Elastic Scheduling for Graceful Degradation of Mixed-Criticality Systems

Zhuoran Sun
Washington University in St. Louis
St. Louis, US
zhuoran.sun@wustl.edu

Marion Sudvarg
Washington University in St. Louis
St. Louis, US
msudvarg@wustl.edu

Christopher Gill
Washington University in St.louis
St. Louis, US
cdgill@wustl.edu

Abstract

Many mixed-criticality system models drop all jobs of low-criticality tasks when a criticality mode switch occurs, ensuring that high-criticality tasks still can meet their deadlines in the new mode. However, this means that even important low-criticality tasks are discarded, which may not be acceptable in some systems in practice. This paper addresses that distinction between criticality and importance through a new Inelastic Graceful Earliest Deadline First with Virtual Deadlines (IG-EDF-VD) scheme that upon a criticality mode switch only discards the least important low-criticality tasks necessary to ensure feasibility. Moreover, we consider elastic scheduling within our mixed-criticality model (EG-EDF-VD), using compression of workload-elastic tasks' utilizations (and, as a result, execution time budgets) to reduce further the number of low-criticality tasks that are dropped.

CCS Concepts

• **Computer systems organization** → **Real-time systems**.

Keywords

Real-Time Systems, Mixed-Criticality Systems, Elastic Scheduling, Graceful Degradation

ACM Reference Format:
Zhuoran Sun, Marion Sudvarg, and Christopher Gill. 2024. Elastic Scheduling for Graceful Degradation of Mixed-Criticality Systems. In *The 32nd International Conference on Real-Time Networks and Systems (RTNS 2024), November 06–08, 2024, Porto, Portugal.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3696355.3699701

1 Introduction

A mixed-criticality real-time system (e.g., a drone using a Fast Integrated Mobility Spectrometer [26, 29] to map the extent of an aerosol plume, or a high-altitude balloon flying the ADAPT telescope [4, 14, 22, 27] to detect and localize gamma-ray bursts) must distinguish high-criticality tasks whose schedulability must be assured for system safety, from low-criticality tasks that may be safely dropped if unavoidable due to system overload. For example, drone navigation tasks to avoid flying into obstacles are high-criticality; in contrast, jobs of tasks that determine the aerosol plume's extent may be dropped safely, and thus are low-criticality, even though they are essential to the drone's mission. Similarly, control tasks to maintain balloon stability and telescope temperature are high-criticality; in contrast, jobs of tasks that collect and analyze gamma-ray data are low-criticality.

Furthermore, within each criticality level, some tasks may be more important to the system than others. For example, tasks involved with mapping the aerosol plume may be more important than tasks that collect and report mission statistics for post-hoc analysis: dropping the latter would have a lesser effect on the mission's outcome than dropping the former. Similarly, tasks for reading out sensors that measure gamma-ray interactions in the telescope may be more important than high-level analysis tasks; dropping all of the former would prevent the latter from executing anyway.

What is needed then is a means to support graceful degradation of system outcomes while still maintaining system safety by reducing the number of low-criticality jobs that are dropped. In this paper, we propose a new Inelastic Graceful Earliest Deadline First with Virtual Deadlines (**IG-EDF-VD**) algorithm, which considers the importance of tasks as well as their criticality. When a mode switch is triggered by an overrun of a high-criticality task, our algorithm only drops jobs of the lowest importance low-criticality tasks as needed to maintain schedulability. We note that the IG-EDF-VD algorithm is the first to integrate importance with dynamic-priority mixed-criticality scheduling (specifically EDF-VD).

Additionally, in practice, some tasks also may be *elastic*. Such tasks may, with varying degrees of flexibility, reduce their utilizations in a continuous range by reducing their workloads (e.g., the amount of input data processed, the number of iterations of a refinement step). Elastic real-time scheduling models allow for principled compression of elastic task utilizations until the cumulative utilization falls below the utilization bound while guaranteeing that each task's utilization remains above some specified minimum value [5]. This, in turn, offers further opportunities for graceful degradation as an alternative to dropping jobs by reducing task utilizations across a mode switch to a higher criticality level.

To this end, we propose an Elastic Graceful Earliest Deadline First with Virtual Deadlines (**EG-EDF-VD**) algorithm that attempts to compress the utilizations of workload-elastic tasks to minimize the number of tasks that must be suspended across a criticality mode switch. Intuitively, the algorithm determines the minimum number of tasks (in order of importance) to suspend when all task utilizations are compressed to their minimum values. Then, given the set of tasks that *must* be dropped, the algorithm applies elastic scheduling's principled compression to achieve the *least* degradation (i.e., reduction in task utilizations) to remain schedulable. As

RTNS 2024, November 06–08, 2024, Porto, Portugal

ACM ISBN 979-8-4007-1724-6/24/11
https://doi.org/10.1145/3696355.3699701

we will show, in many cases, this allows fewer tasks to be suspended compared to IG-EDF-VD.

The remainder of the paper is organized as follows. Section 2 discusses related work. Section 3 presents the considered system models. Section 4 introduces the IG-EDF-VD algorithm, and Section 5 introduces the EG-EDF-VD algorithm. Section 6 evaluates these algorithms using randomly-generated synthetic task sets. Section 7 concludes the paper and discusses directions for future work.

2 Related Work

This paper considers systems of implicit-deadline sporadic real-time tasks scheduled on a single processor core. In this section, we outline relevant prior work in this area, including for mixed-criticality and elastic real-time task systems. In Liu and Layland's classic three-parameter task model [17], a system is schedulable on a preemptive uniprocessor using the Earliest Deadline First (EDF) algorithm if the system's total utilization demand does not exceed 1.

2.1 Mixed-Criticality Systems

In safety-critical systems, statutory certification authorities may require that task worst-case execution times (WCETs) (and therefore, utilizations) be certified to a high level of assurance. The resulting utilization demand may be pessimistic, and not indicative of system load under common-case behavior. This may be too restrictive, especially in mixed-criticality systems that integrate both safety-critical and non-critical tasks. To accommodate both common- and worst-case behaviors, Vestal proposed to classify tasks according to their criticality levels [28]. Criticality may be defined according to a variety of safety standards, such as ISO 26262 [16] for the automotive domain, DO 178C [7] for aeronautics, or IEC 61508 [15] for generic purposes. In this paper, we restrict our analysis to two levels: low and high criticality.

Under Vestal's model, each high-criticality task is characterized with two WCETs: one representing common-case behavior at the certification level demanded by low-criticality tasks, and one representing the worst-case behavior under the assurances of the high-criticality certification level. If a job of a high-criticality task continues to execute beyond its low-criticality WCET, the system switches into high-criticality mode, and jobs of low-criticality tasks are dropped. This allows high-criticality tasks to meet their deadlines even under their more pessimistic high-criticality WCETs while allowing all low-criticality tasks to meet their deadlines if the system remains in low-criticality mode.

Earliest Deadline First with Virtual Deadlines (EDF-VD) [3] is a dynamic-priority scheduling algorithm for mixed-criticality systems. EDF-VD has been proven to be speedup-optimal [1] and outperforms other mixed-criticality scheduling algorithms in terms of its acceptance ratio [8, 9]. Under EDF-VD, $U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}}$ denotes the total utilization of low-criticality tasks; $U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}}$ denotes the total utilization of high-criticality tasks according to their low-criticality WCETs, and $U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}}$ denotes their total utilization according to their high-criticality (most pessimistic) WCETs. Namely, for each of $\chi \in \{\mathrm{LO}, \mathrm{HI}\}$ and $\Gamma_k \in \{\Gamma_{\mathrm{LO}}, \Gamma_{\mathrm{HI}}\}$, we denote:

$$U^{\chi}_{\Gamma_k} = \sum_{\tau_i \in \Gamma_k} U^{\chi}_i$$

Tasks are scheduled according to EDF, but high-criticality tasks τ_i are each assigned a *virtual deadline* $\hat{T}_i \leftarrow xT_i$ where T_i is the task's period and x is computed as

$$x \leftarrow \frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}}}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}}} \tag{1}$$

If a high-criticality task overruns its low-criticality WCET, jobs of all low-criticality tasks are dropped, and high-criticality task deadlines are set to their original T_i values. In [1], Baruah et al. prove that a mixed-criticality system is schedulable under EDF-VD if

$$xU^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} + U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}} \leq 1 \tag{2}$$

In this paper, we refer to the following as the EDF-VD *schedulability function* for a mixed-criticality system Γ:

$$\mathcal{B}(\Gamma) = \frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}}}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}}} \cdot U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} + U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}} \tag{3}$$

From Equations 1 and 2, a mixed-criticality system is schedulable under EDF-VD if $\mathcal{B}(\Gamma) \leq 1$.

2.2 Importance

A key distinction can be made between a task's *criticality*, which is used to validate mixed-criticality systems according to the safety standards noted in Section 2.1, and its *importance* [11, 12]. Fleming and Burns observed that dropping jobs of all low-criticality tasks is unacceptable for many practical applications and proposed suspending low-criticality tasks based on their importance, which is specified by a system developer to capture each task's relative contribution to the system's performance or other objectives [12].

For example, consider a hobbyist drone in "follow-me" mode, with an objective to follow and film a moving target. In this case, flight control tasks are high-criticality, since missing a deadline may result in a crash. In contrast, the video-related tasks are low-criticality. Compared to the target tracking task, the video capturing task is more important, since missing a deadline of the latter results in dropped frames, whereas missing a deadline of the former still retains video frames, but the target might be off-center or missing.

