Mixed-Criticality Scheduling in Compositional Real-Time Systems with Multiple Budget Estimates

Kecheng Yang
Department of Computer Science
Texas State University
yangk@txstate.edu

Zheng Dong
Department of Computer Science
Wayne State University
dong@wayne.edu

***Abstract*—In order to mitigate the pessimism in parameter estimation in real-time systems, mixed-criticality (MC) scheduling has been proposed and studied. In light of the first MC scheduling work focusing on multiple estimates on the worst-case execution times (WCETs), a few following works also have extended this approach to other dimensions, such as period, relative deadlines, and processor speeds. Nonetheless, in most existing work on MC scheduling, a flat-structured scheduling approach is assumed, whereas compositional real-time systems with hierarchical scheduling are of great interest, especially for large-scale real-time systems. In this work, we aim to extend the fundamental ideas and frameworks of MC scheduling to another dimension, namely the budget estimation. To illustrate this approach, we form up a specific scheduling problem in the context of a single virtual processor characterized by the periodic resource model and propose a virtual-deadline-based algorithm to solve it. Furthermore, we have developed a polynomial-time schedulability test and have proved a speed-up bound for the proposed algorithm. We have also derived a range for setting the resource period to ensure schedulability when the bandwidth and task set are given. Moreover, we have conducted schedulability studies and presented our simulation experiment results to evaluate the proposed model and algorithm.**

***Index Terms*—real-time systems, mixed-criticality scheduling, hierarchical scheduling, periodic resource model**

I. Introduction

Real-time systems design is often aiming for providing *guarantees* for meeting deadlines *in all possible scenarios*. As a result, significant *pessimism* usually exists in real-time scheduling analysis and design. The system parameters, *e.g.*, the needed execution time for a piece of code, are provisioned as upper bounds, which are often not tight, on the worst case. Consequently, while the system design and certification need to follow these pessimistic provisioned system parameters, computing resources might be significantly underutilized in practice due to the potentially huge gap between the general scenario during runtime and the worst-case provision used for analysis, design, and certification. One approach to mitigate such pessimism is *mixed-criticality (MC) scheduling* [11, 63]. Under MC scheduling, tasks are grouped to two distinct *criticality levels*—HI (stands for high criticality) and LO (stands for low criticality), and each system parameter might have two estimates—a more pessimistic one that provides an absolutely safe bound on even the worst-case scenario, and a less pessimistic one that covers a dominant majority of scenarios in practice (*e.g.*, take the observed worst case in measurement-based experiments). An MC scheduler is designed based on the criticality levels and system parameter estimates, and needs to guarantee that the deadlines of all tasks (*i.e.*, both HI- and LO-tasks) are met in "normal" scenarios in common practice while the deadlines of all HI-tasks are still met even if any pathological extreme cases do happen.

Although MC scheduling has received much attention in the real-time systems research community, most existing work has been directed to a flat-structured scheduling approach, *i.e.*, a single central scheduler is used to schedule all tasks in the entire system. However, as computing systems, even embedded ones, become increasingly complicated and highly integrated, a single system may need to be designed, analyzed, implemented, and certified as several isolated *components*, with each component having the "illusion" of executing on a dedicated *virtual* platform [23]. In a compositional real-time system, the scheduling follows a hierarchical approach. An upper-level scheduler distributes the computing capacity of the physical computing platform to each component and determines the characteristics of the virtual platform in each component. Then, each component has a lower-level scheduler to schedule the tasks in that component upon that virtual platform.

In order to analyze and certify each component independently, interfaces are needed to characterize the *supply* provided by a virtual processor (VP), *i.e.*, available processing time units from the physical processor to support task execution. For example, the *periodic resource model* [58] is such a fundamental interface, which characterizes a VP by a pair of parameters (Π, Θ), with the interpretation that *at least* Θ time units of processor time is guaranteed to the supported task set every Π time units. Also, the quotient Θ/Π is called the *bandwidth* of this VP.

As Θ indicates the *minimum* budget for *any* resource period, it may be necessarily estimated with significant pessimism, just like what happens to the *worst-case execution time (WCET)* estimates. As a result, the resource may be greatly wasted in the actual runtime.

A motivating scenario. As an example scenario where multiple estimates of a single VP are indeed needed, let us consider a compositional system where MC concepts can happen in two hierarchies. Also, we use HIGH and LOW to denote the

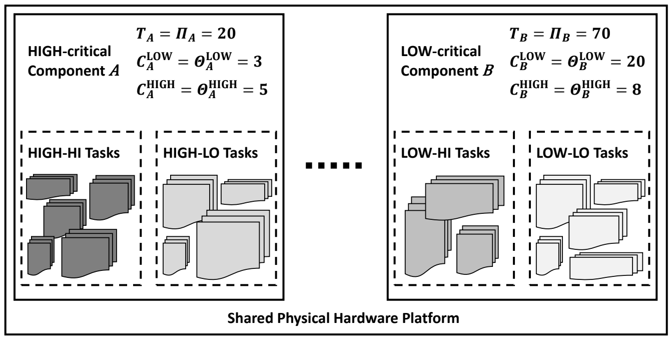

$T_A = \Pi_A = 20$
HIGH-critical Component A
$C_A^{\text{LOW}} = \Theta_A^{\text{LOW}} = 3$
$C_A^{\text{HIGH}} = \Theta_A^{\text{HIGH}} = 5$
HIGH-HI Tasks
HIGH-LO Tasks
$T_B = \Pi_B = 70$
LOW-critical Component B
$C_B^{\text{LOW}} = \Theta_B^{\text{LOW}} = 20$
$C_B^{\text{HIGH}} = \Theta_B^{\text{HIGH}} = 8$
LOW-HI Tasks
LOW-LO Tasks
Shared Physical Hardware Platform

Fig. 1: Illustration of a motivating scenario..

criticalities at the server level and use HI and LO to denote the criticalities within a component, *i.e.*, on a VP.

As shown in Fig. 1, on the upper-level, two MC components A and B (potentially together with other components omitted for simplicity) are supported by two server tasks τ_A and τ_B, which are scheduled as two conventional MC tasks (HIGH-task τ_A and LOW-task τ_B) on the shared physical platform. Then, recent work on MC graceful degradation [8, 51] can apply such that when the HIGH component A requires additional budget (*i.e.*, τ_A overruns smaller budget and requires larger budget (*e.g.*, from 3 to 5 units), potentially for accommodating extreme behaviors of the HIGH-HI tasks within component A), server task τ_B and therefore the LOW component B gets a degraded budget (*e.g.*, from 20 to 8 units) henceforth.

Focusing on the lower-level hierarchy and inside the LOW component B, the local tasks may also be of different criticalities in a nested manner, denoted as LOW-HI and LOW-LO, and the LOW-LO tasks should be sacrificed for the LOW-HI tasks upon the budget degradation for component B. When we analyze and certify the component B independently in an isolated manner, which is exactly what compositional systems aim to achieve), it results in the problem we address in this paper, *i.e.*, the MC scheduling problem with multiple VP budget estimates.

Contributions. In this paper, we extend the work of mixed-criticality scheduling to the resource supply estimates in the context of compositional real-time systems. By provisioning resource budget with multiple estimates, we show that fundamental ideas and framework of mixed-criticality scheduling can be applied in another dimension of scheduling problems that have not been considered before. To illustrate this approach, we form up a specific scheduling problem in the context of a single virtual processor characterizing by the periodic resource model and propose a virtual-deadline-based algorithm to solve it. Our further technical contributions in this paper include:

- a polynomial-time schedulability test for the proposed algorithm is developed;
- a speed-up bound for the proposed algorithm is proved;
- a range for setting the resource period is derived to ensure schedulability when bandwidth and task set are given;
- schedulability studies are conducted and simulation experiment results are presented.

Organization. In the rest of this paper, we describe our system model and review a few direct background results (Sec. II), describe our proposed scheduling algorithm (Sec. III), present a sufficient schedulability test (Sec. IV), prove a speed-up bound (Sec. V), derive a range for setting resource period (Sec. VI), present our experiment evaluation (Sec. VII), discuss related work (Sec. VIII), and conclude (Sec. IX).

II. System Model and Background

In this paper, we consider the preemptive scheduling of a set of tasks of two criticalities on a single VP. Also, we assume time is discrete in this paper, *i.e.*, all parameters representing an amount of time units are assumed to be integers and any scheduling event and decision must happen at an integer time instant.

Resource model. In light of the *periodic resource model* [58], we assume this single virtual processor provides certain available processing time units to the task set every Π time units, and Π is called the *resource period* of this virtual processor. The amount of available time units within every resource period is called the *budget* of this virtual processor. In the original periodic resource model, a single budget parameter Θ is assumed, indicating the minimum amount of budget in any resource period. In contrast, we apply two estimates to the budget: a *critical* budget Θ^C indicating an absolute lower bound on the budget in any resource period (*e.g.*, derived by the most pessimistic analysis), and a *nominal* budget Θ^N indicating a less pessimistic lower bound on the budget in any resource period (*e.g.*, based on observations in empirical experiments). That is, $\Theta^C \le \Theta^N$. Also, the *critical bandwidth* and *nominal bandwidth* are defined by

$$w^C = \frac{\Theta^C}{\Pi} \text{ and } w^N = \frac{\Theta^N}{\Pi}, \text{ respectively.} \quad (1)$$

Task model. On this virtual processor, a set of sporadic tasks $\mathcal{T}$ is supposed to be scheduled. Each task $\tau_i \in \mathcal{T}$ releases a sequences of *jobs* with a minimum separation of T_i time units, where T_i is called the *period* of τ_i. Any job of τ_i may be executed for up to C_i time units to complete, *i.e.*, C_i is the WCET of τ_i. In this paper, we also assume that the deadlines are implicit, *i.e.*, every task τ_i has a *relative deadline* of T_i time units indicating that every job of τ_i has an *absolute deadline* at T_i time units after its release. Furthermore, the *utilization* of task τ_i is defined by $u_i = C_i/T_i$. Each task must be specified as either a *high-critical (*HI*)* or *low-critical (*LO*)* task. We also denote the set of HI-tasks and LO-tasks by $\mathcal{T}_{\text{HI}}$ and $\mathcal{T}_{\text{LO}}$, respectively. That is, $\mathcal{T}_{\text{HI}} \cup \mathcal{T}_{\text{LO}} = \mathcal{T}$ and $\mathcal{T}_{\text{HI}} \cap \mathcal{T}_{\text{LO}} = \emptyset$. We also denote the total utilization of all tasks, HI-task, and LO-tasks, respectively, as follows:

$$U = \sum_{\tau_i \in \mathcal{T}} u_i, \quad U_{\text{HI}} = \sum_{\tau_i \in \mathcal{T}_{\text{HI}}} u_i, \quad U_{\text{LO}} = \sum_{\tau_i \in \mathcal{T}_{\text{LO}}} u_i.$$

We also call a job of a HI-task a HI-job for short and call a job of a LO-task a LO-job for short, respectively. Also, a job is *pending* if it is released and has not completed.

