A Sensitivity Analysis for Mixed Criticality: Trading Criticality with Computational Resource

Luca Santinelli, Zhishan Guo

A Sensitivity Analysis for Mixed Criticality: Trading Criticality with Computational Resource

Luca Santinelli[1] and Zhishan Guo[2]
[1] DTIS - ONERA Toulouse France, luca.santinelli@onera.fr
[2] Missouri University of Science and Technology Rolla MO, guozh@mst.edu

***Abstract*—Mixing workloads with multiple criticality levels raises challenges both in timing analysis and schedulability analysis. The timing models have to characterize the different behaviors that real-time tasks can experience under the various criticality modes. Instead, the schedulability analysis has to combine every task and task interactions providing several guarantees, depending on the criticality level demanded at runtime. With this work, at first we propose representations to model every possible system criticality mode as a combination of task criticality modes. A set of bounding functions is obtained, a bound for each mode combination thus corresponding to a system criticality level. Secondly, we develop the schedulability analysis that applies such sets and derives schedulability conditions with mixed criticalities. The tasks are scheduled with fixed priority and earlies deadline first, and various levels of schedulability are defined from the mode combinations. Finally, we make use of the sensitivity analysis to evaluate the impact that multi mode task behaviors have on schedulability. Trade-offs between schedulability, criticality levels and resource availability are explored. A mixed critical real-time system case study validates the framework proposed.**

I. Introduction

An increasingly important trend in developing real-time systems is the integration of applications with different levels of criticality. The criticality designs the level of assurance needed for a system element against failure e.g., standards ISO 26262, DO 178C, and IEC 61508.

A Mixed Criticality (MC) real-time system is one that has two or more distinct criticality levels e.g., safety critical, mission critical, and/or low-critical. Such systems are defined to execute in a number of criticality modes, each mode specifying execution conditions and system criticalities. All the possible modes have to be characterized and analyzed in order to guarantee the predictability of the system. Please refer to [15] for a good overview of the MC problems for real-time systems.

Safety critical applications have to account for the worst-case behaviors that can possibly happen. The 'best' modeling of task parameters has to assure the coverage of any of the execution conditions, including the worst-cases [30], [18]. Mission critical or low critical applications rely on less constrained/demanding models and the guarantees on them are not as strict as those for safety critical applications [30], [18].

With respect to schedulability, the MC problem consists of multiple correctness criteria: timing constraints of safety critical tasks are guaranteed first, and then less critical tasks are eventually accommodated in the scheduling. Todays research on MC schedulability relies on mode changes in order to provide different assurance levels to the possible execution conditions [19]. Mode changes can be triggered by execution length, processor speed [5], [22], [21], [4], task release patterns [3], and the combination of those [23]. The resource is utilized in the manner such that all tasks are allowed to execute under low-critical modes in a more fairly manner, while priorities will be given to more critical tasks in a dedicated manner upon a mode switch. Such mode based correctness definition is welcomed by the industry, yet providing many research challenges [2].

We believe that some of the challenges to MC modeling and MC schedulability analysis can be addressed with *sensitivity analysis*. The sensitivity analysis applies to task models, and it studies the impact that task parameters have on system schedulability [27], [12]. The goal of this work is to effectively make use of sensitivity analysis for MC problems in order to study the costs for guaranteeing certain criticality levels at runtime.

Contribution: With this paper, we apply sensitivity analysis to models and schedulability analysis with MC. The MC task modeling is laid out with multiple bounding functions such as resource bound functions and workloads. Those functions are defined in order to bound task behaviors under the possible execution modes that can happen at runtime. Each bound represents a criticality level as well as an execution mode for the task set. The set of bounding functions is applied to develop the schedulability conditions with MC. Different levels of schedulability are defined from the possible criticality levels for fixed priority and earliest deadline first scheduling. Finally, the sensitivity analysis applies to evaluate the impact that MC task behaviors have on schedulability. Trade-offs between schedulability, criticality levels and resource availability are explored.

