HI}}}{C_i^{\mathsf{LO}}}$ (denoted by CF), and (v) the probability of each task τ_i having $L_i = \mathsf{HI}$ (denoted by CP). We apply a similar parameter setting to [13], which is a multiprocessor MC scheduling study, as follows: (i) m = 2, 4, 8, (ii) n = m+1, 2m, 3m, 4m, (iii) U = $0.005m, 0.010m, 0.015m, \ldots, 1.000m$, (iv) CF = 2, 3, 4, and (v) CP = 0.1, 0.3, 0.5, 0.7, 0.9.

For a 5-tuple $(m, n, U, \mathtt{CF}, \mathtt{CP})$, each task parameter is determined as follows. Based on a 3-tuple (m, n, U), UUnifast-discard [25] generates every task's utilization (i.e., $u_i = C_i^{\mathsf{LO}}/T_i$). Each task's period T_i is uniformly selected in $[1ms, 1000ms]$; C_i^{LO} is selected based on the chosen task utilization (i.e., $C_i^{\mathsf{LO}} = u_i \cdot T_i$); and L_i is selected as HI with probability CP (and as LO with probability $(1.0 - \mathtt{CP})$). If $L_i = \mathsf{HI}$, C_i^{HI} is set to $(\mathtt{CF} \cdot C_i^{\mathsf{LO}})$;[2] otherwise, C_i^{HI} is set to C_i^{LO}.

For given a list of five parameters, we generate 1000 task sets, yielding 1000 * 3 * 4 * 200 * 3 * 5 = 36,000,000 task sets in total.

5.2 Evaluation Results

With the generated task sets, we compare the following three schedulability tests:

- NP-EDF: the schedulability test in Theorem 1 for NP-EDF;
- NP-EDFVD-S: the schedulability test in Theorem 2 for NP-EDFVD with the system-level parameter (Algorithm 1); and
- NP-EDFVD-T: the schedulability test in Theorem 2 for NP-EDFVD with the task-level parameter (Algorithm 2 with $\epsilon = 0.01$), after applying NP-EDFVD-S.

We show results of implicit-deadline task sets ($D_i = T_i$ for every $\tau_i \in \tau$); the results of constrained-deadline task sets ($D_i \leq T_i$ for every $\tau_i \in \tau$) exhibit a similar trend but with low schedulability.

For performance metric, we use *schedulable ratio* defined as the ratio of the number of tasks that are deemed schedulable by each individual schedulability test, to the number of generated task sets under the given input parameters. For different combinations of input parameters (i.e., n, CP, m and CF), Fig. 3 shows the schedulable ratios for the three schedulability tests, and Fig. 4 shows the relative ratio

2. If $C_i^{\mathsf{HI}} > T_i$ (meaning an infeasible task set), we discard this task set and re-generate another task set.