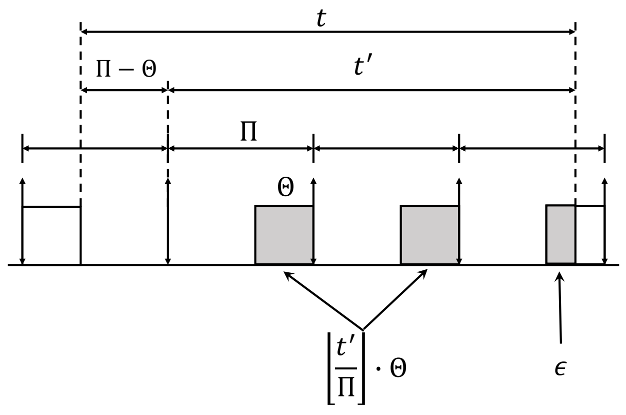


Fig. 2: Worst-case supply of a periodic resource.

Schedulability criteria. Note that, in this paper, two estimates on the budget (Θ^C and Θ^N) for every resource period Π have been applied, while the actual budget during runtime is unknown for the pre-runtime analysis. Therefore, the *schedulability* of the system is defined as

- the deadlines of all (HI- and LO-) tasks must be met, if at least Θ^N available time units are provided as the budget during *every* resource period;
- the deadlines of all HI-tasks must be met, if less than Θ^N (but still at least Θ^C) available time units are provided during *some* resource period.

Utilization-based v.s. demand-based analysis. Under EDF scheduling with virtual deadlines, there are two approaches to analyze conventional MC schedulability problems. One is a utilization-based approach that yields polynomial-time schedulability tests [5, 6]; the other is a demand-based approach that yields pseudo-polynomial-time schedulability tests [33, 34]. In this paper as an initial exploration of the proposed new MC model, we focus on the utilization-based analysis and aims to develop a polynomial-time schedulability test for the problem of MC scheduling with multiple budget estimates.

In the rest of this section, we review algorithm EDF-VD and the periodic resource model in more detail.

A. Algorithm EDF-VD

The scheduling algorithm EDF-VD was first proposed for scheduling implicit-deadline mixed-criticality tasks with *multiple execution time estimates* on a *dedicated* uniprocessor [5, 6], *i.e.*, no budget nor resource period needs to be considered. That is, the original targeted problem by EDF-VD is a completely different scheduling problem from the system model we just introduced earlier in this section. Nonetheless, we will briefly review how EDF-VD works for its originally targeted problem next, in order to illustrate the concept of virtual deadlines that will be adopted to resolve the problem we address in this paper.

Under EDF-VD, each HI-job is assigned a virtual deadline earlier than its actual deadline. Specifically, a scaling factor $0 < x \leq 1.0$ is applied to all HI-tasks so that each HI-task τ_i has a relative virtual deadline $T_i' = x \cdot T_i$. On the other hand, every LO-job is assigned a virtual deadline equal to its actual deadline.

During runtime, EDF-VD starts with scheduling all jobs with respect to their *virtual* deadlines — the earlier the virtual deadline, the higher the priority. If all HI-jobs complete their execution by their less pessimistic execution time estimate, all virtual deadlines and therefore all actual deadlines are guaranteed to be met. If any HI-job executes over its less pessimistic execution time estimate without signaling completion, all LO-jobs are dropped by EDF-VD immediately, and afterward, all HI-jobs are scheduled by EDF with respect to their *actual* deadlines.

The setting of the scaling factor x and utilization-based schedulability tests have been investigated in [5, 6], but we omit the details here due to the differences in the system model.

B. Periodic Resource Model

In the periodic resource model [58], a VP is characterized by two parameters (Π, Θ), which indicate that this VP supplies Θ units of processor time every Π time units, where $0 < \Theta \leq \Pi$.

Note that, a VP corresponding to a dedicated physical processor that is always available is a special case in the periodic resource model where $\Theta = \Pi$.

The *supply bound function (SBF)* of the VP, denoted $\mathsf{sbf}(t)$, indicates the *minimum* processor time this VP can supply during any time interval of length t. Shin and Lee [58] have shown that $\mathsf{sbf}(t)$ can be calculated by

$$\mathsf{sbf}(t) = \begin{cases} 0 & \text{if } t' < 0 \\ \left\lfloor \frac{t'}{\Pi} \right\rfloor \cdot \Theta + \epsilon & \text{if } t' \geq 0 \end{cases}$$

where

$$t' = t - (\Pi - \Theta),$$

$$\epsilon = \max\left(t' - \Pi \left\lfloor \frac{t'}{\Pi} \right\rfloor - (\Pi - \Theta), 0\right).$$

This definition reflects the worst-case scenario illustrated in Fig. 2.

In [58], a *linear* supply bound function, which is a lower-bound on the corresponding supply bound function, is also defined by

$$\mathsf{lsbf}(t) = \frac{\Theta}{\Pi}(t - 2(\Pi - \Theta)).$$

That is, $\forall t, \mathsf{lsbf}(t) \leq \mathsf{sbf}(t)$. Fig. 3 illustrates the functions $\mathsf{sbf}(t)$ and $\mathsf{lsbf}(t)$. On the other hand, under EDF scheduling, the *demand bound function* of a task τ_i is calculated by

$$\mathsf{dbf}(t, \tau_i) = \left\lfloor \frac{t}{T_i} \right\rfloor \cdot C_i,$$

and the demand bound function for a task set $\mathcal{T}$ is

$$\mathsf{dbf}(t, \mathcal{T}) = \sum_{\tau_i \in \mathcal{T}} \mathsf{dbf}(t, \tau_i).$$

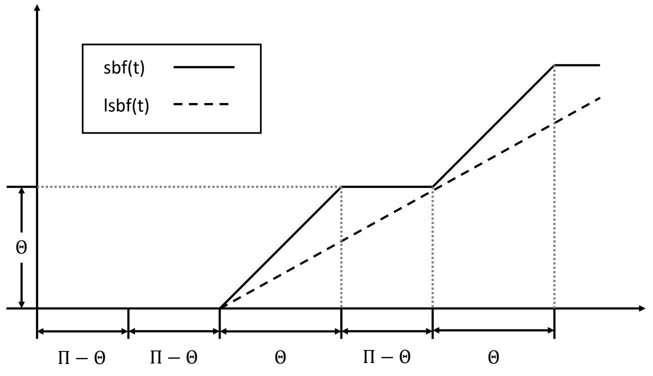


Fig. 3: An illustration for functions $\mathsf{sbf}(t)$ and $\mathsf{lsbf}(t)$.

Linear demand bound functions, which are an upper-bound on the corresponding demand bound functions, are also defined by

$$\mathsf{ldbf}(t, \tau_i) = u_i \cdot t, \text{ and}$$

$$\mathsf{ldbf}(t, \mathcal{T}) = \left(\sum_{\tau_i \in \mathcal{T}} u_i \right) \cdot t.$$

That is, $\forall t$ and $\mathcal{T}$, $\mathsf{dbf}(t, \mathcal{T}) \le \mathsf{ldbf}(t, \mathcal{T})$. Thus, $\mathsf{ldbf}(t, \mathcal{T}) \le \mathsf{lsbf}(t)$ implies $\mathsf{dbf}(t, \mathcal{T}) \le \mathsf{sbf}(t)$. The following lemma and theorem have been shown in [58], and Theorem 1 below in fact provides a utilization-based schedulability test.

Lemma 1. *(Lemma 5 in [58]) Task set $\mathcal{T}$ is schedulable by EDF on a VP (Π, Θ) if $\mathsf{ldbf}(T^{\min}, \mathcal{T}) \le \mathsf{lsbf}(T^{\min})$ where $T^{\min} = \min_{\tau_i \in \mathcal{T}} T_i$.*

Theorem 1. *(Theorem 7 in [58]) Task set $\mathcal{T}$ is schedulable by EDF on a VP (Π, Θ) if*

$$U \le \frac{\Theta}{\Pi} \left(1 - \frac{2(\Pi - \Theta)}{T^{\min}} \right),$$

where $U = \sum_{\tau_i \in \mathcal{T}} u_i$ and $T^{\min} = \min_{\tau_i \in \mathcal{T}} \{T_i\}$.