As a second example, consider a high-energy nuclear physics detector or astrophysics telescope designed to read and process data from particle interactions in the instrument. Such instruments are deployed in diverse environments, where critical control tasks maintain, e.g., stability of a flown telescope, or temperatures and voltages in sensitive electronics; missing a deadline might result in expensive damage. Low-criticality data collection tasks are still important to the mission; for example, real-time gamma-ray trajectory reconstruction and localization aboard the ADAPT telescope will enable rapid follow-up observations of gamma-ray bursts by other telescopes [25]. But compared to the localization task, the reconstruction task is more important, as reconstructed data is necessary for subsequent localization.

Our work in this paper adopts Fleming and Burns' convention of representing importance as a strict total order over the set of

tasks $\{\tau_i\}$ using an *ordinal*, but not *interval*, measure: it does not necessarily support addition or other arithmetic operations on its values, nor does it capture any notion of multiple less important tasks being more important in combination than a single task of higher importance. We note that an *interval* importance may be applicable to many real-world applications, but it is left to future work.

Fleming and Burns used this notion of importance to reduce the number of tasks suspended across a mode switch in a fixed-priority mixed-criticality model [12]. Our paper is the first to integrate it with a dynamic-priority scheduling model—specifically, we consider EDF-VD [3].

2.3 Adaptability and Elastic Scheduling

Several other alternatives to job and task dropping have been proposed. For instance, the imprecise mixed-criticality model allows low-criticality jobs to run at a reduced utilization instead of being dropped [18].

Another example is Su and Zhu's elastic task model for uniprocessor mixed-criticality systems, which characterizes each low-criticality task with a maximum period that reflects its minimum service requirement [21]. If the system switches to a high-criticality mode, periods are expanded to these values (with opportunities for early releases given sufficient slack). Compared to the imprecise mixed-criticality model, this elastic model is more adaptive, with multiple early release points for slack reclamation. We argue that while this is an *adaptive* model, it is not quite a truly *elastic* model because low-criticality tasks still have their utilizations fully degraded to their minimum acceptable values.

The elastic model of Buttazzo et al. for implicit-deadline sporadic tasks on a uniprocessor [5, 6] characterizes each task τ_i with a continuous range $[U_i^{\min}, U_i^{\max}]$ from which its utilization U_i can be assigned. Under normal operation, tasks execute at their maximum utilizations $U_i^{\max}$. Each task may also be assigned an elastic constant E_i representing "the flexibility of the task to vary its utilization" [5]. We note that a task with $E_i = 0$ (equivalently, $U_i^{\min} = U_i^{\max}$) cannot be compressed and is referred to as an *inelastic* task. Conversely, a task is referred to as an *elastic* task if $E_i > 0$.

If the system becomes overloaded, task utilizations may be reduced by increasing their periods (period-elasticity) or decreasing their execution times (workload-elasticity [20]). Under Buttazzo's model, task utilizations are compressed from their maximum values $U_i^{\max}$ in proportion to their elastic constants E_i until the system becomes schedulable. Specifically, this means that for all elastic tasks τ_i and τ_j with $U_i > U_i^{\min}$ and $U_j > U_j^{\min}$, the following relationship must be satisfied:

$$\left(\frac{U_i^{\max} - U_i}{E_i}\right) = \left(\frac{U_j^{\max} - U_j}{E_j}\right) \tag{4}$$

In [19], Orr and Baruah introduce a term Φ to capture this equilibrium value; each task's utilization value is thus:

$$U_i(\Phi) \stackrel{\text{def}}{=} \max\left(U_i^{\max} - \Phi \cdot E_i, U_i^{\min}\right) \tag{5}$$

We refer to Φ as the *system compression level*.

If an elastic task's utilization reaches its minimum, it is not reduced further, although other elastic tasks may be compressed more if needed to reach schedulability. In [23, 24], Sudvarg et al. assign a parameter ϕ_i to each elastic task, representing the system compression level needed for an elastic task to reach its minimum utilization:

$$\phi_i \stackrel{\text{def}}{=} \frac{U_i^{\max} - U_i^{\min}}{E_i} \tag{6}$$

We refer to ϕ_i as the *maximum compression level* of task τ_i. Rather than forcing each low-criticality task to execute at its minimum utilization if the system criticality level increases, as in Su and Zhu's model [21] where tasks execute at their maximum periods, we suggest instead that utilizations should be compressed only to the extent necessary to accommodate low-criticality task execution in high-criticality mode. In this paper, we propose an elastic mixed-criticality model for workload-elastic tasks and present an algorithm to minimize the number of low-criticality tasks dropped while compressing task workloads as little as possible to still guarantee schedulability across a mode switch.

3 System Model

In this paper, we consider two system models. Section 4 uses Vestal's classic preemptive mixed-criticality model from [28] with an additional ordinal importance parameter assigned to each low-criticality task per Fleming and Burns [12]. Section 5 extends this model to Buttazzo's elastic scheduling [5, 6] by additionally parameterizing each task with a continuous range of allowed utilizations and an elasticity.

3.1 Mixed-Criticality with Importance

In traditional mixed-criticality systems, if a job of a high-criticality task executes for time C_i^{LO} without completing, a system criticality mode switch is triggered. In this high-criticality mode, jobs of all low-criticality tasks are discarded.

In Section 4, we introduce a scheduling algorithm that allows, to the extent possible, low-criticality tasks to continue executing across a mode switch. It assumes a finite set Γ of mixed-criticality, implicit-deadline, sporadic tasks executing on a single-core processor. Each task τ_i is characterized by the tuple $\tau_i = (\chi_i, I_i, T_i, U_i^{\text{LO}}, U_i^{\text{HI}})$, where:

- $\chi_i \in \{\text{LO}, \text{HI}\}$ denotes the task's criticality.
- I_i denotes the unique importance of a low-criticality task, i.e, $\forall \tau_i, \tau_j$ where $i \neq j$, $I_i \neq I_j$. Note that, as in [12], importance is *ordinal* but not *interval*; one should not perform arithmetic on the value. After a mode switch, our algorithm suspends low-criticality tasks in order of increasing importance until the remaining tasks are schedulable.
- T_i denotes the task's period or minimum inter-arrival time.
- U_i^{LO} denotes the task's utilization under low-criticality mode. From this, the worst-case execution time in low-criticality mode $C_i^{\text{LO}} = U_i^{\text{LO}} \cdot T_i$ can be derived.
- U_i^{HI} denotes the task's utilization under high-criticality mode. For high-criticality tasks, $U_i^{\text{HI}} \geq U_i^{\text{LO}}$. For low-criticality tasks not suspended across a mode switch, $U_i^{\text{HI}} = U_i^{\text{LO}}$. Similarly, $C_i^{\text{HI}} = U_i^{\text{HI}} \cdot T_i$ can be derived.

We denote the set of high-criticality tasks as Γ_{HI} and the set of low-criticality tasks as Γ_{LO}.

3.2 Elastic Mixed-Criticality with Importance

An intuitive extension of the above model to Buttazzo's elastic scheduling [5, 6] for workload-elastic tasks is to characterize each task as

$$\tau_i = (\chi_i, I_i, T_i, U_i^{\text{LO,min}}, U_i^{\text{LO,max}}, U_i^{\text{HI,min}}, U_i^{\text{HI,max}}, E_i^{\text{LO}}, E_i^{\text{HI}})$$

where

- $\chi_i \in \{\text{LO}, \text{HI}\}$ denotes the task's criticality.
- I_i denotes ordinal importance, as in Section 3.1.
- T_i again denotes the task's period.
- $\left[U_i^{\text{LO, min}}, U_i^{\text{LO, max}}\right]$ denotes the continuous range of allowed values from which the task's assigned utilization U_i^{LO} may be selected under low-criticality mode.
- $\left[U_i^{\text{HI, min}}, U_i^{\text{HI, max}}\right]$ denotes the continuous range of allowed values from which the task's assigned utilization U_i^{HI} may be selected under high-criticality mode.
- E_i^{LO} and E_i^{HI} denote the elasticity of the task under low- and high-criticality modes.

A task τ_i has $E_i^{\text{LO}} = E_i^{\text{HI}} = 0$ if it is *inelastic*, while $E_i^{\text{LO}} > 0$ and $E_i^{\text{HI}} > 0$ if the task is *elastic*.

For workload-elastic tasks, which we consider in this paper, the period T_i remains constant. From the other parameters, the task's WCET values can be derived as $C_i^{\chi,\text{min}} = U_i^{\chi,\text{min}} \cdot T_i$ and $C_i^{\chi,\text{max}} = U_i^{\chi,\text{max}} \cdot T_i$.

We extend Orr and Baruah's definition of the system compression level (Equation 5) to this system model:

$$U_i^{\chi}(\Phi) \stackrel{\text{def}}{=} \max\left(U_i^{\chi,\text{max}} - \Phi \cdot E_i^{\chi}, U_i^{\chi,\text{min}}\right) \tag{7}$$

Similarly, we can extend Sudvarg's definition of the maximum compression level of a task (Equation 6):

$$\phi_i^{\chi} \stackrel{\text{def}}{=} \frac{U_i^{\chi,\text{max}} - U_i^{\chi,\text{min}}}{E_i^{\chi}} \tag{8}$$

In this paper, we restrict ϕ_i to remain constant across criticality levels. This guarantees two properties that are necessary to the development of the EG-EDF-VD algorithm. First, for a given system compression level Φ, it guarantees that a task that is fully compressed to its minimum utilization $U_i^{\chi,\text{min}}$ in one criticality level remains so in the other level. Second, it guarantees that elastic tasks do not exceed their budgets upon a mode switch. These properties are further motivated and proven in Section 5.