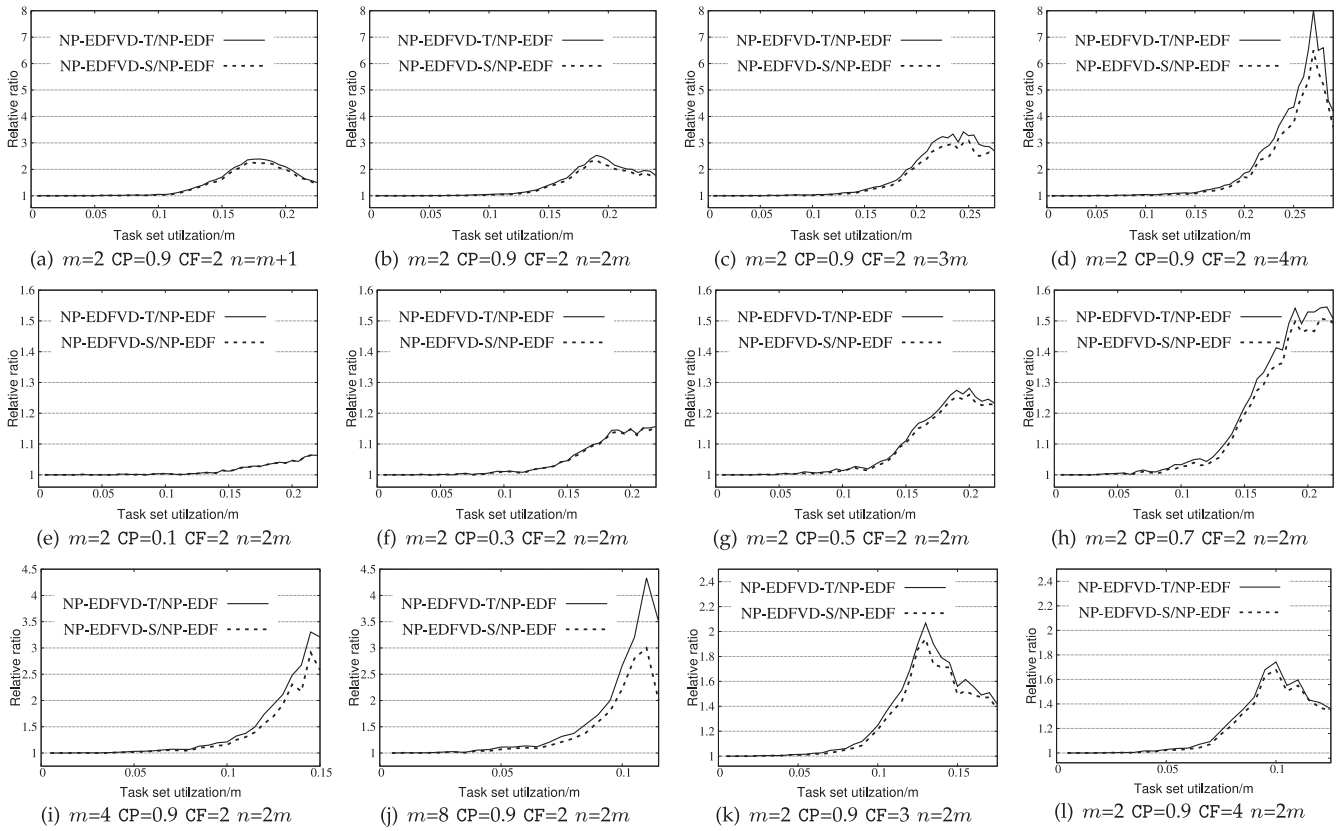


Fig. 4. Relative ratio between the schedulable ratios of NP-EDFVD-S and NP-EDF (denoted by NP-EDFVD-S/NP-EDF), and that between the schedulable ratios of NP-EDFVD-T and NP-EDF (denoted by NP-EDFVD-T/NP-EDF) over varying n, CP, m and CF.

between the schedulable ratios of NP-EDFVD-S and NP-EDF, and that between the schedulable ratios of NP-EDFVD-T and NP-EDF, according to varying value of U/m.[3] Now we discuss how each input parameter influences the overall schedulable ratio of the three schedulability tests (with Fig. 3) and impact of the system-level and task-level virtual-deadline assignment schemes (with Fig. 4).

The number of tasks (in subfigures (a), (b), (c) and (d) of Figs. 3 and 4). As seen in Figs. 3a, 3b, 3c and 3d, schedulable ratios of the three schedulability tests for a given task set utilization (U) become improved as the number of tasks in a task set (n) increases. This is because, a large number of tasks in a task set for a given U reduces average utilization of each task, which tends to yield low V_{max}^{LO} and V_{max}^{TR} in Eqs. (1) and (9), respectively. Also, Figs. 4a, 4b, 4c and 4d show that schedulability improvement achieved by NP-EDFVD-S (as well as NP-EDFVD-T) compared to NP-EDF becomes magnified as the number of tasks in a task set is increased. For example, NP-EDFVD-T exhibits up to 700.0 percent schedulability improvement over NP-EDF (with $n = 4m$ and $U/m = 0.27$ in Fig. 4d). If we compare schedulability improvement by NP-EDFVD-T, with that by NP-EDFVD-S, the former outperforms the latter up to 150.0 percent (with the same parameters). These observations indicate that a lower average utilization of tasks (due to a greater value of n) allows a better chance for task sets deemed unschedulable by NP-EDF to become schedulable by NP-EDFVD-S (or NP-EDFVD-T); this is because, a lower average utilization of tasks tends to yield a smaller V_{max}^{TR}, which allows unschedulable task sets by NP-EDF to be schedulable by slightly adjusting the virtual deadline of HI-tasks.