III. Algorithm EDF-VDVP

In this section, we present our scheduling algorithm EDF-VDVP.[1] For notational simplicity, we denote

$$\gamma^N = \frac{2(\Pi - \Theta^N)}{T^{\min}}, \tag{2}$$

$$\gamma^C = \frac{2(\Pi - \Theta^C)}{T^{\min}_{\text{HI}}}, \tag{3}$$

where

$$T^{\min} = \min_{\tau_i \in \mathcal{T}} \{T_i\}, \text{ and } T^{\min}_{\text{HI}} = \min_{\tau_i \in \mathcal{T}_{\text{HI}}} \{T_i\}.$$

Intuitively, γ^N and γ^C measure the proportional "gap" between the total utilization bound for applying Theorem 1 and the bandwidth in nominal and critical modes, respectively. It is clearly required by the system model that $\Pi \ge \Theta^N \ge \Theta^C$, which implies $\gamma^N \ge 0$ and $\gamma^C \ge 0$. Furthermore, because $2(\Pi - \Theta^N)$ and $2(\Pi - \Theta^C)$, as illustrated in Fig. 2, are the maximum time duration of no available budget for the VP in nominal and critical modes, respectively, it is necessary to require that $2(\Pi - \Theta^N) < T^{\min}$ and $2(\Pi - \Theta^C) < T^{\min}_{\text{HI}}$ as well; otherwise, the potential scenario of the entire scheduling window of a job (*i.e.*, from its release to its deadline) falling into such a time duration of absolutely no available budget clearly prevents that job from meeting its deadline in the worst case. Thus, it is necessary for any feasible system that

$$0 \le \gamma^N < 1 \text{ and } 0 \le \gamma^C < 1 \tag{4}$$

[1]"EDF-VDVP" stands for "EDF with virtual deadlines and varying processor supply."

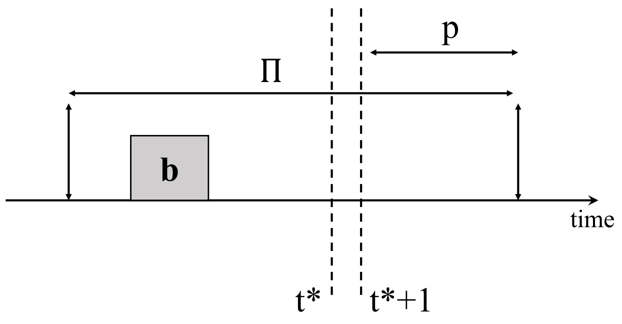


Fig. 4: An illustration for the mode-switch time instant t^*.

Similar to many mixed-criticality scheduling algorithms, EDF-VDVP also has two modes to cope with the two estimates. In many prior work on multiple estimates on execution time, the mode switch point is rather straightforward — when a job has executed for its less pessimistic execution time without signaling completion. In our context, the less pessimistic budget Θ^N is the greater one, and therefore the mode switch cannot be in the same manner as before.

Mode switch. Under EDF-VDVP, there are two modes, namely the *nominal* mode and *critical* mode. The system always starts with the nominal mode. At every time instant, the scheduler keeps track of that 1) b time units budget have already received during the current resource period, and 2) there are still p time units until the end of current resource period. The mode switch must happen when a time unit has no available supply budget, *i.e.*, for some time instant t^*, the time unit $[t^*, t^* + 1)$ is not available to provide any execution for tasks in $\mathcal{T}$. If at time $t^* + 1$ it becomes true that $b + p < \Theta^N$, then EDF-VDVP notices that a mode switch to the critical mode is inevitable and therefore switches to critical-mode scheduling immediately. We define t^* (the time instant followed by an unavailable time unit) as the *mode-switch* time instant. Note that, the scheduler detecting the mode switch at time instant t^* or $t^* + 1$ does not affect any scheduling decision as $[t^*, t^* + 1)$ is an unavailable time unit anyway. Fig. 4 illustrates the mode switch time instant t^*.

Scheduling with virtual deadlines. Besides the differences in the assumed system model, EDF-VDVP schedules tasks in the same virtual-deadline-based manner as EDF-VD does. Every job of HI-task τ_i has a virtual deadline $T_i' = x \cdot T_i$ time units after its release, where x is the system-wide virtual deadline scaling factor such that $0 < x \leq 1$. In contrast, the virtual deadlines of LO-tasks is the same as their actual deadlines. In the runtime, the EDF-VDVP always begins with nominal mode where the pending job with the earliest *virtual* deadline is scheduled for execution whenever available processing time is allocated for the VP. When a mode switch to the critical mode is triggered, EDF-VDVP discards any pending and upcoming LO-jobs, and afterwards, schedules the pending HI-job with earliest *actual* deadline for execution whenever available processing time is allocated for the VP.

Calculating the scaling factor x. For a given task set $\mathcal{T}$ and given parameters Π, Θ^N, the scaling fact x is calculated by

$$x = \frac{U_{\text{HI}} + w^N \gamma^N}{w^N - U_{\text{LO}}}. \tag{5}$$

In any feasible system where $\mathcal{T}_{\text{HI}}$ is not empty, it is clearly necessary that $w^N \geq U > U_{\text{LO}}$, which implies $x > 0$. On the other hand, we further require that $x \leq 1$, which is validated by the schedulability test as presented in Theorem 2 in the next section.

Moreover, to further understand the implications behind the requirement of $x \leq 1$ in the context, it might be worth noting that the requirement of $x \leq 1$ can also be implied by an assumption that the system is deemed schedulable by Theorem 1 in nominal mode when no mode switch is triggered, because $U_{\text{HI}} + U_{\text{LO}} \leq w^N \cdot \left(1 - \gamma^N\right) \implies x \leq 1$.

IV. Schedulability Test

In this section, we present a sufficient schedulability test for EDF-VDVP. The schedulability test needs to validate the guarantees in the two modes corresponding to the nominal and critical budgets, respectively, so that the schedulability criteria defined in Sec. II are met.

The following lemma shows that with x being set by (5), the deadlines of all (HI- and LO-) tasks are met in the nominal mode, since the virtual deadline of any job is no later than its actual deadline.

Lemma 2. *Given that $0 < x \leq 1$, all virtual deadlines of HI- and LO-tasks are met in the nominal mode if*

$$x \geq \frac{U_{\text{HI}} + w^N \gamma^N}{w^N - U_{\text{LO}}}.$$

Proof. In the nominal mode, treating the virtual deadlines as the actual deadlines, every HI-task τ_i can be viewed as a sporadic task τ_i' with a shorter period (and shorter relative deadline) $T_i' = x \cdot T_i$. Therefore,

$$u_i' = \frac{C_i}{T_i'} = \frac{C_i}{x \cdot T_i} = \frac{u_i}{x}.$$

On the other hand, the period, deadline, and therefore utilization of every LO-task remain unchanged. That is, each task $\tau_i \in \mathcal{T}$ can be viewed as a sporadic task with period T_i' such that

$$T_i' = \begin{cases} T_i, \text{ if } \tau_i \in \mathcal{T}_{\text{LO}} \\ x \cdot T_i, \text{ if } \tau_i \in \mathcal{T}_{\text{HI}}, \end{cases}$$

and therefore, given that $0 < x \leq 1$,

$$\begin{aligned} &\forall i, T_i' \geq x \cdot T_i \\ \implies &\min_{\tau_i \in \mathcal{T}} \{T_i'\} \geq x \cdot T^{\min} \\ \implies &\frac{2(\Pi - \Theta^N)}{\min_{\tau_i \in \mathcal{T}} \{T_i'\}} \leq \frac{2(\Pi - \Theta^N)}{x \cdot T^{\min}} \\ \implies &1 - \frac{2(\Pi - \Theta^N)}{\min_{\tau_i \in \mathcal{T}} \{T_i'\}} \geq 1 - \frac{2(\Pi - \Theta^N)}{x \cdot T^{\min}}. \end{aligned} \tag{6}$$

Furthermore, the budget supply in the nominal mode follows the periodic resource model with parameters (Π, Θ^N). Thus, by Theorem 1, the virtual deadlines of all tasks are met if

$$\begin{aligned} &\sum_{\tau_i \in \mathcal{T}_{\text{LO}}} u_i + \sum_{\tau_i \in \mathcal{T}_{\text{HI}}} \frac{u_i}{x} \leq \frac{\Theta^N}{\Pi} \left(1 - \frac{2(\Pi - \Theta^N)}{\min_{\tau_i \in \mathcal{T}} \{T_i'\}}\right) \\ \overset{\text{by (6)}}{\Longleftarrow} &\sum_{\tau_i \in \mathcal{T}_{\text{LO}}} u_i + \sum_{\tau_i \in \mathcal{T}_{\text{HI}}} \frac{u_i}{x} \leq \frac{\Theta^N}{\Pi} \left(1 - \frac{2(\Pi - \Theta^N)}{x \cdot T^{\min}}\right) \\ \iff &U_{\text{LO}} + \frac{U_{\text{HI}}}{x} \leq w^N \cdot (1 - \frac{\gamma^N}{x}) \\ \iff &x \geq \frac{U_{\text{HI}} + w^N \gamma^N}{w^N - U_{\text{LO}}}. \end{aligned}$$

The lemma follows. □

Furthermore, the following lemma provides a sufficient condition for all actual deadlines of HI-tasks being met in the critical mode.

Lemma 3. *Given that $0 < x \leq 1$, all actual deadlines of HI-tasks in the critical mode will be met if*

$$x \leq 1 - \frac{U_{\text{HI}} + w^C \gamma^C}{w^C}.$$

Proof. At the mode switch point from the nominal to critical mode, a job from any task $\tau_i \in \mathcal{T}_{\text{HI}}$ must either be completed or have its virtual deadline at or after the mode switch point, because by Lemma 2, all virtual deadlines of HI-jobs are met in the nominal mode. That is, if not completed yet, a job of a HI-task τ_i must have an actual deadline at least $(1 - x)T_i$ time units after this mode-switch point. Afterwards, any job from any task $\tau_i \in \tau_{\text{HI}}$ has at least T_i, which is greater than $(1 - x)T_i$ (as $0 < x < 1.0$), time units from their releases in the HI-mode to their corresponding deadlines.