By rearranging the definition of ϕ_i, we can express the elasticity at each criticality level as

$$\begin{cases} E_i^{\text{LO}} = \dfrac{U_i^{\text{LO, max}} - U_i^{\text{LO, min}}}{\phi_i} \\ E_i^{\text{HI}} = \dfrac{U_i^{\text{HI, max}} - U_i^{\text{HI, min}}}{\phi_i} \end{cases} \tag{9}$$

By doing so, we may re-parameterize each task τ_i with one fewer parameter as

$$\tau_i = (\chi_i, I_i, T_i, U_i^{\text{LO,min}}, U_i^{\text{LO,max}}, U_i^{\text{HI,min}}, U_i^{\text{HI,max}}, \phi_i)$$

We motivate this model by illustrating how elasticity may be orthogonal to criticality for workload-elastic tasks.

Example 1. *Consider a high-criticality iterative refinement task τ_i that runs every 200ms, requiring 4 iterations at minimum and ideally 10. The WCET of one iteration is 5ms under low-criticality and 10ms under high-criticality for certification.*

We can characterize task τ_i with $T_i = 200ms$, $U_i^{\text{LO,min}} = 0.1$, $U_i^{\text{HI,min}} = 0.2$, $U_i^{\text{LO,max}} = 0.25$, $U_i^{\text{HI,max}} = 0.5$, and $\phi_i = 6$. Then the allowed range of the task's WCET is $[20ms, 50ms]$ under low-criticality mode and $[40ms, 100ms]$ under high-criticality mode. Suppose the EG-EDF-VD algorithm determines that a system compression level $\Phi = 2$ guarantees schedulability. Then utilizations are assigned as $U_i^{\text{LO}}(\Phi) = 0.2$ and $U_i^{\text{HI}}(\Phi) = 0.4$, with corresponding execution times $C_i^{\text{LO}} = 40ms$ and $C_i^{\text{HI}} = 80ms$ guaranteeing that at least 8 iterations can complete in either criticality level.

We note that, although the number of iterations completed represents a *discrete* value, it is still worthwhile to assign a *continuous* range of allowed utilizations. For example, for $\Phi = 2.5$, the task in the above example would have execution time budgets $C_i^{\text{LO}} = 37.5ms$ and $C_i^{\text{HI}} = 75ms$, guaranteeing completion of 7 iterations. However, since workloads are characterized according to the WCET, faster execution might allow 8 iterations to complete; this becomes more likely as utilization is continuously increased.

4 Inelastic Graceful EDF-VD

This section introduces the Inelastic Graceful EDF-VD (**IG-EDF-VD**) algorithm, which extends the EDF-VD algorithm by incorporating *ordinal importance* according to the model defined in Section 3.1. The idea is to find the maximal set, in order of importance, of low-criticality tasks that can continue to execute in high-criticality mode. In Section 4.1, we first motivate the algorithm with an example. Next, we describe the IG-EDF-VD algorithm in Section 4.2. Finally, we analyze the schedulability of the IG-EDF-VD scheduling scheme from an adaption of the EDF-VD schedulability test in Section 4.3.

4.1 Motivation

Example 2. *Consider the implicit-deadline task system Γ whose parameters are shown in Table 1.*

	χ_i	I_i	T_i	U_i^{LO}	U_i^{HI}
τ_1	HI	-	91.735	0.255	0.518
τ_2	HI	-	4.286	0.095	0.132
τ_3	LO	1	1.710	0.245	0.245
τ_4	LO	2	92.718	0.111	0.111
τ_5	LO	3	2.300	0.094	0.094

Table 1: Sample inelastic taskset

Applying the EDF-VD schedulability function from Equation 3, we get

$$\mathcal{B}(\Gamma) = \frac{0.35}{1 - 0.45} \cdot 0.45 + 0.65 \approx 0.936$$

Since $\mathcal{B}(\Gamma) \leq 1$, the task set Γ is schedulable under EDF-VD.

Algorithm 1 IG-EDF-VD schedulability test

Require: An implicit-deadline sporadic task system Γ where Γ_{LO} is sorted in ascending order of importance.
Ensure: An IG-EDF-VD schedule is feasible for the returned set of undroppable tasks.
1: **if** $U^{\text{LO}}_{\Gamma_{\text{LO}}} + U^{\text{HI}}_{\Gamma_{\text{HI}}} \leq 1$ **then return** Γ_{LO} ▷ Attempt EDF
2: $\Gamma_{\text{UD}} \leftarrow \Gamma_{\text{LO}}$
3: $\Gamma_{\text{DR}} \leftarrow \varnothing$
4: **while** $\Gamma_{\text{UD}} \neq \varnothing$ **do**
5: Move the least important task τ_i from Γ_{UD} to Γ_{DR}
6: **if** $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}) \leq 1$ **then return** Γ_{UD} ▷ Per Eqn. 10
7: **end while**
8: **return** INFEASIBLE

Now consider a task set Γ' that is identical to Γ except that τ_5 is a high-criticality task instead of a low-criticality task. Applying the same schedulability function, we get

$$\mathcal{B}(\Gamma') = \frac{0.444}{1 - 0.356} \cdot 0.356 + 0.744 \approx 0.989$$

Since $\mathcal{B}(\Gamma') \leq 1$, the task set Γ' is also schedulable.

To minimize the number of low-criticality tasks dropped, we propose IG-EDF-VD, as outlined in Algorithm 1. It finds the maximal subset (in order of importance) of *undroppable* low-criticality tasks Γ_{UD} that can be treated as high-criticality tasks while an EDF-VD schedule remains feasible.

4.2 IG-EDF-VD Algorithm

Like EDF-VD, the IG-EDF-VD algorithm first performs a schedulability test and computes a modified period $\hat{T}_i \leq T_i$. However, IG-EDF-VD also partitions the low-criticality tasks into subsets of undroppable and droppable tasks — Γ_{UD} and Γ_{DR} — based on importance. Any task in Γ_{UD} is always more important than any task in Γ_{DR}, i.e, $\forall \tau_i \in \Gamma_{\text{UD}}$ and $\forall \tau_j \in \Gamma_{\text{DR}}$, $I_i > I_j$. Tasks in Γ_{UD} are treated as if they are high-criticality. The algorithm finds the *maximal* assignment of tasks to set Γ_{UD} (in order of importance) for which schedulability remains guaranteed.

During runtime, the IG-EDF-VD algorithm assigns virtual deadlines $t_a + \hat{T}_i$ if $\tau_i \in \Gamma_{\text{HI}}$ or $\tau_i \in \Gamma_{\text{UD}}$ and $t_a + T_i$ if $\tau_i \in \Gamma_{\text{DR}}$ to tasks τ_i arriving at time t_a. If a high-criticality job executes beyond its low-criticality WCET without signaling that it has completed execution:

- All droppable tasks' jobs are immediately suspended.
- Subsequent run-time scheduling of all undroppable and high-criticality tasks continues to be done according to EDF, but actual job deadlines $t_a + T_i$ are used.

Note that even though the undroppable tasks are treated as high-criticality tasks, they will never trigger an overrun since their low-criticality WCET is the same as their high-criticality WCET, which they will never exceed.

For each of $\chi \in \{\text{LO}, \text{HI}\}$ and $\Gamma_k \in \{\Gamma_{\text{UD}}, \Gamma_{\text{DR}}\}$, we define a utilization parameter as follows:

$$U^{\chi}_{\Gamma_k} = \sum_{\tau_i \in \Gamma_k} U^{\chi}_i$$

With that, we define the IG-EDF-VD schedulability function for a mixed-criticality system Γ as follows:

$$\mathcal{B}(\Gamma, \Gamma_{\text{UD}}) = \frac{U^{\text{LO}}_{\Gamma_{\text{HI}}} + U^{\text{LO}}_{\Gamma_{\text{UD}}}}{1 - U^{\text{LO}}_{\Gamma_{\text{DR}}}} \cdot U^{\text{LO}}_{\Gamma_{\text{DR}}} + U^{\text{HI}}_{\Gamma_{\text{UD}}} + U^{\text{HI}}_{\Gamma_{\text{HI}}} \tag{10}$$

Note that when $\Gamma_{\text{UD}} = \varnothing$, we have $\Gamma_{\text{DR}} = \Gamma_{\text{LO}}$, and the schedulability function is equivalent to that of EDF-VD. As we will show in Section 4.3, this function bounds the IG-EDF-VD algorithm, i.e., if $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}) \leq 1$, then a schedule for the system Γ is feasible under IG-EDF-VD.

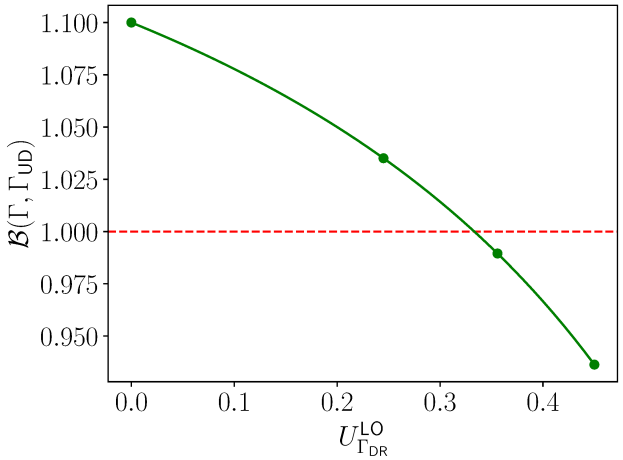

Figure 1: Schedulability function against utilization of droppable tasks of a sample inelastic task set

The schedulability function of the task set in Example 2 is shown in Figure 1. Each point represents one partition of the task set. The x-axis represents the value $U^{\text{LO}}_{\Gamma_{\text{DR}}}$ for the corresponding partition, while the y-axis represents the schedulability function $\mathcal{B}(\Gamma, \Gamma_{\text{UD}})$ of that partition. A horizontal reference line shows the schedulability bound where $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}) = 1$. Any partition that lies on or below the reference line is schedulable under IG-EDF-VD; those above the line are not deemed schedulable.