The probability of CP *(in subfigures (e), (f), (g), (h) and (b) of Figs. 3 and 4).* It is observed from Fig. 3 that a larger CP (i.e., a higher number of HI-tasks in a task set) degrades schedulability performance of the three schedulability tests. This is due to our proposed task set generation method, in which U is determined by the summation of C_i^{LO}/T_i of all tasks in a task set. A higher value of CP increases the number of HI-tasks in a task set for a given U, there by decreasing schedulability. On the other hand, such an increased number of HI-tasks allows a larger room for schedulability performance improvement achieved by virtual deadline assignment as shown in Fig. 4. NP-EDFVD-T improves schedulable ratio of NP-EDF up to 153.1 percent for CP = 0.9 (Fig. 4b for $U/m = 0.19$) while NP-EDFVD-T does it up to 7.1 percent for CP = 0.1 (Fig. 4e for $U/m = 0.225$).

The number of processors (in subfigures (b), (i) and (j) of Figs. 3 and 4). As m increases, schedulable ratios of the three schedulability tests decrease for a given n/m and U/m as shown in Figs. 3b, 3i and 3j. To interpret this observation, we divide Eq. (1) by m, thereby resulting in $\frac{V_{sum}^{\mathsf{LO}}}{m} + \frac{m-1}{m} \cdot V_{max}^{\mathsf{LO}} \leq 1.0$. While the first term tends to have a similar expected value for a fixed value of U/m, V_{max}^{LO} simply depends on the largest V_i^{LO}; a larger m tends to have a larger V_{max}^{LO}. In addition, $\frac{m-1}{m}$ also becomes larger as m increases: 0.5, 0.75, and

3. Individual subfigures in Fig. 3 plot up to $U/m = 0.3$ for x-axis since most lines converge to 0 after 0.3. When it comes to Fig. 4, each x-axis of individual subfigures plots up to *at most* 0.3 (but different values) in order to avoid a small sample size where the number of task sets deemed schedulable by NP-EDF is very small.

0.875 for $m = 2$, 4, and 8. On the other hand, our virtual deadline assignment schemes significantly make up such degradation stemming from increasing m. For example, the schedulability improvement by NP-EDFVD-T is 153.1 percent for $m = 2$ (in Fig. 4b), while it is increased to 230.4 percent and 333.3 percent for $m = 4$ and 8 (in Figs. 4i and 4j). If we focus on the gap between schedulability improvement by NP-EDFVD-T and NP-EDFVD-S, the gap increases as m increases. That is, the gap is 153.1- 135.4 = 17.7 percent for $m = 2$, but it is increased to 230.4 - 191.3 = 39.1 percent, and 333.3 − 200.0 = 133.3 percent for $m = 4$ and 8, respectively. This indicates that the task-level virtual deadline assignment scheme (rather than the system-level one) has higher opportunities to improve performance of NP-EDF by adjusting individual task's virtual deadline.

The ratio of CF *(in subfigures (b), (k) and (l) of Figs. 3 and 4).* We observe that overall schedulability performance of the three schedulability tests degrades as CF increases (as shown in Figs. 3b, 3k and 3l); the reason is the same as that for CP. Note that schedulability improvement by NP-EDFVD-T and NP-EDFVD-S does not increase as CF increases (as shown in Figs. 4b, 4k and 4l). This is because, the room for improvement is limited with a large value of C_i^{HI} where it is difficult for unschedulable task sets to become schedulable with any virtual deadline assignment.