That is, in the critical mode, every HI-task τ_i can be viewed as a sporadic task τ_i' with a shorter period (and shorter relative deadline) $T_i' = (1 - x) \cdot T_i$. Therefore,

$$u_i' = \frac{C_i}{T_i'} = \frac{C_i}{(1 - x) \cdot T_i} = \frac{u_i}{1 - x}.$$

Furthermore, the budget supply in the critical mode follows the periodic resource model with parameters (Π, Θ^C). Please note that (Π, Θ^C) specification has been actually maintained since time 0, as Θ^C denotes the *minimum* budget in each resource period, Therefore, any time intervals, including the ones in the critical mode can be viewed as a certain time interval under periodic resource model (Π, Θ^C). Thus, by Theorem 1, the actual deadlines of all HI-tasks are met in the critical mode if

$$\begin{aligned}
&\sum_{\tau_i \in \mathcal{T}_{\text{HI}}} u'_i \leq \frac{\Theta^C}{\Pi}\left(1 - \frac{2(\Pi - \Theta^C)}{\min_{\tau_i \in \mathcal{T}_{\text{HI}}}\{T'_i\}}\right) \\
\iff &\sum_{\tau_i \in \mathcal{T}_{\text{HI}}} u'_i \leq \frac{\Theta^C}{\Pi}\left(1 - \frac{2(\Pi - \Theta^C)}{(1-x)\cdot T^{\min}_{\text{HI}}}\right) \\
\iff &\frac{U_{\text{HI}}}{1-x} \leq w^C \cdot \left(1 - \frac{\gamma^C}{1-x}\right) \\
\iff &1 - x \geq \frac{U_{\text{HI}} + w^C\gamma^C}{w^C} \\
\iff &x \leq 1 - \frac{U_{\text{HI}} + w^C\gamma^C}{w^C}
\end{aligned}$$

Note that the second last step is because $0 < x < 1.0$. The lemma follows. □

By Lemmas 2 and 3, the following theorem holds and serves as a sufficient schedulability test.

Theorem 2. *A mixed-criticality task set $\mathcal{T}$ is schedulable by EDF-VDVP on a VP with resource period Π, nominal budget Θ^N, and critical budget Θ^C, if*

$$\frac{U_{\text{HI}} + w^N\gamma^N}{w^N - U_{\text{LO}}} + \frac{U_{\text{HI}} + w^C\gamma^C}{w^C} \leq 1. \tag{7}$$

Proof. Because it is clear that $\frac{U_{\text{HI}}+w^C\gamma^C}{w^C} > 0$, (7) implies $x < 1$ as x is defined by (5). Therefore, an additional explicit requirement of $x \leq 1$ as discussed in Sec. III is redundant and can be omitted in this theorem. Furthermore, $0 < x \leq 1$ implies the fact that the virtual deadline of every job is at or before its actual deadline. Thus, by Lemmas 2 and 3, the theorem follows. □

V. Speed-up Bound

In this section, we derive a *speed-up bound* for EDF-VDVP, given its schedulability test as presented by Theorem 2. The interpretation of speed-up bound in the context of this paper is as follows.

Speed-up bound. A scheduling algorithm $\mathcal{A}$ has a speed-up bound of ρ if and only if any task set that is schedulable by some (potentially optimal) algorithm on a **unit-speed** processor with the resource model 3-tuple $(\Theta^C, \Theta^N, \Pi)$ must also be schedulable by algorithm $\mathcal{A}$ on a **speed-**ρ processor with the same resource model 3-tuple.

Please note that each of Θ^C, Θ^N, and Π is defined by *time units* while the amount of work completed by consuming one time unit budget is proportionally scaled by the processor speed. As a result, please note that Θ^C, Θ^N, Π, and therefore γ^C and γ^N, are all independent from the processor speed. Also, please note that we focus on MC systems where $\mathcal{T}_{\text{HI}} \neq \emptyset$; otherwise, it reduces to a conventional non-MC system.

Limitation. Our results in this section about the speed-up bound for EDF-VDVP rely on the assumption that

$$\gamma^N + \gamma^C < 1, \tag{8}$$

which is more strict than (4) that is necessarily required in any feasible MC system. In other words, the speed-up bound in this section only applies to systems that satisfy (8). Nonetheless, (8) was *not* required in the schedulability test derived earlier in Sec. IV, and therefore the schedulability test itself applies to a larger set of MC systems than the set of MC systems to which the speed-up bound in the section applies. Also, please note that (8) does not impose restrictions on task set utilizations — given an arbitrary feasible task set and VP bandwidth, (8) only implies that the resource period (*i.e.*, Π), or more precisely the maximum length of time interval of no budget (*i.e.*, $2(\Pi-\Theta^N)$ and $2(\Pi-\Theta^C)$), needs to be sufficiently small in comparison to the task periods. For example, while not necessary, requiring $\Pi \leq T^{\min}/4$ is able to sufficiently guarantee (8) holds.

In order to derive a speed-up bound for EDF-VDVP, we establish the following preliminary lemma first.

Lemma 4. *In any feasible MC system, the following inequality holds for all $0 < s \leq 1 - \gamma^N$ that*

$$\frac{w^N \cdot (s + \gamma^N) - U_{\text{LO}}}{w^N - U_{\text{LO}}} \leq s + \gamma^N.$$

Proof. In a feasible MC system, the total utilization of tasks in LO-mode must not exceed the bandwidth of the VP in LO-mode. That is, it must hold that $U = U_{\text{LO}} + U_{\text{HI}} \leq w^N$. Since $\mathcal{T}_{\text{HI}} \neq \emptyset \implies U_{\text{HI}} > 0$, we then have $0 \leq U_{\text{LO}} < w^N$, which implies $w^N - U_{\text{LO}} > 0$. Therefore,

$$\begin{aligned}
&\frac{w^N \cdot (s + \gamma^N) - U_{\text{LO}}}{w^N - U_{\text{LO}}} \leq s + \gamma^N \\
\iff &-U_{\text{LO}} \leq -U_{\text{LO}} \cdot (s + \gamma^N) \\
\Longleftarrow\ &1 \geq s + \gamma^N \\
\Longleftarrow\ &0 < s \leq 1 - \gamma^N.
\end{aligned}$$

Thus, the lemma follows. □

Then, we prove the following lemma, which will directly imply a speed-up bound for EDF-VDVP by scaling speed unit by $1/s$.

Lemma 5. *Any task set that is schedulable by some (potentially optimal) algorithm on a* ***speed-**s *processor* $(0 < s < 1)$ *with the resource model 3-tuple* $(\Theta^C, \Theta^N, \Pi)$ *must also be schedulable by EDF-VDVP on a* ***unit-speed*** *processor with the same resource model 3-tuple, if*

$$s \leq \frac{1 - \gamma^N - \gamma^C}{2}. \tag{9}$$

Proof. The goal of the proof is to show that the *necessary* conditions for a task set $\mathcal{T}$ to be schedulable (by any algorithm) on a speed-s processor will *sufficiently* imply that this task set $\mathcal{T}$ is schedulable by EDF-VDVP on a unit-speed processor, by applying Theorem 2.

For the schedulability on a speed-s processor, it is necessary that in each of nominal and critical modes, the total utilization of the tasks does not exceed the computing capacity to be provided in the long-term, where the latter is calculated by the VP bandwidth multiplying the processor speed. That is, it is necessarily true that

$$U \leq w^N \cdot s \quad \text{and} \quad U_{\text{HI}} \leq w^C \cdot s. \tag{10}$$

Therefore, we have

$$\begin{aligned} & \frac{U - U_{\text{LO}} + w^N \gamma^N}{w^N - U_{\text{LO}}} + \frac{U_{\text{HI}} + w^C \gamma^C}{w^C} \leq 1 \\ \overset{\{\text{by (10)}\}}{\Longleftarrow} \quad & \frac{w^N \cdot s - U_{\text{LO}} + w^N \gamma^N}{w^N - U_{\text{LO}}} + \frac{w^C \cdot s + w^C \gamma^C}{w^C} \leq 1 \\ \Longleftrightarrow \quad & \frac{w^N \cdot (s + \gamma^N) - U_{\text{LO}}}{w^N - U_{\text{LO}}} + s + \gamma^C \leq 1. \end{aligned}$$

By (9), $s \leq \frac{1-\gamma^N}{2} \leq 1 - \gamma^N$. Therefore, by Lemma 4, we have

$$\begin{aligned} & \frac{w^N \cdot (s + \gamma^N) - U_{\text{LO}}}{w^N - U_{\text{LO}}} + s + \gamma^C \leq 1 \\ \Longleftarrow \quad & s + \gamma^N + s + \gamma^C \leq 1 \\ \Longleftrightarrow \quad & s \leq \frac{1 - \gamma^N - \gamma^C}{2}. \end{aligned}$$

That is, by combining the above two steps of derivation, we have

$$\begin{aligned} & s \leq \frac{1 - \gamma^N - \gamma^C}{2} \\ \Longrightarrow & \frac{U - U_{\text{LO}} + w^N \gamma^N}{w^N - U_{\text{LO}}} + \frac{U_{\text{HI}} + w^C \gamma^C}{w^C} \leq 1, \end{aligned} \tag{11}$$

where (11) is in fact identical to (7).

Thus, by Theorem 2, the system must be schedulable under EDF-VDVP on a unit-speed processor, and the lemma follows. □

By Lemma 5 above, scaling up the speed unit by $1/s$ yields the following theorem for a speed-up bound for EDF-VDVP when (8) holds.

Theorem 3. *For all systems where* $\gamma^N + \gamma^C < 1$, *algorithm EDF-VDVP has a speed-up bound of*

$$\frac{2}{1 - \gamma^N - \gamma^C}.$$

Proof. By scaling up the speed unit by $1/s$ in Lemma 5, we have the following statement. Thus, the lemma follows.

> Any task set that is schedulable by some (potentially optimal) algorithm on a **unit-speed** processor with the resource model 3-tuple $(\Theta^C, \Theta^N, \Pi)$ must also be schedulable by EDF-VDVP on a **speed-**$(1/s)$ processor $(0 < s < 1)$ with the same resource model 3-tuple, if s satisfies (9).