Organization of the paper: Section II presents real-time models that make use of bounding functions to characterize both resource requests and resource provisioning. Section III illustrates how we instantiate real-time modeling into MC modeling. Mode combinations are defined according to the scheduling policies applied and result into multiple possible bounds for the whole application. Section IV provides detailed notion of scheduling with MC, and the sensitivity analysis we develop to study the effects of criticality levels on schedulability and resource usage. Section V validates the modeling and analysis framework we propose with a test case of a

realistic real-time application. Section VI concludes the work and points out future research directions.

A. Related work

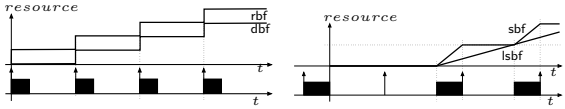
MC systems are typically defined to execute in a number of criticality modes. According to Vestal's definition [30], mode switch can be defined as follows: if any task attempts to execute for a longer time or more frequently like in case of faults, then a mode change occurs imposing high-criticality behavior to the tasks and the system [7], [16]. Under classic MC model, *all* low-critical tasks could be dropped from the system upon a mode switch, which may be a result of one single high-criticality task overrunning for 1 ms, or a 1 ms speed drop of one of the many processors. Obviously huge pessimism is involved under such modeling, even with the recent developments in providing multiple assurance levels to the possible execution conditions [19], [10], [8], [9]. We hereby apply real-time calculus basics [29] to derive bounds to resource request and resource provisioning in the interval domain. Those models have to be adapted to the MC problem for multiple bounding curves and different execution conditions. The resource usage can be further improved allowing MC application sharing computation without jeopardizing high criticality tasks.

The industry perspective to the MC problem focuses more on partitioning and separating applications by their criticality level. Safety critical applications would be timely and/or spacey separated from mission critical applications, [18]. Both models and schedulability are guaranteed within the partition which allow for compositional approaches especially handy with application qualification. Our work aims at studying all mode configurations which are possible at runtime with and without partitioning. We do not propose an alternative MC scheduling algorithm. It seems to us that this could be a way to close the gap between academic and industrial perspective to MC problems.

Traditional sensitivity analysis applies to real-time systems in order to study the impact that task parameters have on schedulability. It translates into abstract representations such as the (α, Δ)-space [27], and the C-space [12], [25], [20] where to map parameter values into schedulability conditions. Effective sensitivity analysis has yet to be built for MC problems. We hereby focus on the (α, Δ)-space to model schedulability conditions with MC and to which apply sensitivity analysis. In particular, the sensitivity analysis is hereby used for evaluating the different possible mode configurations, and to quantify the cost of changing the computational resource or schedulability conditions.

II. Computational models

A real-time task τ_i consists of a sequence of recurring jobs, each to be executed before a given deadline. In the periodic case, it is $\tau_i = (T_i, D_i, C_i)$, with T_i as the period, D_i as the deadline (it is assumed $D_i \leq T_i$), and C_i as the worst-case execution time (WCET). Tasks are grouped into task sets $\Gamma = \{\tau_1, \ldots, \tau_n\}$, equivalently real-time applications.



(a) Task τ_i execution with rbf and dbf

(b) Periodic server sbf with its linear approximation lsbf

Fig. 1: Resource, demand and supply bound functions.

Any executing task can be seen as a trace of events [13] with the cumulative function $R(t)$ to define the amount of computation resource requested by the task within $[0, t]$. For τ_i activated at time $t = 0$, its resource bound function (rbf) $\mathsf{rbf}_i(t)$ in any interval $[0, t]$ is: $\mathsf{rbf}_i(t) \stackrel{def}{=} \max\left\{0, \left(\left\lceil \frac{t}{T_i} \right\rceil \cdot C_i\right)\right\}$. $\mathsf{rbf}_i(t)$ upper bounds any resource request R_i such that $R_i(t) \leq \mathsf{rbf}_i(t)$ for all t. An example of rbf is represented in Figure 1(a) where the rbf is linked to task arrivals and immediate executions.