6 Related Work

Since Vestal's seminal work [1], there have been a body of research on MC scheduling for a uniprocessor platform. Baruah et al. proposed RTA schedulability tests for static and adaptive mixed-criticality priority assignment schemes, and demonstrated a dominance relationship between the two [2], [3]. Also, they proposed a new scheduling algorithm referred to as preemptive EDFVD (Earliest Deadline First with Virtual Deadlines) [4], [5]. In preemptive EDFVD, smaller (virtual) relative deadlines are assigned in LO-mode for HI-criticality tasks by using a single system-level scaling factor so as to guarantee schedulability across mode changes. Ekberg and Yi [27] improved upon preemptive EDFVD by enabling task-level deadline scaling factors. Li et al. introduced an OCBP (Own Criticality Based Priority) scheduling algorithm and its schedulability analysis for general task sets [6], [7]. Based on OCBP scheduling, Guan et al. proposed a more efficient algorithm called PLRS [8]. As to new models, Su et al. proposed an E-MC (Elastic Mixed-Criticality) model [9]. Also, Baruah introduced a general model of mixed-criticality recurrent real-time tasks considering different estimates on WCET, relative deadline, and period depending on criticality levels [10].

The first work discussing MC multiprocessor scheduling was by Anderson et al. [11] and extended in 2010 [12]. They considered five levels of criticality and suggested an implementation scheme called MC^2, employing different scheduling algorithms according to criticality level. Pathan proposed a response time analysis (RTA) for global preemptive FP (Fixed-Priority) scheduling, which is applicable to Ausley's optimal priority assignment [13]. Li et al. extended preemptive EDFVD to multiprocessors with respect to both global and partitioning scheduling for MC systems, and compared their effectiveness [14], [15]. Su et al. studied the E-MC model in multicore systems considering the systems with or without task migrations [16]. Lee et al. incorporated the concept of a fluid scheduling model into the MC domain and introduced a new scheduling algorithm, called MC-Fluid, which executes each task according to its criticality-dependent execution rate [17].

While a lot of studies on preemptive scheduling for MC systems have been made, research on non-preemptive scheuduling have not matured, and have been limited to uniprocessor systems and distributed systems. Baruah's RTA approach for adaptive mixed-criticality priority assignment scheme was extended by Zhao et al. to incorporate preemption thresholds into the model [19], [20]. Burns and Davis considered deferred preemption that exploits the notion of final non-preemptive region (FNPR) [22]. Baruah and Guo studied non-preemptive scheduling on unreliable processors and proved that the polynomial-time optimal scheduling strategies cannot exist for non-preemptive MC scheduling [21]. Hanzálek et al. addressed the non-preemptive mixed-criticality match-up scheduling problem arising from the areas of the communication protocols, and they also proved the NP-hardness of the problem [23]. As far as we know, there has been no work considering non-preemptive scheduling for MC multiprocessor systems, and this paper is the first work to develop schedulability tests for NP-EDF as well as NP-EDFVD.

7 Conclusion

In this paper, we developed schedulability tests of NP-EDF and NP-EDFVD for MC multiprocessor systems, which is the first attempt for non-preemptive scheduling on MC multiprocessor systems. To this end, we first investigated the schedulability analysis techniques of an existing NP-EDF schedulability test for SC multiprocessor systems, and generalized the techniques to address the system transition. We then extended the proposed NP-EDF schedulability test to NP-EDFVD. After posing the virtual deadline assignment problem for NP-EDFVD, we developed an optimal assignment policy for the system-level deadline-reduction parameter and a suboptimal policy for the task-level parameter. Our simulation results demonstrated that the NP-EDFVD schedulability analysis with the proposed assignment polices exhibits up to 134.3 percent schedulability improvement, compared to the NP-EDF schedulability test. In future, we plan to target other existing schedulability tests of non-preemptive scheduling for SC multiprocessor systems, and develop tighter schedulability tests for MC multiprocessor systems.

Acknowledgments

This research was supported by the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (2017R1A2B2002458, 2017H1D8A2031628, 2017K2A9A1A01092689) and the Ministry of Education (2016R1A6A3A11930688). This research was also supported by the IITP (Institute for Information & communications Technology Promotion) funded by the MSIT (Ministry of Science and ICT) (2015-0-00914, IITP-2017-2015-0-00742, 2014-0-00065, Resilient Cyber-Physical Systems Research; 2017M3C4A7065927). Jinkyu Lee is the corresponding author.