Given that $\gamma^N + \gamma^C < 1$, $s \leq \frac{1-\gamma^N-\gamma^C}{2}$, *i.e.*, (9), is equivalent to $\frac{1}{s} \geq \frac{2}{1-\gamma^N-\gamma^C}$. Thus, the theorem follows. □

Clairvoyance v.s. non-clairvoyance. It may be worth noting that the necessary feasibility conditions used in the speed-up bound derivation in this section are fundamental ones that even a *clairvoyant*[2] scheduler must respect as well. Therefore, the speed-up bound for EDF-VDVP, which is a non-clairvoyant scheduler, is established with respect to any (clairvoyant or non-clairvoyant) schedulers. It is interesting to investigate in future work whether a better speed-up bound with respect to non-clairvoyant schedulers only could be achieved by exploring tighter necessary feasibility conditions for non-clairvoyance scheduling.

VI. Setting Resource Period

In the prior sections, we have focused on the problem of determining whether a given task set is schedulable when the resource model 3-tuple $(\Theta^C, \Theta^N, \Pi)$ is also given. In this section, we view this problem from another angle, as a system design problem of selecting a proper resource period Π.

In general, a larger resource period Π is desirable in practice for less overheads on budget replenishment and preemptions due to budget exhausting. On the other hand, a larger resource period Π can jeopardize the schedulability due to the looser budget guarantees implied by the larger resource period, even if the long-term bandwidth is preserved.

In particular, we study the optimization problem of maximizing the resource period Π with given nominal bandwidth w^N and critical bandwidth w^C while the schedulability is required. Please note that, given the fixed values of w^N and w^C, the nominal and critical budgets are contingent on the resource period Π and can be derived by

$$\Theta^N = \Pi \cdot w^N \quad \text{and} \quad \Theta^C = \Pi \cdot w^C. \tag{12}$$

Therefore, by (2) and (3), we have

$$\gamma^N = \frac{2(\Pi - \Theta^N)}{T^{\min}} = \frac{2(1 - w^N)}{T^{\min}} \Pi, \tag{13}$$

$$\gamma^C = \frac{2(\Pi - \Theta^C)}{T^{\min}_{\text{HI}}} = \frac{2(1 - w^C)}{T^{\min}_{\text{HI}}} \Pi, \tag{14}$$

where

$$T^{\min} = \min_{\tau_i \in \mathcal{T}} \{T_i\}, \text{ and } T^{\min}_{\text{HI}} = \min_{\tau_i \in \mathcal{T}_{\text{HI}}} \{T_i\}.$$

Furthermore, by definition, any bandwidth on a uniprocessor cannot exceed 1 and any overutilization in the long term must result in non-schedulability, and we restrict our attention to non-trivial MC systems where $\mathcal{T}_{\text{HI}}$ is not empty and $w^C < w^N$. Therefore, it is clearly necessary that

$$U_{\text{LO}} < U \leq w^N \leq 1 \text{ and } 0 < U_{\text{HI}} \leq w^C < w^N \leq 1. \tag{15}$$

The following theorem provides a range for selecting the resource period Π in order to guarantee the system to be schedulable. In the case of the following range for Π is empty,

[2]A clairvoyant scheduler is able to "look into the future" and know the exact budget allocation for the entire schedule at the beginning of the scheduling.

i.e., $1 - \frac{U_{\mathrm{HI}}}{w^N - U_{\mathrm{LO}}} - \frac{U_{\mathrm{HI}}}{w^C} \leq 0$, the implications are that no such a resource period can be found by the presented approach.

Theorem 4. *Considering the problem of scheduling the MC task set $\mathcal{T}$ on a uniprocessor VP and given the values of nominal bandwidth w^N and critical bandwidth w^C, the system is schedulable under EDF-VDVP if*

$$0 < \Pi \leq \frac{1 - \frac{U_{\mathrm{HI}}}{w^N - U_{\mathrm{LO}}} - \frac{U_{\mathrm{HI}}}{w^C}}{\frac{2w^N(1-w^N)}{(w^N - U_{\mathrm{LO}})T^{\min}} + \frac{2(1-w^C)}{T_{\mathrm{HI}}^{\min}}},$$

where

$$T^{\min} = \min_{\tau_i \in \mathcal{T}} \{T_i\}, \text{ and } T_{\mathrm{HI}}^{\min} = \min_{\tau_i \in \mathcal{T}_{\mathrm{HI}}} \{T_i\}.$$

Proof. By Theorem 2, (7) sufficiently guarantees the schedulability of the MC system under EDF-VDVP. Also, by (13) and (14), it is true that (7) is equivalent to

$$\left(\frac{2w^N(1-w^N)}{(w^N - U_{\mathrm{LO}})T^{\min}} + \frac{2(1-w^C)}{T_{\mathrm{HI}}^{\min}}\right)\Pi \leq 1 - \frac{U_{\mathrm{HI}}}{w^N - U_{\mathrm{LO}}} - \frac{U_{\mathrm{HI}}}{w^C}. \tag{16}$$

By (15), it is clear that

$$\frac{2w^N(1-w^N)}{(w^N - U_{\mathrm{LO}})T^{\min}} + \frac{2(1-w^C)}{T_{\mathrm{HI}}^{\min}} > 0.$$

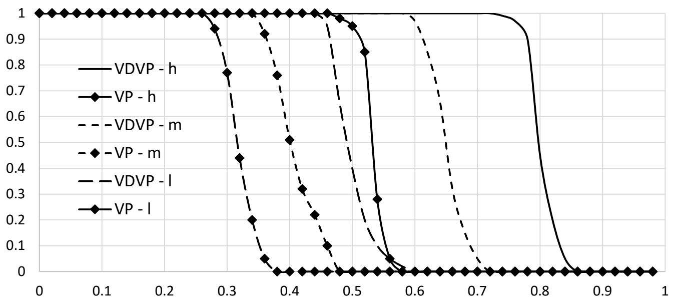
Therefore, (16) is equivalent to

$$\Pi \leq \frac{1 - \frac{U_{\mathrm{HI}}}{w^N - U_{\mathrm{LO}}} - \frac{U_{\mathrm{HI}}}{w^C}}{\frac{2w^N(1-w^N)}{(w^N - U_{\mathrm{LO}})T^{\min}} + \frac{2(1-w^C)}{T_{\mathrm{HI}}^{\min}}},$$

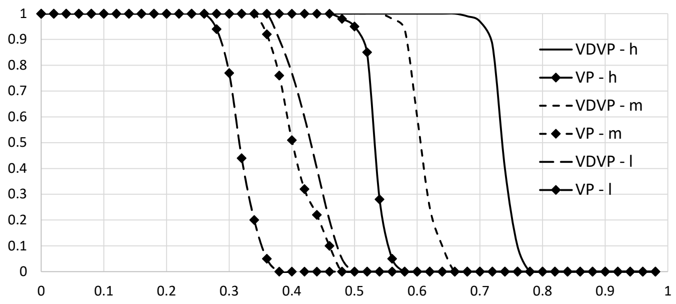
and the theorem follows. □

VII. Evaluation

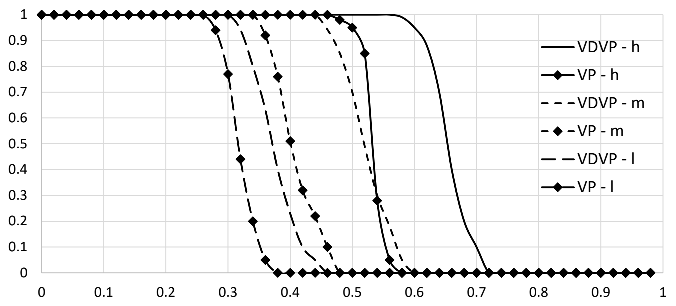
We have conducted extensive simulations using synthesized task sets to evaluate our proposed approaches. Our goal is to examine the effectiveness of the proposed model and algorithm EDF-VDVP along with its schedulability test. Specifically, we evaluate our MC schedulability analysis under EDF-VDVP (this analysis is denoted by "VDVP") given its schedulability test in Theorem 2 and compare it against the conventional non-MC schedulability analysis under EDF in a compositional real-time system (this analysis is denoted by "VP") by Theorem 1 in [58]. Especially, in the case of "VP," because of the lack of the notion of tasks' criticalities, every deadline of all tasks must be met in all scenarios. Furthermore, since only one budget estimate Θ is provided in the conventional non-MC periodic resource model [58] and it models the *minimum* budget for *any* resource period, it must set $\Theta = \Theta^C \leq \Theta^N$ to cover the worst-case budget scenario with the single budget estimate Θ. In order to compare the schedulability results provided by VDVP and VP, the metric is to measure the acceptance ratio of the above tests with respect to a given goal of task set utilization level U. The experimental results demonstrated in this section were performed on a PC, enabling figures to be obtained for large numbers of randomly generated task sets.