One may observe in Figure 1 that the schedulability function is monotonically decreasing as $U^{\text{LO}}_{\Gamma_{\text{DR}}}$ increases, which we prove in Lemma 4.4 of Section 4.3. This means that as tasks move from Γ_{UD} to Γ_{DR}, $U^{\text{LO}}_{\Gamma_{\text{DR}}}$ increases and thus the schedulability function decreases. Therefore, we consider dropping jobs of low-criticality tasks via linear search from the least to most important task.

4.3 IG-EDF-VD Schedulability Analysis

Since the algorithm treats undroppable tasks Γ_{UD} as if they are high-criticality tasks, we can consider the set Γ of tasks scheduled under IG-EDF-VD to be equivalent to a set Γ' scheduled by traditional EDF-VD where

$$U'^{\text{HI}}_{\Gamma_{\text{HI}}} = U^{\text{HI}}_{\Gamma_{\text{HI}}} + U^{\text{LO}}_{\Gamma_{\text{UD}}} \tag{11}$$

$$U'^{\text{LO}}_{\Gamma_{\text{HI}}} = U^{\text{LO}}_{\Gamma_{\text{HI}}} + U^{\text{LO}}_{\Gamma_{\text{UD}}} \tag{12}$$

$$U'^{\text{LO}}_{\Gamma_{\text{LO}}} = U^{\text{LO}}_{\Gamma_{\text{DR}}} \tag{13}$$

We use this to prove the sufficient schedulability condition used by IG-EDF-VD's analysis.

Lemma 4.1. *The following is sufficient for ensuring IG-EDF-VD successfully schedules all low-criticality behaviors of* Γ.

$$x \geq \frac{U^{LO}_{\Gamma_{HI}} + U^{LO}_{\Gamma_{UD}}}{1 - U^{LO}_{\Gamma_{DR}}} \tag{14}$$

Proof. Applying the schedulability condition for EDF-VD [1, Theorem 1], the following ensures IG-EDF-VD successfully schedules all low-criticality behaviors of Γ.

$$x \geq \frac{U'^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}}}{1 - U'^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}}}$$

Equations 12 and 13 imply that

$$x \geq \frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}}}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}}$$

□

Therefore, Algorithm 1 sets $x = \frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}}}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}}$ to ensure it successfully schedules all low-criticality behaviors of Γ.

Lemma 4.2. *The following is sufficient for ensuring IG-EDF-VD successfully schedules all high-criticality behaviors of* Γ.

$$xU^{LO}_{\Gamma_{DR}} + U^{HI}_{\Gamma_{HI}} + U^{LO}_{\Gamma_{UD}} \leq 1 \tag{15}$$

Proof. Applying [1, Theorem 2], the following condition is sufficient for ensuring IG-EDF-VD successfully schedules all high-criticality behaviors of Γ.

$$xU'^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} + U'^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}} \leq 1$$

Equations 11 and 13 imply that

$$xU^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}} + U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}} \leq 1$$

□

As such, Algorithm 1 will always ensure IG-EDF-VD successfully schedules all high-criticality behaviors of Γ. Now we can prove the IG-EDF-VD schedulability function defined in Section 4.2 by combining Lemma 4.1 and Lemma 4.2:

Theorem 4.3. *If the IG-EDF-VD schedulability function*

$$\mathcal{B}(\Gamma, \Gamma_{UD}) \leq 1$$

then a schedule for the system Γ *is feasible under IG-EDF-VD.*

Proof. For $x = \frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}}}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}}$, IG-EDF-VD successfully schedules all low-criticality behaviors of Γ from Lemma 4.1. Now we have $xU^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}} + U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}} \leq 1$, so IG-EDF-VD successfully schedules all high-criticality behaviors of Γ from Lemma 4.2. Therefore, a schedule for the system Γ is feasible. □

Now that we have shown a sufficient schedulability function, we prove a result related to the observation we made in Section 4.2: as we move tasks from Γ_{UD} to Γ_{DR}, the schedulability function decreases. Therefore, it is valid to consider dropping jobs of low-criticality tasks in order from the least important to the most important task, terminating once a schedulable configuration is found. To prove this, we make use of the following properties. First, we assume $U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} \leq 1$; otherwise, the system is trivially unschedulable. Second, we assume $U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} < 1$; otherwise, no critical task can be scheduled even in low-criticality mode.

Lemma 4.4. *For a task set* Γ *with* $U^{LO}_{\Gamma_{HI}} + U^{LO}_{\Gamma_{LO}} \leq 1$ *and* $U^{LO}_{\Gamma_{DR}} < 1$, *the IG-EDF-VD schedulability function* $\mathcal{B}(\Gamma, \Gamma_{UD})$ *is monotonically non-increasing with respect to* $U^{LO}_{\Gamma_{DR}}$ *as tasks are moved from* Γ_{UD} *to* Γ_{DR}.

Proof. We prove this by showing that the derivative with respect to $U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}$ is non-positive for Γ. Since $U^{\mathrm{HI}}_i = U^{\mathrm{LO}}_i$ for all low-criticality tasks, $U^{\mathrm{HI}}_{\Gamma_{\mathrm{UD}}} = U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}}$. This implies that the schedulability function $\mathcal{B}(\Gamma, \Gamma_{\mathrm{UD}})$ is equivalent to:

$$\frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}}}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}} \cdot U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}} + U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}}$$

Since we partitioned Γ_{LO} into Γ_{DR} and Γ_{UD}, the total utilization of Γ_{DR} and Γ_{UD} must equal that of Γ_{LO}. Then from $U^{\mathrm{LO}}_{\Gamma_{\mathrm{UD}}} = U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}$, we have:

$$\mathcal{B}(\Gamma, \Gamma_{\mathrm{UD}}) = \frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + (U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}})}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}} \cdot U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}} + (U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}) + U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}}$$

As the set of low- and high-criticality tasks does not change with respect to the partition of low-criticality tasks, we may consider $U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}}$, $U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}}$, and $U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}}$ to be constants. Therefore, we can rearrange the schedulability function as follows:

$$(U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}}) \cdot \frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}} - \frac{(U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}})^2}{1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}} - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}} + (U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} + U^{\mathrm{HI}}_{\Gamma_{\mathrm{HI}}})$$

Taking the derivative, we get:

$$\begin{aligned}
&\frac{d}{dU^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}} \mathcal{B}(\Gamma, \Gamma_{\mathrm{UD}}) \\
=&(U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}}) \cdot \frac{1}{(1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}})^2} - (\frac{1}{(1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}})^2} - 1) - 1 \\
=&\frac{U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} - 1}{(1 - U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}})^2}
\end{aligned}$$

Since $U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} \leq 1$, the numerator is non-positive. Also, as $U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}} \neq 1$, the denominator must be positive. Therefore, the derivative must be non-positive, so the schedulability function is monotonically non-increasing with $U^{\mathrm{LO}}_{\Gamma_{\mathrm{DR}}}$. □

Theorem 4.5. *Let* Γ *be a task set and* Γ_{UD}, Γ_{DR} *represent a partition of its low-criticality tasks. Let* $\Gamma'_{UD}, \Gamma'_{DR}$ *be another partition where* τ_i *is moved from* Γ_{UD} *to* Γ_{DR}, *i.e,* $\Gamma'_{UD} = \Gamma_{UD} \setminus \tau_i$ *and* $\Gamma'_{DR} = \Gamma_{DR} \cup \tau_i$. *Then* $\mathcal{B}(\Gamma, \Gamma'_{UD}) \leq 1$ *if* $\mathcal{B}(\Gamma, \Gamma_{UD}) \leq 1$.

Proof. Clearly, if $\mathcal{B}(\Gamma, \Gamma_{\mathrm{UD}}) \leq 1$ then $U^{\mathrm{LO}}_{\Gamma_{\mathrm{HI}}} + U^{\mathrm{LO}}_{\Gamma_{\mathrm{LO}}} \leq 1$. Applying Lemma 4.4, we have

$$\mathcal{B}(\Gamma, \Gamma'_{\mathrm{UD}}) \leq \mathcal{B}(\Gamma, \Gamma_{\mathrm{UD}})$$

By the transitive property, we get

$$\mathcal{B}(\Gamma, \Gamma'_{\mathrm{UD}}) \leq 1$$

□

	χ_i	I_i	T_i	$U_i^{\text{LO,min}}$	$U_i^{\text{LO,max}}$	$U_i^{\text{HI,min}}$	$U_i^{\text{HI,max}}$	ϕ_i
τ_1	HI	-	91.735	0.255	0.255	0.518	0.518	-
τ_2	HI	-	4.286	0.095	0.095	0.132	0.132	-
τ_3	LO	1	1.710	0.225	0.245	0.225	0.245	0.030
τ_4	LO	2	92.718	0.082	0.111	0.082	0.111	4.028
τ_5	LO	3	2.300	0.092	0.094	0.092	0.094	0.002

Table 2: Sample elastic taskset

Therefore, a schedule for Γ will still be feasible if a task is moved from Γ_{UD} to Γ_{DR} per Theorem 4.3.