REFERENCES

[1] S. Vestal, "Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance," in *Proc. IEEE Real-Time Syst. Symp.*, 2007, pp. 239–243.

[2] S. Baruah and A. Burns, "Implementing mixed criticality systems in ADA," in *Proc. 16th Ada-Eur. Int. Conf. Reliable Softw. Technol.*, 2011, pp. 174–188.

[3] S. Baruah, A. Burns, and R. I. Davis, "Response-time analysis for mixed criticality systems," in *Proc. IEEE Real-Time Syst. Symp.*, 2011, pp. 34–43.

[4] S. Baruah, V. Bonifaci, G. D'Angelo, A. MarchettiSpaccamela, S. van der Ster, and L. Stougie, "Mixed-criticality scheduling of sporadic task systems," in *Proc. 19th Annu. Eur. Symp. Algorithms*, 2011, pp. 555–566.

[5] S. Baruah, V. Bonifaci, G. D'Angelo, H. L. A. Marchetti-Spaccamela, S. van der Ster, and L. Stougie, "The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems," in *Proc. Euromicro Conf. Real-Time Syst.*, 2012, pp. 145–154.

[6] S. Baruah, H. Li, and L. Stougie, "Toward the design of certifiable mixed-criticality systems," in *Proc. IEEE Real-Time Technol. Appl. Symp.*, 2010, pp. 13–22.

[7] H. Li and S. Baruah, "An algorithm for scheduling certifiable mixed-criticality sporadic task systems," in *Proc. IEEE Real-Time Syst. Symp.*, 2010, pp. 183–192.

[8] N. Guan, P. Ekberg, M. Stigge, and W. Yi, "Effective and efficient scheduling of certifiable mixed-criticality sporadic task systems," in *Proc. IEEE Real-Time Syst. Symp.*, 2011, pp. 13–23.

[9] H. Su and D. Zhu, "An elastic mixed-criticality task model and its scheduling algorithm," in *Proc. Des. Autom. Test Eur. Conf. Exhibition*, 2013, pp. 147–152.

[10] S. Baruah, "Schedulability analysis for a general model of mixed-criticality recurrent real-time tasks," in *Proc. IEEE Real-Time Syst. Symp.*, 2016, pp. 25–34

[11] J. H. Anderson, S. Baruah, and B. B. Brandenburg, "Multicore operating-system support for mixed criticality," in *Proc. Workshop Mixed-Criticality: Roadmap Evolving UAV Certification*, 2009, pp. 1–11.

[12] M. Mollison, J. Erickson, J. Anderson, S. Baruah, and J. Scoredos, "Mixed criticality real-time scheduling for multicore systems," in *Proc. 7th IEEE Int. Conf. Embedded Softw. Syst.*, 2010, pp. 1864–1871.

[13] R. Pathan, "Schedulability analysis of mixed-criticality systems on multiprocessors," in *Proc. Euromicro Conf. Real-Time Syst.*, 2012, pp. 309–320.

[14] H. Li and S. Baruah, "Global mixed-criticality scheduling on multiprocessors," in *Proc. Euromicro Conf. Real-Time Syst.*, 2012, pp. 166–175.

[15] S. Baruah, B. Chattopadhyay, H. Li, and I. Shin, "Mixed-criticality scheduling on multiprocessors," *Real-Time Syst.*, vol. 50, no. 10, pp. 142–177, 2014.

[16] H. Su, D. Zhu, and D. Mossé, "Scheduling algorithms for elastic mixed-criticality tasks in multicore systems," in *Proc. IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl.*, 2013, pp. 352–357.

[17] J. Lee, K. Phan, X. Gu, J. Lee, A. Easwaran, I. Shin, and I. Lee, "Mc-fluid: Fluid model-based mixed-criticality scheduling on multiprocessorss," in *Proc. IEEE Real-Time Syst. Symp.*, 2014, pp. 41–52.

[18] G. Buttazzo, M. Bertogna, and G. Yao, "Limited preemptive scheduling for real-time systems: A survey," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 3–15, Feb. 2013.