(a) $\frac{|\mathcal{T}_{\mathrm{HI}}|}{|\mathcal{T}_{\mathrm{LO}}|} = \frac{1}{3}$



(b) $\frac{|\mathcal{T}_{\mathrm{HI}}|}{|\mathcal{T}_{\mathrm{LO}}|} = 1$

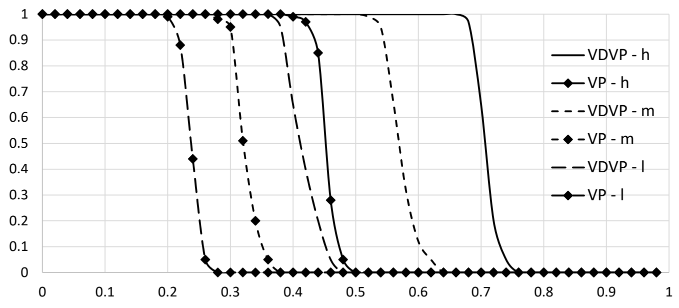


(c) $\frac{|\mathcal{T}_{\mathrm{HI}}|}{|\mathcal{T}_{\mathrm{LO}}|} = 3$

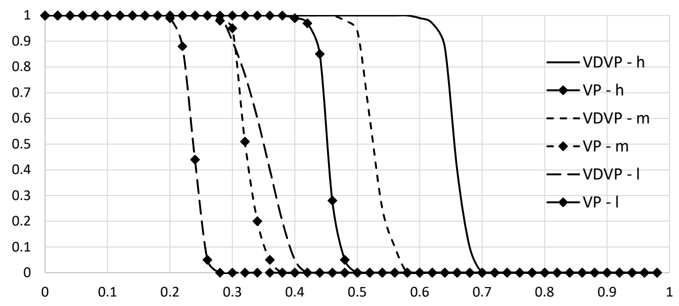
Fig. 5: Schedulability results when the resource period Π is small.

A. Simulation Setup

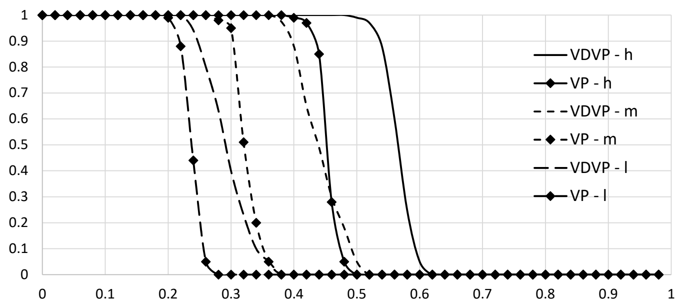
The task parameters are randomly generated as follows: of the n tasks in each task set, $\frac{n}{K}$ tasks were assigned to each of the K order of magnitude ranges identified (e.g., when $K = 4$, 4 different ranges are selected, i.e., $[1, 10]$, $[10, 100]$, $[100, 10000]$, $[1000, 10000]$). Task periods were determined based on a uniform random distribution, according to each assigned range. This period generation method is proposed to replicate the type of period distributions found in commercial real-time systems (by varying K from 2 to 6 [22]). In all task sets, task deadlines were set equal to their periods. For each utilization level studied, the UUniFast-Discard method [14] is implemented to determine individual task utilization u_i and each task's execution time was set accordingly, i.e., $C_i = T_i \times u_i$, given the previously selected task periods. A total of 100,000 task sets were generated in all experiments. A similar task sets generation approach is used by Davis and



(a) $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|} = \frac{1}{3}$

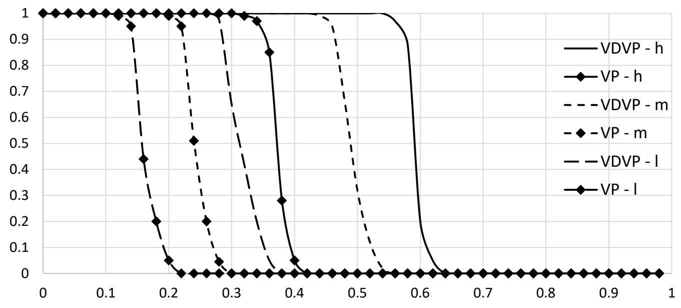


(b) $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|} = 1$

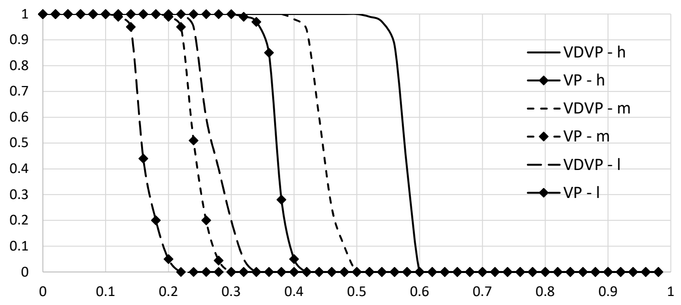


(c) $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|} = 3$

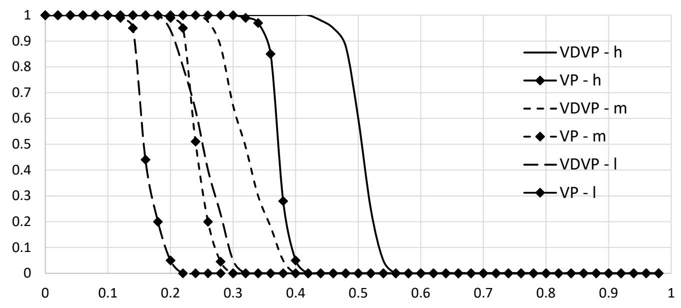
Fig. 6: Schedulability results when the resource period Π is medium.



(a) $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|} = \frac{1}{3}$



(b) $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|} = 1$



(c) $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|} = 3$

Fig. 7: Schedulability results when the resource period Π is large.

Burns [22]. The value of $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|}$ to control the ratio between the number of high critical tasks and the number of low critical tasks is also parameterized in evaluations. (For example, $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|}$ = $\frac{1}{3}$, 1 and 3.) Similarly, the value of $\frac{\Theta}{\Pi}$ denotes the percentage of time units which can be utilized to execute real-time tasks on the virtual processor. $\frac{\Theta}{\Pi}$ are distributed using three uniform distributions: $[0.2, 0.4]$ (the bandwidth of virtual processor is low), $(0.4, 0.6]$ (medium), and $(0.6, 0.8]$ (high). Since the analytical schedulability results of both VDVP and VP are impacted by the specific values of Θ and Π as well, Π is distributed using three uniform distributions: $[1, 10]$ (The resource period of the virtual processor is small), $(10, 100]$ (medium), and $(100, 1000]$ (large). Furthermore, the budget Θ can be calculated from the ratio and the resource period Π. Each such task set was generated by creating tasks until total utilization exceeded the corresponding utilization cap, and by then reducing the last task's utilization so that the total utilization equalled the utilization cap.

B. Schedulability Results

In all graphs, the x-axis denotes the task set utilization cap and the y-axis denotes the fraction of generated task sets that were schedulable. In Fig. 5 (respectively, Fig. 6 and Fig. 7), the resource period of the virtual processor Π is set to be small (respectively, medium and large). In each figure, three graphs are presented, where $\frac{|\mathcal{T}_{\rm HI}|}{|\mathcal{T}_{\rm LO}|} = \frac{1}{3}$, 1 and 3 are assumed. Each graph gives three curves per tested approach for the cases of high, medium, and low bandwidth of the virtual processor, respectively. As seen in each graph, the label "VDVP-h (m/l)" indicates the approach of our utilization-based test assuming high (medium/low) bandwidth of the virtual processor. Similarly, "VP" labels are used to denote the test given in Theorem 1 in [58] under three scenarios.

The obtained schedulability results are shown in Fig. 5 - Fig. 7. Each curve plots the fraction of the generated task sets successfully scheduled by the corresponding approach, as a function of total utilization. As we mentioned earlier, all tasks in each task set are not differentiated with criticalities in the case of "VP" and every deadline of all tasks must be met in all scenarios, thus the three curves labeled by "VP" in three figures remain the same.

From the experimental results, we can see that, in all tested scenarios, VDVP achieves the best performance, in many cases improving upon VP by a substantial margin. For example, as seen in Fig. 5(b), when the number of high critical tasks is equal to the number of low critical tasks and the utilization of the virtual processor is high, VDVP can achieve 100% schedulability when U equals 0.66 while VP fails to do so when U merely exceeds 0.46. Note that when $\frac{|\mathcal{T}_{\text{HI}}|}{|\mathcal{T}_{\text{LO}}|}$ is smaller, the improvement margin by VDVP over VP increases. This is because in this case, under MCS model, if more low critical tasks are discarded in critical mode, it will increase the chance for the high criticality tasks to be schedulable. Another interesting observation is when more low critical tasks are involved, the system is easier to be schedulable under VDVP. Because fewer high critical tasks are scheduled concurrently with the same amount of supplied computing capacity in the high criticality mode. Also in each graph, we observe that all six tests perform better under higher bandwidth of the virtual processor. This is because when the bandwidth is larger, more processing time is available for executing the task system, which clearly helps all six tests achieve higher schedulability. Additionally, when the resource period of the virtual processor is small, the task set seems easier to be schedulable. The intuitive reason is that when the resource is more frequently scheduled, the resource may obtain better utilization when serving the real-time tasks. On average, VDVP yields an over 30% improvement w.r.t. schedulability compared to VP.

VIII. Related work

In the last decade, multicore processors have become ubiquitous and there has been extensive research work on how to efficiently utilize these parallel machines with different types of real-time tasks, including mixed-criticality real-time task scheduling and compositional real-time task scheduling.

The first work on the verification of a mixed-criticality system was published by Vestal (of Honeywell Aerospace) in 2007 [64]. It used an extension of standard fixed-priority (FP) real-time scheduling theory and proposed a restrictive work-flow model, focused on a single processor and made use of Response Time Analysis [3]. It showed that neither rate monotonic [50] nor deadline monotonic [49] priority assignment is optimal for MCS; however, Audsley's optimal priority assignment algorithm [4] was found to be applicable. This paper was followed by two publications in 2008 by Baruah and Vestal [11], and Huber et al. [45]. Later on, EDF-VD scheduling of MC tasks has been further investigated [29, 34, 72].

Beside the original mixed-criticality task model, a bunch of techniques addressing mixed-criticality systems in different scenarios are proposed: letting any LO-criticality job that has started, run to completion [12]; reducing the priorities of the LO-criticality tasks [7], or similar with EDF scheduling [44]; increasing the periods and deadlines of LO-criticality jobs [37, 46, 57, 60, 61, 62], called task stretching, the elastic task model or multi-rate; decreasing the computation times of some or all of the LO-criticality tasks [19], perhaps by utilising an imprecise mixed-criticality (IMC) model [43] or budget control [39]; moving some LO-criticality tasks to a different processor that has not experienced a criticality mode change [65]; improving resource utilization while guaranteeing safe execution of critical applications [38, 40, 47]. Furthermore, mixed-criticality scheduling involving varying-speed processors [9, 10, 13, 69] and graceful degradation [8, 36, 41, 42, 51, 71] has also been studied.