5 EG-EDF-VD

This section introduces the Elastic Graceful EDF-VD (EG-EDF-VD) algorithm, which extends the IG-EDF-VD algorithm from Section 4 to the elastic system model in Section 3.2. The idea is still to find the maximal set of low-criticality tasks that can continue to execute in high-criticality mode. EG-EDF-VD takes advantage of elasticity to do so, compressing task utilizations to the extent necessary to further reduce the number of suspended tasks.

We first motivate this algorithm in Section 5.1 by considering an elastic version of Example 2. We then introduce the EG-EDF-VD algorithm in Section 5.2. In Section 5.3, we prove several properties of our elastic mixed-criticality system model; these are leveraged in Section 5.4 to develop an algorithm that efficiently finds the minimum system compression level Φ deemed schedulable. In Section 5.5, we discuss how EG-EDF-VD schedules tasks across a mode switch.

5.1 Motivation

Example 3. *Let us revisit Example 2. Now, all low-criticality tasks are also elastic, with parameters shown in Table 2.*

Let $\Gamma^{\min}$ denote the task system where all elastic tasks are fully compressed (i.e., $\Phi \geq \max_i\{\phi_i\}$) and $\Gamma^{\max}$ denote the task system where no tasks are compressed at all (i.e., $\Phi = 0$), which is equivalent to the inelastic system Γ in Example 2.

Let us treat tasks τ_4 and τ_5 as high-criticality, i.e., $\Gamma_{\text{UD}} = \{\tau_4, \tau_5\}$. Applying the IG-EDF-VD schedulability function,

$$\mathcal{B}(\Gamma^{\max}, \Gamma_{\text{UD}}) = \frac{0.555}{1 - 0.245} \cdot 0.245 + 0.855 \approx 1.035$$

and

$$\mathcal{B}(\Gamma^{\min}, \Gamma_{\text{UD}}) = \frac{0.524}{1 - 0.225} \cdot 0.225 + 0.824 \approx 0.976$$

As $\mathcal{B}(\Gamma^{\max}, \Gamma_{\text{UD}}) > 1$, IG-EDF-VD does not guarantee schedulability. However, the fully compressed system is schedulable since $\mathcal{B}(\Gamma^{\min}, \Gamma_{\text{UD}}) \leq 1$. Therefore, we propose the EG-EDF-VD algorithm, which extends IG-EDF-VD by leveraging elasticity to drop even fewer tasks.

5.2 EG-EDF-VD Algorithm

The primary goal of the EG-EDF-VD algorithm is still to find the maximal subset (in order of importance) of undroppable low-criticality tasks Γ_{UD}. It first compresses all elastic tasks to their minimum utilizations, then uses Algorithm 1 to find the maximal resulting assignment of tasks to Γ_{UD}. Once this is found, EG-EDF-VD finds the minimum system compression level Φ for which the partition remains schedulable. This is done by binary search with tunable precision ϵ using the compression algorithm in Section 5.4.

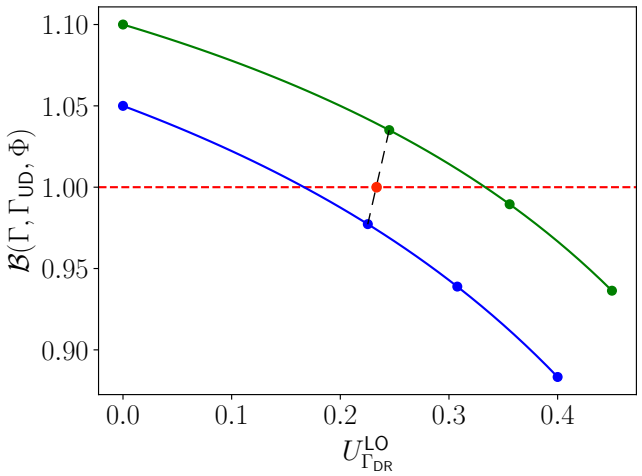

Figure 2: Schedulability function against utilization of droppable tasks of a sample elastic task set

From Equation 7, the utilization of each task τ_i for a given value of Φ at criticality level χ can be expressed as:

$$U_i^{\chi}(\Phi) \stackrel{\text{def}}{=} \max\left(U_i^{\chi,\max} - \Phi \cdot E_i^{\chi}, U_i^{\chi,\min}\right) \tag{16}$$

Then for each of $\chi \in \{\text{LO}, \text{HI}\}$ and $\Gamma_k \in \{\Gamma_{\text{HI}}, \Gamma_{\text{UD}}, \Gamma_{\text{DR}}\}$, we can now include the system compression level in the corresponding total utilization parameter as:

$$U_{\Gamma_k}^{\chi}(\Phi) = \sum_{\tau_i \in \Gamma_k} U_i^{\chi}(\Phi)$$

To simplify notation in the remainder of this section, we abbreviate $U_{\Gamma_k}^{\chi}(\Phi)$ as $U_{\Gamma_k}^{\chi}$. With that, we define the EG-EDF-VD schedulability function for a mixed-criticality system Γ at system compression level Φ as follows.

$$\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi) = \frac{U_{\Gamma_{\text{HI}}}^{\text{LO}} + U_{\Gamma_{\text{UD}}}^{\text{LO}}}{1 - U_{\Gamma_{\text{DR}}}^{\text{LO}}} \cdot U_{\Gamma_{\text{DR}}}^{\text{LO}} + U_{\Gamma_{\text{UD}}}^{\text{HI}} + U_{\Gamma_{\text{HI}}}^{\text{HI}} \tag{17}$$

One may note that this is exactly the same as the function for IG-EDF-VD except that utilization is now a function of the system compression level Φ. Because this expression includes task utilizations for both low- and high-criticality system modes, we consider the problem of finding a *single* value of Φ that, when applied to both criticality levels, guarantees schedulability. Algorithm 2 in Section 5.4 finds the minimum such value for the maximal assignment of tasks to Γ_{UD}.

Figure 2 shows how the EG-EDF-VD schedulability function for the set of tasks in Example 3 changes with Φ. The schedulability function for $\Gamma^{\min}$ (where $\Phi \geq \max_i\{\phi_i\}$) is shown as the bottom

(blue) curve, and that for $\Gamma^{\max}$ (where $\Phi = 0$) is shown as the top (green) curve.

Again, each point represents an assignment of tasks to Γ_{UD}. The x-axis represents the value $U^{\text{LO}}_{\Gamma_{\text{DR}}}$ of that partition, and the y-axis is the schedulability function value $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi)$. A horizontal reference line shows the schedulability bound where $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi) = 1$. For these tasks, EG-EDF-VD first finds the maximal schedulable partition, i.e., the leftmost point under the reference line on the bottom curve. It then decompresses the system, finding the minimum value of Φ for which the bound is reached, as shown by the red dot where the dashed line connecting the top and bottom curves intersects the bound.

5.3 Properties of the Elastic Task Model

Orthogonal to criticality, we partition the set of tasks Γ into two subsets, $\Gamma_{\text{FIX}}(\Phi)$ and $\Gamma_{\text{VAR}}(\Phi)$, as in [23]. $\Gamma_{\text{FIX}}(\Phi)$ are tasks that have already reached their minimum utilization $U_i^{\chi}(\Phi) = U_i^{\chi,\min}$; $\Gamma_{\text{VAR}}(\Phi)$ are tasks that can still be compressed, i.e., $U_i^{\chi}(\Phi) > U_i^{\chi,\min}$. Note that inelastic tasks are always in $\Gamma_{\text{FIX}}(\Phi)$ for any value of Φ. Because we restrict ϕ_i to remain constant across criticality levels, the assignment of tasks to $\Gamma_{\text{FIX}}(\Phi)$ and $\Gamma_{\text{VAR}}(\Phi)$ also remains unchanged across a mode switch. We prove this by showing that one can determine whether an elastic task τ_i is variable or fixed by simply comparing the system compression level Φ and the task's maximum compression level ϕ_i.

Lemma 5.1. *An elastic task $\tau_i \in \Gamma_{FIX}(\Phi)$ if and only if $\phi_i \leq \Phi$. Equivalently, $\tau_i \in \Gamma_{VAR}(\Phi)$ if and only if $\phi_i > \Phi$.*

Proof. If $\tau_i \in \Gamma_{\text{FIX}}(\Phi)$, then $U_i^{\chi}(\Phi) = U_i^{\chi,\min}$. Since we know $U_i^{\chi}(\Phi) = \max\left(U_i^{\chi,\max} - \Phi \cdot E_i^{\chi}, U_i^{\chi,\min}\right)$, we can derive

$$U_i^{\chi,\max} - \Phi \cdot E_i^{\chi} \leq U_i^{\chi,\min}$$
$$\frac{U_i^{\chi,\max} - U_i^{\chi,\min}}{E_i^{\chi}} \leq \Phi$$
$$\phi_i \leq \Phi$$

If $\phi_i \leq \Phi$, by the definition of ϕ_i and Φ, we get

$$\frac{U_i^{\chi,\max} - U_i^{\chi,\min}}{E_i^{\chi}} \leq \frac{U_i^{\chi,\max} - U_i^{\chi}(\Phi)}{E_i^{\chi}}$$
$$U_i^{\chi,\max} - U_i^{\chi,\min} \leq U_i^{\chi,\max} - U_i^{\chi}(\Phi)$$
$$U_i^{\chi}(\Phi) \leq U_i^{\chi,\min}$$

By definition, $U_i^{\chi}(\Phi) = \max(U_i^{\chi,\max} - \Phi \cdot E_i^{\chi}, U_i^{\chi,\min})$, which implies that $U_i^{\chi}(\Phi) \geq U_i^{\chi,\min}$. Therefore, it must be the case that $U_i^{\chi}(\Phi) = U_i^{\chi,\min}$ and we have $\tau_i \in \Gamma_{\text{FIX}}(\Phi)$.