[19] Q. Zhao, Z. Gu, and H. Zeng, "Integration of resource synchronization and preemption-thresholds into EDF-based mixed-criticality scheduling algorithm," in *Proc. IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl.*, 2013, pp. 227–236.

[20] Q. Zhao, Z. Gu, and H. Zeng, "PT-AMC: Integrating preemption thresholds into mixed-criticality scheduling," in *Proc. Des. Autom. Test Eur. Conf. Exhibition*, 2013, pp. 141–146.

[21] S. K. Baruah and Z. Guo, "Mixed criticality scheduling upon unreliable processors," University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 2013, pp. 1–10.

[22] A. Burns and R. I. Davis, "Adaptive mixed criticality scheduling with deferred preemption," in *Proc. IEEE Real-Time Syst. Symp.*, 2014, pp. 21–30.

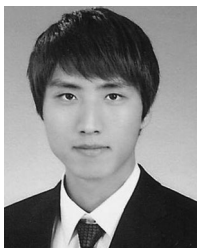
[23] Z. Hanzálek, T. Tunys, and P. Šucha, "An analysis of the non-preemptive mixed-criticality match-up scheduling problem," *J. Scheduling*, vol. 19, pp. 601–607, 2016.

[24] S. Baruah, "The non-preemptive scheduling of periodic tasks upon multiprocessors," *Real-Time Syst.*, vol. 32, no. 1, pp. 9–20, 2006.

[25] R. Davis and A. Burns, "Priority assignment for global fixed priority pre-emptive scheduling in multiprocessor real-time systems," in *Proc. IEEE Real-Time Syst. Symp.*, 2009, pp. 398–409.

[26] E. Bini and G. Buttazzo, "Measuring the performance of schedulability tests," *Real-Time Syst.*, vol. 30, no. 1–2, pp. 129–154, May 2005.

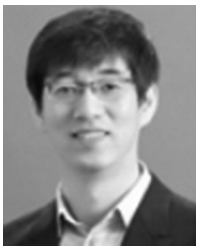
[27] P. Ekberg and W. Yi, "Bounding and shaping the demand of mixed-criticality sporadic tasks," in *Proc. Euromicro Conf. Real-Time Syst.*, 2012, pp. 135–144.



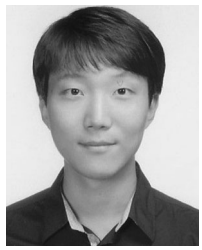
Hyeongboo Baek received the BS degree in computer science and engineering from Konkuk University, South Korea, the MS and PhD degrees from the Department of Computer Science at Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2010 and 2016. He is currently a postdoctoral researcher with Sungkyunkwan University, South Korea. His research interests include real-time embedded systems, cyber-physical systems and security. He won the best paper award from the 33rd IEEE Real-Time Systems Symposium (RTSS) in 2012.

Namyong Jung received the BS degree from Sungkyunkwan University, in 2016. He is working toward the MS degree with Sungkyunkwan University. His research interests include timing guarantees of real-time embedded systems.

Hoon Sung Chwa received the BS, MS, and PhD degrees all in computer science, from the Korea Advanced Institute of Science and Technology, in 2009, 2011, and 2016, respectively. He is currently a research fellow with the University of Michigan, Ann Arbor, Michigan. His research interests include system design and analysis with timing guarantees and resource management in real-time embedded systems and cyber-physical systems. He won two best paper awards from the 33rd IEEE Real-Time Systems Symposium (RTSS) in 2012 and from the IEEE International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA) in 2014. He is a member of 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, KAIST, South Korea, where he joined in 2008. He has been a postdoctoral research fellow with Malardalen University, Sweden, and a visiting scholar with 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 the *Journal of Computing Science and Engineering*. He has been co-chair 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.



Jinkyu Lee received the BS, MS, and PhD degrees in computer science from the Korea Advanced Institute of Science and Technology, South Korea, in 2004, 2006, and 2011, respectively. He is an assistant professor in the Department of Computer Science and Engineering, Sungkyunkwan University, South Korea, where he joined in 2014. 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. He is a member of the IEEE.