There also has been extensive research on compositional real-time scheduling. Insik Shin and Insup Lee present a formal description of compositional real-time scheduling problems in [58, 59]. They identify issues that need to be addressed by solutions and provide their framework for the solutions, which is based on the periodic interface. Following this work, [30] introduces the Explicit Deadline Periodic (EDP) resource model, and present compositional analysis techniques under EDF and DM. It shows that these techniques are bandwidth optimal, in that they do not incur any bandwidth overhead in abstraction or composition. ARINC specification 653-2 [32] describes the interface between application software and underlying middleware in a distributed real-time avionics system. Authors develop compositional techniques for automated scheduling of partitions and processes in such systems. This work is followed by [20]. It proposes a compositional approach to formal specification and schedulability analysis of real-time applications running under a Time Division Multiplexing (TDM) global scheduler and preemptive Fixed Priority (FP) local schedulers, according to the ARINC-653 standard.

In addition to the above initial work on compositional real-time systems, several novel scheduling algorithms[30, 35, 54] are proposed to schedule real-time task systems under different system architectures [2, 18, 21, 24, 25, 26, 27, 28, 48, 55, 56, 66, 67, 68]. Correspondingly, to validate the schedulability of each real-time task system, schedulability analysis frameworks [1, 30, 31] are proposed for analyzing real-time tasks in different real-time applications. Furthermore, when real-time task re-weighting techniques [15, 16, 17, 52, 53] are applied to the server tasks at the supply level of each component, it may result in the VP of a component to have varying budgets.

IX. Conclusion

In this paper, we have made efforts to extend the fundamental ideas and frameworks of MC scheduling to a new dimension in the context of compositional real-time systems. We proposed to use multiple parameters to provision the VP supply. To illustrate this approach, we form up a specific scheduling problem in the context of a single virtual processor characterizing by the periodic resource model and propose a virtual-deadline-based algorithm to solve it. The proposed algorithm is supported by a polynomial-time schedulability test and a speed-up bound. In the context of bandwidth and task set are given, we have also derived a range for setting the resource period to ensure schedulability. Moreover, we have conducted schedulability studies for evaluation, and our simulation experiment results demonstrate the effectiveness of our proposed model and algorithm.

Future Work. The work in this paper can be further extended in several ways. First, in addition to implicit-deadline tasks, constrained- and arbitrary-deadline may be considered. Second, we plan to incorporate multiple budget estimates with the classic multiple WCET estimates together in MC scheduling. Also, in addition to the resource budget, multiple estimates on the resource period are another possible dimension and need further investigation. Finally, multiprocessor extension may need more efforts due to the complicated supply analysis induced by parallelism but is definitely an interesting topic.

Acknowledgment

This work is supported by start-up and REP grants from Texas State University and a start-up grant from Wayne State University.

Preliminary partial results of this work have been presented in the workshop WMC 2019 [70]. For completeness, this paper and [70] may have overlaps in presentation, most of which concerning the motivation, background, and system model. In particular, entire Secs. V, VI, VII have not appeared in [70].

We would like to thank the anonymous reviewers for their detailed and helpful comments. We are especially grateful to an anonymous reviewer for an earlier manuscript of this work who pointed out a flaw in the schedulability analysis inherited from [70] so that we were able to correct it in this paper.

We would also like to thank the organizers and participants in Dagstuhl Seminar 19101 for the valuable discussions.

References

[1] Madhukar Anand, Arvind Easwaran, Sebastian Fischmeister, and Insup Lee. Compositional feasibility analysis of conditional real-time task models. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 391–398. IEEE, 2008.

[2] Madhukar Anand, Sebastian Fischmeister, and Insup Lee. Composition techniques for tree communication schedules. In *19th Euromicro Conference on Real-Time Systems (ECRTS'07)*, pages 235–246. IEEE, 2007.

[3] Neil Audsley, Alan Burns, Mike Richardson, Ken Tindell, and Andy J Wellings. Applying new scheduling theory to static priority pre-emptive scheduling. *Software Engineering Journal*, 8(5):284–292, 1993.

[4] Neil C Audsley, Alan Burns, Robert I Davis, Ken W Tindell, and Andy J Wellings. Fixed priority pre-emptive scheduling: An historical perspective. *Real-Time Systems*, 8(2-3):173–198, 1995.

[5] S. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, S. Van Der Ster, and L. Stougie. The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems. In *Proceedings of the 24th Euromicro Conference on Real-Time Systems*, pages 145–154, 2012.

[6] S. Baruah, V. Bonifaci, G. D'Angelo, A. Marchetti-Spaccamela, S. Van Der Ster, and L. Stougie. Mixed-criticality scheduling of sporadic task systems. In *Proceedings of the 19th Annual European Symposium on Algorithms*, pages 555–566, 2011.

[7] Sanjoy Baruah and Alan Burns. Implementing mixed criticality systems in ada. In *International Conference on Reliable Software Technologies*, pages 174–188. Springer, 2011.

[8] Sanjoy Baruah, Alan Burns, and Zhishan Guo. Scheduling mixed-criticality systems to guarantee some service under all non-erroneous behaviors. In *2016 28th Euromicro Conference on Real-Time Systems (ECRTS)*, pages 131–138. IEEE, 2016.

[9] Sanjoy Baruah and Zhishan Guo. Mixed-criticality scheduling upon varying-speed processors. In *2013 IEEE 34th Real-Time Systems Symposium*, pages 68–77. IEEE, 2013.

[10] Sanjoy Baruah and Zhishan Guo. Scheduling mixed-criticality implicit-deadline sporadic task systems upon a varying-speed processor. In *2014 IEEE Real-Time Systems Symposium*, pages 31–40. IEEE, 2014.

[11] Sanjoy Baruah and Steve Vestal. Schedulability analysis of sporadic tasks with multiple criticality specifications. In *2008 Euromicro Conference on Real-Time Systems*, pages 147–155. IEEE, 2008.

[12] Sanjoy K Baruah, Alan Burns, and Robert I Davis. Response-time analysis for mixed criticality systems. In *2011 IEEE 32nd Real-Time Systems Symposium*, pages 34–43. IEEE, 2011.

[13] Ashikahmed Bhuiyan, Sai Sruti, Zhishan Guo, and Kecheng Yang. Precise scheduling of mixed-criticality tasks by varying processor speed. In *Proceedings of the 27th International Conference on Real-Time Networks and Systems*, pages 123–132, 2019.

[14] Enrico Bini and Giorgio C Buttazzo. Measuring the performance of schedulability tests. *Real-Time Systems*, 30(1-2):129–154, 2005.

[15] Aaron Block, James H Anderson, and Gary Bishop. Fine-grained task reweighting on multiprocessors. *Journal of Embedded Computing*, 4(2):71–86, 2010.

[16] Aaron Block, James H Anderson, and UmaMaheswari C Devi. Task reweighting under global scheduling on multiprocessors. *Real-Time Systems*, 39(1-3):123–167, 2008.

[17] Aaron Block, Björn Brandenburg, James H Anderson, and Stephen Quint. An adaptive framework for multiprocessor real-time system. In *2008 Euromicro Conference on Real-Time Systems*, pages 23–33. IEEE, 2008.

[18] Abdeldjalil Boudjadar, Alexandre David, Jin Hyun Kim, Kim G Larsen, Marius Mikučionis, Ulrik Nyman, and Arne Skou. Hierarchical scheduling framework based on compositional analysis using uppaal. In *International Workshop on Formal Aspects of Component Software*, pages 61–78. Springer, 2013.

[19] Alan Burns and Sanjoy Baruah. Towards a more practical model for mixed criticality systems. In *Workshop on Mixed-Criticality Systems (colocated with RTSS)*, 2013.

[20] Laura Carnevali, Alessandro Pinzuti, and Enrico Vicario. Compositional verification for hierarchical scheduling of real-time systems. *IEEE Transactions on Software Engineering*, 39(5):638–657, 2012.

[21] Sanjian Chen, Linh TX Phan, Jaewoo Lee, Insup Lee, and Oleg Sokolsky. Removing abstraction overhead in the composition of hierarchical real-time systems. In *2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 81–90. IEEE, 2011.

[22] Robert I Davis, Attila Zabos, and Alan Burns. Efficient exact schedulability tests for fixed priority real-time systems. *IEEE Transactions on Computers*, 57(9):1261–1276, 2008.

[23] Z. Deng and J. Liu. Scheduling real-time applications in an open environment. In *Proceedings of the 18th IEEE Real-Time Systems Symposium*, pages 308–319, 1997.

[24] Zheng Dong and Cong Liu. Closing the loop for the selective conversion approach: A utilization-based test for hard real-time suspending task systems. In *2016 IEEE Real-Time Systems Symposium (RTSS)*, pages 339–350. IEEE, 2016.

[25] Zheng Dong and Cong Liu. Analysis techniques for supporting hard real-time sporadic gang task systems. *Real-Time Systems*, 55(3):641–666, 2019.

[26] Zheng Dong and Cong Liu. An efficient utilization-based test for scheduling hard real-time sporadic dag task systems on multiprocessors. In *2019 IEEE Real-Time Systems Symposium (RTSS)*, pages 181–193. IEEE, 2019.

[27] Zheng Dong, Cong Liu, Soroush Bateni, Kuan-Hsun Chen, Jian-Jia Chen, Georg von der Brüggen, and Junjie Shi. Shared-resource-centric limited preemptive scheduling: A comprehensive study of suspension-based partitioning approaches. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 164–176. IEEE, 2018.