Since $\tau_i \in \Gamma_{\text{FIX}}(\Phi)$ if and only if $\phi_i \leq \Phi$, the following is equivalent: $\tau_i \in \Gamma_{\text{VAR}}(\Phi)$ if and only if $\phi_i > \Phi$. □

Since ϕ_i remains constant across criticality levels, the partition will remain the same upon a mode switch for a fixed value of Φ. We now show that for two system compression levels Φ_1 and Φ_2, the partition of tasks into Γ_{FIX} and Γ_{VAR} is the same as long as there is no task τ_i for which $\phi_i \in (\Phi_1, \Phi_2]$.

Theorem 5.2. *Let Γ be a task system and $\Phi_1 \leq \Phi_2$ be two system compression levels of Γ. If there does not exist a task τ_i in Γ such that $\Phi_1 < \phi_i \leq \Phi_2$, then $\Gamma_{FIX}(\Phi_1) = \Gamma_{FIX}(\Phi_2)$ and $\Gamma_{VAR}(\Phi_1) = \Gamma_{VAR}(\Phi_2)$.*

Proof. Because there does not exist any task τ_i in Γ such that $\Phi_1 < \phi_i \leq \Phi_2$, it follows that for all tasks τ_i, $\phi_i \leq \Phi_1 \leq \Phi_2$ or $\Phi_1 \leq \Phi_2 < \phi_i$. If $\tau_i \in \Gamma_{\text{FIX}}(\Phi_1)$, then $\phi_i \leq \Phi_1$ by Lemma 5.1. Therefore, it must be the case that $\phi_i \leq \Phi_1 \leq \Phi_2$. By the same lemma, we get $\tau_i \in \Gamma_{\text{FIX}}(\Phi_2)$. By symmetry, if $\tau_i \in \Gamma_{\text{FIX}}(\Phi_2)$, then we have $\tau_i \in \Gamma_{\text{FIX}}(\Phi_1)$. Therefore, $\Gamma_{\text{FIX}}(\Phi_1) = \Gamma_{\text{FIX}}(\Phi_2)$; equivalently $\Gamma_{\text{VAR}}(\Phi_1) = \Gamma_{\text{VAR}}(\Phi_2)$. □

Finally, we show that $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi)$ is monotonically non-increasing with Φ.

Theorem 5.3. *For a task set Γ with $U^{LO}_{\Gamma_{DR}} \leq 1$, the function $\mathcal{B}(\Gamma, \Gamma_{UD}, \Phi)$ defined in Equation 17 is monotonically non-increasing with Φ.*

Proof. From Equation 17,

$$\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi) = \left(U^{\text{LO}}_{\Gamma_{\text{HI}}} + U^{\text{LO}}_{\Gamma_{\text{UD}}}\right) \frac{U^{\text{LO}}_{\Gamma_{\text{DR}}}}{1 - U^{\text{LO}}_{\Gamma_{\text{DR}}}} + U^{\text{HI}}_{\Gamma_{\text{UD}}} + U^{\text{HI}}_{\Gamma_{\text{HI}}}$$

For two values of Φ, $\Phi_1 \leq \Phi_2$, we know from Equation 16 that $U_i(\Phi_1) \geq U_i(\Phi_2)$. Furthermore, for $0 \leq U^{\text{LO}}_{\Gamma_{\text{DR}}} < 1$, $\frac{U^{\text{LO}}_{\Gamma_{\text{DR}}}}{1-U^{\text{LO}}_{\Gamma_{\text{DR}}}}$ increases as $U^{\text{LO}}_{\Gamma_{\text{DR}}}$ increases. Thus, $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi)$ is monotonically non-increasing as Φ_i increases. □

These properties allow us to develop an algorithm, based on binary search, that finds the smallest system compression level Φ—to within some arbitrary degree of precision ϵ—for which a set of tasks partitioned according to IG-EDF-VD in Algorithm 1 remains schedulable according to the bound in Equation 3.

5.4 Elastic Compression Algorithm

Given an elastic mixed-criticality task system Γ characterized as in Section 3.2, EG-EDF-VD begins by using Algorithm 1 to find the *maximal* assignment (in order of importance) of tasks τ_i to Γ_{UD}. It does so by compressing each elastic task τ_i to its minimum utilization $U_i^{\chi,\min}$, then applying Algorithm 1 in Section 4.2 to the resulting task set $\Gamma^{\min}$. Algorithm 2 is then invoked to find the smallest system compression level Φ for which that partition remains schedulable.

If $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, 0) \leq 1$, it immediately returns 0 as the partition is already schedulable without any compression. Otherwise, it iterates through all elastic tasks τ_i in non-increasing order of their maximum compression levels ϕ_i, computing $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i)$. It uses a variable Φ_{prev}, initialized to 0, to track the previously-tested value of Φ.

If $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i)$ is 1, then $\Phi = \phi_i$ is the smallest value deemed schedulable due to the monotonicity property of Theorem 5.3. Otherwise, if $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i) < 1$, we know from Theorem 5.2 that there is no task τ_j such that $\Phi_{\text{prev}} < \phi_j < \phi_i$ because we are iterating over elastic tasks in order of ϕ_i. Therefore, by monotonicity of $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i)$, the algorithm uses binary search to find the minimum (with precision ϵ) schedulable value of Φ in the range $(\Phi_{\text{prev}}, \phi_i]$.

Algorithm 2 Compression

Require: A schedulable partition of an elastic implicit-deadline sporadic task system Γ.

Ensure: Minimum value of Φ (with precision ϵ) for which $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi) \leq 1$.

1: $\Gamma_{\text{FIX}} \leftarrow \{\tau_i \in \Gamma | \tau_i \text{ is inelastic}\}$ ▷ all inelastic tasks
2: $\Gamma_{\text{VAR}} \leftarrow \{\tau_i \in \Gamma | \tau_i \text{ is elastic}\}$ ▷ all elastic tasks
3: **if** $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, 0) \leq 1$ **then return** 0 ▷ already schedulable
4: Sort tasks $\tau_i \in \Gamma$ in ascending order by ϕ_i
5: $\Phi_{\text{prev}} \leftarrow 0$
6: **for all** $\tau_i \in \Gamma_{\text{VAR}}$ **do**
7: **if** $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i) = 1$ **then**
8: **return** ϕ_i ▷ exact match
9: **else if** $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i) < 1$ **then**
10: $\Phi \leftarrow \text{BinarySearch}(\phi_{\text{prev}}, \phi_i, \epsilon)$
11: **return** Φ ▷ find in interval
12: **end if**
13: Move τ_i from Γ_{VAR} to Γ_{FIX}
14: $\Phi_{\text{prev}} \leftarrow \phi_i$
15: **end for**
16: **return** Infeasible ▷ not schedulable

However, if $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i) > 1$, it moves τ_i from Γ_{VAR} to Γ_{FIX}, updates Φ_{prev}, and proceeds to the next task.

For a set Γ of n tasks, the time complexity of sorting the tasks by ϕ_i is $O(n \log n)$. Then the loop in Lines 6–15 requires no more than n iterations. Each iteration except the last computes the bound function $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \phi_i)$ once. The last iteration requires at most $\log\left(\frac{\phi_{\max}}{\epsilon}\right)$ steps to find the solution via binary search. Since the bound function requires $O(n)$ time to compute, the overall time complexity of Algorithm 2 is $O\left(n^2 + n \log\left(\frac{\phi_{\max}}{\epsilon}\right)\right)$.

5.5 Mode Switch Behavior

During runtime, EG-EDF-VD executes similarly to IG-EDF-VD: a job arriving at time t_a is assigned a virtual deadline $t_a + xT_i$ if $\tau_i \in \Gamma_{\text{HI}}$ where x is computed according to Equation 4.1 for the current system compression level Φ. If a job of a high-criticality task executes for longer than its assigned C_i^{LO} for the system compression level Φ, tasks in Γ_{DR} are suspended and the remaining jobs are scheduled according to their actual deadlines $t_a + T_i$. Since we only consider workload-elastic tasks, T_i is constant across criticality levels.

Tasks that are not suspended have their workloads adjusted to values derived from the utilizations assigned by EG-EDF-VD under high-criticality mode. Because these assignments are computed by the algorithm during offline analysis, no additional computation is required during the mode switch. (Note, however, that the cost of communicating and enforcing the new workload assignment is application-specific.) Furthermore, after the mode switch, a high-criticality task's execution time budget does not *decrease* since Φ is constant. This means that the amount of time any job executed prior to the mode switch does not exceed the budget it is assigned in high-criticality mode, so schedulability remains guaranteed.

Lemma 5.4. *For a given value of Φ, for any task τ_i,*

$$U_i^{HI}(\Phi) \geq U_i^{LO}(\Phi)$$

Proof. If $U_i^{\text{LO}}(\Phi) = U_i^{\text{LO,min}}$, then by Lemma 5.1, $U_i^{\text{HI}}(\Phi) = U_i^{\text{HI,min}}$, and we require $U_i^{\text{HI, min}} \geq U_i^{\text{LO, min}}$.

For tasks τ_i with $U_i^{\chi}(\Phi) > U_i^{\chi,\min}$, we have $U_i^{\chi}(\Phi) = U_i^{\chi,\max} - \Phi \cdot E_i^{\chi}$. And by definition, $U_i^{\chi,\max} = U_i^{\chi,\min} + \phi_i \cdot E_i^{\chi}$. Rearranging, we get $U_i^{\chi,\min} = U_i^{\chi,\max} - \phi_i \cdot E_i^{\chi}$. Then,

$$\begin{aligned}
U_i^{\text{HI,min}} &\geq U_i^{\text{LO,min}} \\
U_i^{\text{HI,max}} - \phi_i \cdot E_i^{\text{HI}} &\geq U_i^{\text{LO,max}} - \phi_i \cdot E_i^{\text{LO}} \\
\phi_i &\leq \frac{U_i^{\text{HI,max}} - U_i^{\text{LO,max}}}{E_i^{\text{HI}} - E_i^{\text{LO}}} \\
\Phi &< \frac{U_i^{\text{HI,max}} - U_i^{\text{LO,max}}}{E_i^{\text{HI}} - E_i^{\text{LO}}} \\
U_i^{\text{HI,max}} - \Phi \cdot E_i^{\text{HI}} &> U_i^{\text{LO,max}} - \Phi \cdot E_i^{\text{LO}} \\
U_i^{\text{HI}} &\geq U_i^{\text{LO}}
\end{aligned}$$

Therefore, $U_i^{HI}(\Phi) \geq U_i^{LO}(\Phi)$ for all tasks τ_i at any given system compression level Φ. □

Thus, the behavior of EG-EDF-VD is correct.

6 Evaluation

In this section, we illustrate the effectiveness of the proposed IG-EDF-VD and EG-EDF-VD algorithms by evaluating sets of randomly-generated synthetic tasks, parameterized according to the elastic mixed-criticality model in Section 3.2. For each set of tasks, we compare the number of low-criticality tasks that must be suspended across a mode switch, demonstrating the improvements realized over EDF-VD. For this simple empirical study, each set of tasks includes 5 low-criticality and 5 high-criticality tasks. Task utilizations are assigned as follows:

(1) The total maximum utilization of low-criticality tasks is fixed at $U_{\Gamma_{\text{LO}}}^{\max} = 0.4 - \varepsilon$; we use the Dirichlet-Rescale (DRS) algorithm [13] to distribute this in an unbiased random fashion across their individual utilizations. The value ε is used to compensate for numerical instability due to floating-point rounding errors; we set $\varepsilon = 0.001$.
(2) The total minimum utilization of low-criticality tasks is fixed at $U_{\Gamma_{\text{LO}}}^{\min} = 0.35 - \varepsilon$. We use DRS to distribute this across the individual utilization values such that each value $U_i^{\min}$ does not exceed $U_i^{\max}$.
(3) The total maximum utilization of high-criticality tasks in high-criticality mode $U_{\Gamma_{\text{HI}}}^{\text{HI, max}}$ is swept over the range $0.76 - \varepsilon$ to $1.1 - \varepsilon$ with a step size of 0.01; for each value, we generate 10 sets of tasks. We again use DRS to distribute this across the $U_i^{\text{HI, max}}$ values.
(4) The total minimum utilization of high-criticality tasks in high-criticality mode is fixed at $U_{\Gamma_{\text{HI}}}^{\text{HI, min}} = 0.75 - \varepsilon$; we use DRS to distribute this such that each value $U_i^{\text{HI, min}}$ does not exceed $U_i^{\text{HI, max}}$.
(5) The total maximum utilization of high-criticality tasks in low-criticality mode is fixed at $U_{\Gamma_{\text{HI}}}^{\text{LO, max}} = 0.2 - \varepsilon$; this is again

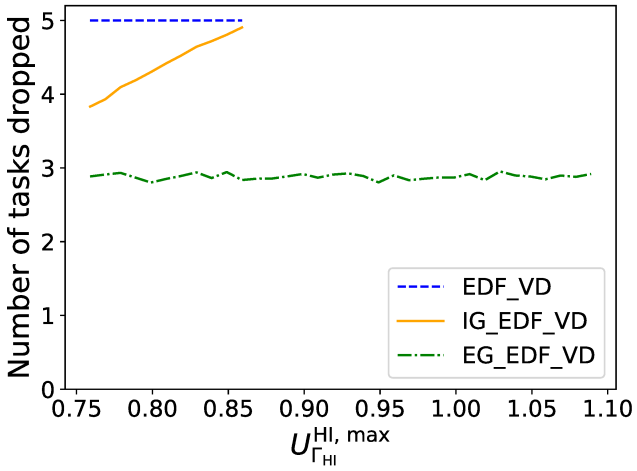

Figure 3: Number of tasks dropped

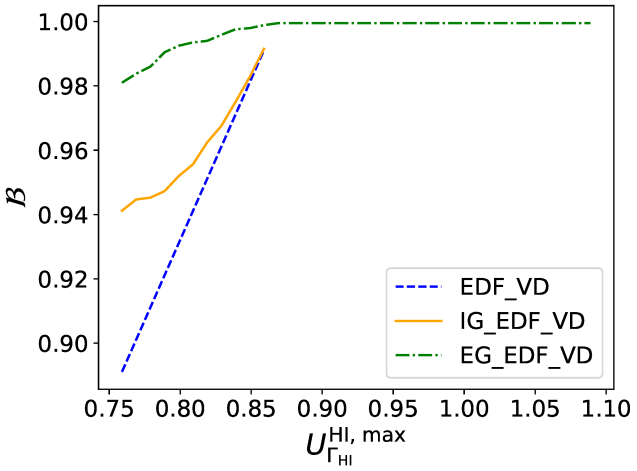

Figure 4: Schedulability function

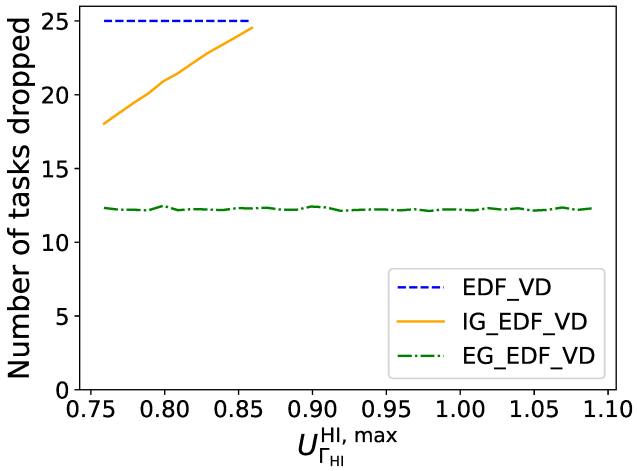

Figure 5: Number of tasks dropped

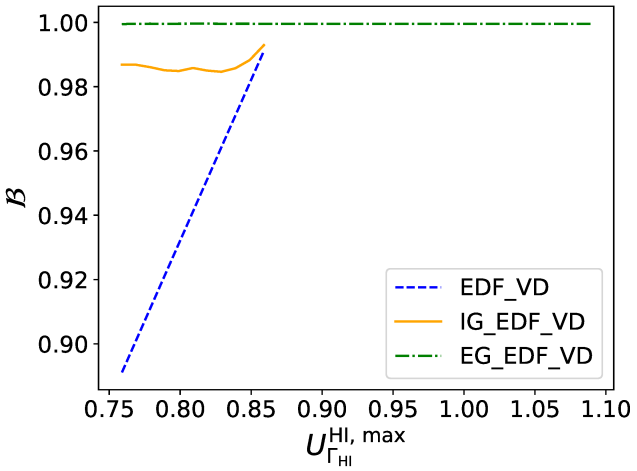

Figure 6: Schedulability function

distributed with DRS such that each value $U_i^{\text{LO, max}}$ does not exceed $U_i^{\text{HI, max}}$.

(6) The total minimum utilization of high-criticality tasks in low-criticality mode is fixed at $U^{\text{LO, min}}_{\Gamma_{\text{HI}}} = 0.15 - \varepsilon$. We use DRS to distribute this such that each value $U_i^{\text{LO, min}}$ does not exceed $U_i^{\text{LO, max}}$ or $U_i^{\text{HI, min}}$.

Each task's period T_i is generated using a log-uniform distribution (per [10]) from the range $[1, 1000]$. Workloads C_i are then derived from the utilization and period. The maximum compression level ϕ_i of each task is generated uniformly from the range $[0, 1]$; elasticity is then derived using Equation 9. A random sequence of importance values I_i are assigned to low-criticality tasks.

Figure 3 shows — for the 1000 task sets generated for each value of $U^{\text{HI, max}}_{\Gamma_{\text{HI}}}$ — the mean number of low-criticality tasks that must be dropped. Whereas EDF-VD will always drop all low-criticality tasks, we can see that IG-EDF-VD is able to execute some low-criticality tasks across a mode switch by incorporating importance. However, IG-EDF-VD drops more low-criticality tasks as $U^{\text{HI}}_{\Gamma_{\text{HI}}}$ increases until all low-criticality tasks have to be dropped.

For EDF-VD and IG-EDF-VD, which cannot take advantage of task elasticity, each elastic task is treated as *inelastic*, i.e., only the system compression level $\Phi = 0$ is considered. On the other hand, the number of low-criticality tasks dropped by EG-EDF-VD remains roughly the same regardless of $U^{\text{HI, max}}_{\Gamma_{\text{HI}}}$ because it takes advantage of task elasticity to compress utilizations.