[28] Zheng Dong, Cong Liu, Alan Gatherer, Lee McFearin, Peter Yan, and

James H Anderson. Optimal dataflow scheduling on a heterogeneous multiprocessor with reduced response time bounds. In *29th Euromicro Conference on Real-Time Systems (ECRTS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[29] Arvind Easwaran. Demand-based scheduling of mixed-criticality sporadic tasks on one processor. In *2013 IEEE 34th Real-Time Systems Symposium*, pages 78–87. IEEE, 2013.

[30] Arvind Easwaran, Madhukar Anand, and Insup Lee. Compositional analysis framework using edp resource models. In *28th IEEE International Real-Time Systems Symposium (RTSS 2007)*, pages 129–138. IEEE, 2007.

[31] Arvind Easwaran and Insup Lee. Compositional schedulability analysis for cyber-physical systems. *ACM Sigbed Review*, 5(1):6, 2008.

[32] Arvind Easwaran, Insup Lee, Oleg Sokolsky, and Steve Vestal. A compositional scheduling framework for digital avionics systems. In *2009 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTAS)*, pages 371–380. IEEE, 2009.

[33] Pontus Ekberg and Wang Yi. Bounding and shaping the demand of mixed-criticality sporadic tasks. In *2012 24th Euromicro Conference on Real-Time Systems*, pages 135–144. IEEE, 2012.

[34] Pontus Ekberg and Wang Yi. Bounding and shaping the demand of generalized mixed-criticality sporadic task systems. *Real-time systems*, 50(1):48–86, 2014.

[35] Nathan Fisher and Farhana Dewan. Approximate bandwidth allocation for compositional real-time systems. In *2009 21st Euromicro Conference on Real-Time Systems*, pages 87–96. IEEE, 2009.

[36] Oliver Gettings, Sophie Quinton, and Robert I Davis. Mixed criticality systems with weakly-hard constraints. In *Proceedings of the 23rd International Conference on Real Time and Networks Systems*, pages 237–246, 2015.

[37] Chris Gill, James Orr, and Steven Harris. Supporting graceful degradation through elasticity in mixedcriticality federated scheduling. In *Proc. 6th Workshop on Mixed Criticality Systems (WMC), RTSS*, pages 19–24, 2018.

[38] Xiaozhe Gu and Arvind Easwaran. Dynamic budget management with service guarantees for mixed-criticality systems. In *2016 IEEE Real-Time Systems Symposium (RTSS)*, pages 47–56. IEEE, 2016.

[39] Xiaozhe Gu and Arvind Easwaran. Dynamic budget management and budget reclamation for mixed-criticality systems. *Real-Time Systems*, pages 1–46, 2019.

[40] Xiaozhe Gu, Arvind Easwaran, Kieu-My Phan, and Insik Shin. Resource efficient isolation mechanisms in mixed-criticality scheduling. In *2015 27th Euromicro Conference on Real-Time Systems*, pages 13–24. IEEE, 2015.

[41] Z. Guo, L. Santinelli, and K. Yang. EDF schedulability analysis on mixed-criticality systems with permitted failure probability. In *Proceedings of the 21st IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 187–196, 2015.

[42] Zhishan Guo, Kecheng Yang, Sudharsan Vaidhun, Samsil Arefin, Sajal K Das, and Haoyi Xiong. Uniprocessor mixed-criticality scheduling with graceful degradation by completion rate. In *2018 IEEE Real-Time Systems Symposium (RTSS)*, pages 373–383. IEEE, 2018.

[43] Lin Huang, I Hou, Sachin S Sapatnekar, Jiang Hu, et al. Graceful degradation of low-criticality tasks in multiprocessor dual-criticality systems. In *Proceedings of the 26th International Conference on Real-Time Networks and Systems*, pages 159–169. ACM, 2018.

[44] Pengcheng Huang, Georgia Giannopoulou, Nikolay Stoimenov, and Lothar Thiele. Service adaptions for mixed-criticality systems. In *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 125–130. IEEE, 2014.

[45] Bernhard Huber, Christian El Salloum, and Roman Obermaisser. A resource management framework for mixed-criticality embedded systems. In *2008 34th Annual Conference of IEEE Industrial Electronics*, pages 2425–2431. IEEE, 2008.

[46] Mathieu Jan, Lilia Zaourar, and Maurice Pitel. Maximizing the execution rate of low criticality tasks in mixed criticality system. *Proc. WMC, RTSS*, pages 43–48, 2013.

[47] Jaewoo Lee, Kieu-My Phan, Xiaozhe Gu, Jiyeon Lee, Arvind Easwaran, Insik Shin, and Insup Lee. Mc-fluid: Fluid model-based mixed-criticality scheduling on multiprocessors. In *2014 IEEE Real-Time Systems Symposium*, pages 41–52. IEEE, 2014.

[48] Hennadiy Leontyev and James H Anderson. A hierarchical multiprocessor bandwidth reservation scheme with timing guarantees. *Real-Time Systems*, 43(1):60–92, 2009.

[49] Joseph Y-T Leung and Jennifer Whitehead. On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance evaluation*, 2(4):237–250, 1982.

[50] Chung Laung Liu and James W Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)*, 20(1):46–61, 1973.

[51] Di Liu, Jelena Spasic, Nan Guan, Gang Chen, Songran Liu, Todor Stefanov, and Wang Yi. EDF-VD scheduling of mixed-criticality systems with degraded quality guarantees. In *2016 IEEE Real-Time Systems Symposium (RTSS)*, pages 35–46. IEEE, 2016.

[52] Chenyang Lu, John A Stankovic, Tarek F Abdelzaher, Gang Tao, Sang Hyuk Son, and Michael Marley. Performance specifications and metrics for adaptive real-time systems. In *Proceedings 21st IEEE Real-Time Systems Symposium*, pages 13–23. IEEE, 2000.

[53] Chenyang Lu, John A Stankovic, Sang H Son, and Gang Tao. Feedback control real-time scheduling: Framework, modeling, and algorithms. *Real-Time Systems*, 23(1-2):85–126, 2002.

[54] Shanmuga Priya Marimuthu and Samarjit Chakraborty Chakraborty. A framework for compositional and hierarchical real-time scheduling. In *12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06)*, pages 91–96. IEEE, 2006.

[55] Andrew Nelson, Kees Goossens, and Benny Akesson. Dataflow formalisation of real-time streaming applications on a composable and predictable multi-processor soc. *Journal of Systems Architecture*, 61(9):435–448, 2015.

[56] Linh TX Phan, Meng Xu, Jaewoo Lee, Insup Lee, and Oleg Sokolsky. Overhead-aware compositional analysis of real-time systems. In *2013 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 237–246. IEEE, 2013.

[57] Saravanan Ramanathan, Arvind Easwaran, and Hyeonjoong Cho. Multi-rate fluid scheduling of mixed-criticality systems on multiprocessors. *Real-Time Systems*, 54(2):247–277, 2018.

[58] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *Proceedings of the 24th IEEE Real-Time Systems Symposium*, pages 1–12, 2003.

[59] Insik Shin and Insup Lee. Compositional real-time scheduling framework with periodic model. *ACM Transactions on Embedded Computing Systems (TECS)*, 7(3):30, 2008.

[60] Hang Su, Peng Deng, Dakai Zhu, and Qi Zhu. Fixed-priority dual-rate mixed-criticality systems: Schedulability analysis and performance optimization. In *2016 IEEE 22nd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 59–68. IEEE, 2016.

[61] Hang Su, Nan Guan, and Dakai Zhu. Service guarantee exploration for mixed-criticality systems. In *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 1–10. IEEE, 2014.

[62] Hang Su and Dakai Zhu. An elastic mixed-criticality task model and its scheduling algorithm. In *2013 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 147–152. IEEE, 2013.

[63] S. Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *Proceedings of the 28th IEEE Real-Time Systems Symposium*, pages 239–243, 2007.

[64] Steve Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *28th IEEE International Real-Time Systems Symposium (RTSS 2007)*.

[65] Hao Xu and Alan Burns. Semi-partitioned model for dual-core mixed criticality system. In *Proceedings of the 23rd International Conference on Real Time and Networks Systems*, pages 257–266. ACM, 2015.

[66] Meng Xu, Linh Thi Xuan Phan, Oleg Sokolsky, Sisu Xi, Chenyang Lu, Christopher Gill, and Insup Lee. Cache-aware compositional analysis of real-time multicore virtualization platforms. *Real-Time Systems*, 51(6):675–723, 2015.

[67] Jungwoo Yang, Hyungseok Kim, Sangwon Park, Changki Hong, and Insik Shin. Implementation of compositional scheduling framework on virtualization. *ACM SIGBED Review*, 8(1):30–37, 2011.

[68] Kecheng Yang and James H Anderson. On the dominance of minimum-parallelism multiprocessor supply. In *2016 IEEE Real-Time Systems Symposium (RTSS)*, pages 215–226. IEEE, 2016.

[69] Kecheng Yang, Ashikahmed Bhuiyan, and Zhishan Guo. F2VD: Fluid rates to virtual deadlines for precise mixed-criticality scheduling on

a varying-speed processor. In *Proceedings of the 39th IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.

[70] Kecheng Yang and Zheng Dong. Mixed-criticality scheduling with varying processor supply in compositional real-time systems. In *Proc. 7th Workshop on Mixed Criticality Systems (WMC), RTSS*, pages 26–31, 2019.

[71] Kecheng Yang and Zhishan Guo. EDF-based mixed-criticality scheduling with graceful degradation by bounded lateness. In *2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–6. IEEE, 2019.

[72] Tianyu Zhang, Nan Guan, Qingxu Deng, and Wang Yi. On the analysis of edf-vd scheduled mixed-criticality real-time systems. In *Proceedings of the 9th IEEE International Symposium on Industrial Embedded Systems (SIES 2014)*, pages 179–188. IEEE, 2014.