Figure 4 shows the mean value of the schedulability function $\mathcal{B}(\Gamma)$ achieved by each algorithm. For the evaluated task sets, both EDF-VD and IG-EDF-VD can schedule up to $U^{\text{HI}}_{\Gamma_{\text{HI}}} = 0.86 - \varepsilon$, after which $\mathcal{B}(\Gamma) > 1$ for EDF-VD and $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}) > 1$ for IG-EDF-VD. Below this, the value of the schedulability function for IG-EDF-VD lies above that of EDF-VD (but does not exceed 1) as a result of treating undroppable tasks as high-criticality tasks. On the other hand, EG-EDF-VD can schedule task sets beyond $U^{\text{HI, max}}_{\Gamma_{\text{HI}}} = 0.86 - \varepsilon$ by compressing task utilizations until $\mathcal{B}(\Gamma, \Gamma_{\text{UD}}, \Phi) = 1$. We can also see that the value of the schedulability function for IG-EDF-VD closely follows the schedulability bound as it tries to decompress as much as possible.

To illustrate the scalability of our proposed approaches, we repeated this experiment with task sets having 25 low-criticality and 25 high-criticality tasks, keeping all other parameters the same.

Results are shown in Figures 5 and 6. Comparing Figure 5 to Figure 3, the proportion of tasks dropped by each algorithm does not change significantly. However, as the number of tasks increases, the number of possible partitions increases, allowing us to find a partition that takes the schedulability function closer to the bound. In other words, the points along the curves illustrated in Figures 1 and 2 are more densely placed. This is reflected by Figure 6, which compared to Figure 4 shows IG-EDF-VD and EG-EDF-VD bringing the schedulability function closer to 1 for smaller maximum utilizations.

7 Conclusions

In this work, we proposed the IG-EDF-VD scheduling algorithm that allows only the least important low-criticality tasks to be dropped upon a mode switch while the schedule is still feasible. We then extended it to the EG-EDF-VD scheduling algorithm that compresses the utilizations of workload-elastic tasks to allow fewer low-criticality tasks to be dropped upon a mode switch.

As future work, we will extend our algorithms to existing analysis of EDF-VD for more than two criticality levels [2, 30]. We will also explore period-elastic tasks, for which additional analysis will be required if period can change across criticality levels. Additionally, we plan to relax the requirement that the maximum compression level ϕ_i for each task remains constant across criticality levels. Finally, we will explore improvements to the time complexity of Algorithm 2.

Acknowledgments

The research presented in this paper was supported in part by NSF grants CPS-2229290 and CNS-2141256, NASA award 80NSSC21K1741, and a Washington University seed grant.

References

[1] Sanjoy Baruah, Vincenzo Bonifaci, Gianlorenzo DAngelo, Haohan Li, Alberto Marchetti-Spaccamela, Suzanne Van Der Ster, and Leen Stougie. 2012. The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems. In *2012 24th Euromicro Conference on Real-Time Systems*. IEEE, 145–154.

[2] Sanjoy Baruah, Vincenzo Bonifaci, Gianlorenzo D'angelo, Haohan Li, Alberto Marchetti-Spaccamela, Suzanne Van Der Ster, and Leen Stougie. 2015. Preemptive uniprocessor scheduling of mixed-criticality sporadic task systems. *Journal of the ACM (JACM)* 62, 2 (2015), 1–33.

[3] Sanjoy K Baruah, Vincenzo Bonifaci, Gianlorenzo d'Angelo, Alberto Marchetti-Spaccamela, Suzanne Van Der Ster, and Leen Stougie. 2011. Mixed-criticality scheduling of sporadic task systems. In *Algorithms–ESA 2011: 19th Annual European Symposium, Saarbrücken, Germany, September 5-9, 2011. Proceedings 19*. Springer, 555–566.

[4] James H. Buckley, Jeremy Buhler, and Roger D. Chamberlain. 2024. The Advanced Particle-astrophysics Telescope (APT): Computation in Space. In *Proc. of 21st ACM International Conference on Computing Frontiers Workshops and Special Sessions*. 122–127. https://doi.org/10.1145/3637543.3652980 Invited paper at the conference Special Session on Computer Architectures in Space (CompSpace).

[5] Giorgio C Buttazzo, Giuseppe Lipari, and Luca Abeni. 1998. Elastic task model for adaptive rate control. In *Proceedings 19th IEEE Real-Time Systems Symposium (Cat. No. 98CB36279)*. IEEE, 286–295.

[6] Giorgio C Buttazzo, Giuseppe Lipari, Marco Caccamo, and Luca Abeni. 2002. Elastic scheduling for flexible workload management. *IEEE Trans. Comput.* 51, 3 (2002), 289–302.

[7] 2011. *Software Considerations in Airborne Systems and Equipment Certification*. RTCA, Inc.

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

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

[10] P. Emberson, R. Stafford, and R.I. Davis. 2010. Techniques For The Synthesis Of Multiprocessor Tasksets. In *WATERS workshop at the Euromicro Conference on Real-Time Systems*. 6–11. 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems ; Conference date: 06-07-2010.

[11] Alexandre Esper, Geoffrey Nelissen, Vincent Nélis, and Eduardo Tovar. 2015. How realistic is the mixed-criticality real-time system model?. In *Proceedings of the 23rd International Conference on Real Time and Networks Systems*. 139–148.

[12] Tom Fleming and Alan Burns. 2014. Incorporating the notion of importance into mixed criticality systems. In *Proc. 2nd Workshop on Mixed Criticality Systems (WMC), RTSS*. 33–38.

[13] David Griffin, Iain Bate, and Robert I Davis. 2020. Generating utilization vectors for the systematic evaluation of schedulability tests. In *2020 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 76–88.

[14] Ye Htet, Marion Sudvarg, Jeremy Buhler, Roger D. Chamberlain, and James H. Buckley. 2023. Localization of Gamma-ray Bursts in a Balloon-Borne Telescope. In *Proc. of Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W)*. 395–398. https://doi.org/10.1145/3624062.3624107 Presented at 1st Workshop on Enabling Predictive Science with Optimization and Uncertainty Quantification in HPC (EPSOUQ-HPC), Denver, CO, USA.

[15] 2010. *Functional safety of electrical/electronic/programmable electronic safety-related systems*. IEC.

[16] 2018. *Road vehicles - Functional safety*. ISO.

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

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

[19] James Orr and Sanjoy Baruah. 2019. Multiprocessor scheduling of elastic tasks. In *Proceedings of the 27th International Conference on Real-Time Networks and Systems*. 133–142.

[20] James Orr, Chris Gill, Kunal Agrawal, Sanjoy Baruah, et al. 2018. Elasticity of Workloads and Periods of Parallel Real-Time Tasks. In *Proc. of 26th International Conference on Real-Time Networks and Systems*. ACM, 61–71. https://doi.org/10.1145/3273905.3273915

[21] Hang Su and Dakai Zhu. 2013. An Elastic Mixed-Criticality task model and its scheduling algorithm. In *2013 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 147–152. https://doi.org/10.7873/DATE.2013.043

[22] Marion Sudvarg, Jeremy Buhler, Roger D. Chamberlain, Chris Gill, James Buckley, and Wenlei Chen. 2023. Parameterized Workload Adaptation for Fork-Join Tasks with Dynamic Workloads and Deadlines. In *Proc. of IEEE 29th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. 232–242. https://doi.org/10.1109/RTCSA58653.2023.00035

[23] Marion Sudvarg, Chris Gill, and Sanjoy Baruah. 2021. Linear-time admission control for elastic scheduling. *Real-Time Systems* 57, 4 (2021), 485–490.

[24] Marion Sudvarg, Chris Gill, and Sanjoy Baruah. 2024. Improved implicit-deadline elastic scheduling. In *Proceedings of the 14th IEEE International Symposium on Industrial Embedded Systems (SIES 2024). IEEE.*

[25] Marion Sudvarg, Ye Htet, Sanjoy Baruah, Jeremy Buhler, Roger Chamberlain, Chris Gill, and Jim Buckley. 2024. Adaptive Execution for Real-Time Observations of Astrophysical Transients. In *Proc. of 13th International Real-Time Scheduling Open Problems Seminar (RTSOPS)*.

[26] M. Sudvarg, A. Li, D. Wang, S. Baruah, J. Buhler, P. Ekberg, C. Gill, and N. Zhang. 2024. Elastic Scheduling for Harmonic Task Systems. In *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*.

[27] Marion Sudvarg, Chenfeng Zhao, Ye Htet, Meagan Konst, Thomas Lang, Nick Song, Roger D. Chamberlain, Jeremy Buhler, and James H. Buckley. 2024. HLS Taking Flight: Toward Using High-Level Synthesis Techniques in a Space-Borne Instrument. In *Proc. of 21st ACM International Conference on Computing Frontiers (CF)*. 115–125. https://doi.org/10.1145/3649153.3649209

[28] Steve Vestal. 2007. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *28th IEEE international real-time systems symposium (RTSS 2007)*. IEEE, 239–243.

[29] Jian Wang, Michael Pikridas, Steven R. Spielman, and Tamara Pinterich. 2017. A fast integrated mobility spectrometer for rapid measurement of sub-micrometer aerosol size distribution, Part I: Design and model evaluation. *Journal of Aerosol Science* 108 (2017), 44–55. https://doi.org/10.1016/j.jaerosci.2017.02.012

[30] Tianyu Zhang, Nan Guan, Qingxu Deng, and Wang Yi. 2014. 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)*. IEEE, 179–188.