The computational resource is provided by reservation mechanisms, also known as servers. Although with different peculiarities, a lot of servers can be modeled as periodic servers which guarantee to provide Q (server capacity) units of time/resource in each period P (server period) [1]. With $S(t)$ the amount of resource made available up to time t by server S, the resource provisioning can be lower bounded in $[0, t]$ with the supply bound function sbf. $\mathsf{sbf}_S(t)$ is the minimum amount of time (computational resource) provided by S in any interval $[0, t]$ of length $t \geq 0$: $\mathsf{sbf}_S(t) \stackrel{def}{=} min_{t_0 \geq t} \int_{t_0}^{t_0+t} S(x)dx$ [26], [28]. The bounded-delay function lsbf is the linear approximation that lower bounds sbf in $[0, t]$, $\forall t \quad \mathsf{lsbf}_S(t) \leq \mathsf{sbf}_S(t)$; $\mathsf{lsbf}(t) \stackrel{def}{=} \max\{0, \ \alpha(\Delta)\}$. $\alpha \stackrel{def}{=} \lim_{t\to\infty} \frac{\mathsf{sbf}(t)}{t}$ is the resource provisioning rate, and $\Delta \stackrel{def}{=} \inf\{q \mid \alpha(t-q) \leq \mathsf{sbf}(t) \ \forall t\}$ is the longest interval with no resource provisioning [26], [28]. Figure 1(b) illustrates that and compares sbf with its linear approximation lsbf.

From lsbf, it is possible to define an (α, Δ)-space where to represent resource provisioning as well as resource requests [27]. An application Γ can be mapped into the (α, Δ)-space with its feasibility region Φ_Γ. Φ_Γ depends on the scheduling policy applied, and collects all the service supply pairs (α, Δ) that guarantee the schedulability of Γ.

A. Real-time schedule

Two common preemptive scheduling policies are the Fixed Priority (FP) and the Earliest Deadline First (EDF) [11], [6]. In this work we apply both.

The FP schedulability is guaranteed if each task in Γ, with static priority ordering, has enough resource to execute within its deadline. A task set Γ executing within a server S can be guaranteed under FP iff: $\forall\ i\ \exists\ t\ \in SchedP\ :\ \mathsf{wbf}_i(t) \leq \mathsf{sbf}_S(t)$. The tasks are ordered by priority, from higher to lower priority, where $hp(i) = \{\tau_1, \tau_2, \ldots, \tau_i\}$ denotes the sub-set of all tasks with a priority higher than or equal to τ_i; $SchedP$ defines the set of time instances where FP schedulability has to be verified [11], [14]. The level-i workload wbf_i is the

resource request from τ_i and it includes all the contributions of all higher priority tasks than τ_i: $\mathsf{wbf}_i(t) \stackrel{\text{def}}{=} \sum_1^{hp(i)} \mathsf{rbf}_k$.

The feasibility region Φ_Γ in the (α, Δ)-space for Γ is defined from the FP schedulability condition and the definition of lsbf. Thus, $\forall i\ \exists t \in SchedP_i \ : \ \mathsf{wbf}_i(t) \leq \alpha(t-\Delta)$ means that $\forall i$ it exists a t such that $\Delta \leq t - \frac{\mathsf{wbf}_i(t)}{\alpha}$. For all i, $\Delta \leq \max_{t \in SchedP_i} \left\{ t - \frac{\mathsf{wbf}_i(t)}{\alpha} \right\}$, and $\Delta \leq \min_i \max_{t \in SchedP_i} \left\{ t - \frac{\mathsf{wbf}_i(t)}{\alpha} \right\}$.

The EDF schedulability is guaranteed if the computational resource that Γ requires to execute is less than or equal to the available computational resource. A task set Γ within a server S can be guaranteed iff: $\forall t\ \mathsf{dbf}_\Gamma(t) \leq \mathsf{sbf}_S(t)$. In particular, the set of time instances where to check EDF schedulability can be reduced to the set D of deadlines within the task set hyperperiod [6]. The demand bound function dbf_i of τ_i is the resource requested by τ_i to fully execute by its deadline: $\mathsf{dbf}_i(t) \stackrel{\text{def}}{=} \mathsf{rbf}_i(t - D_i)$. It is the minimum possible resource request in order to execute the task by its deadline. dbf_Γ is the resource demand of the whole task set Γ: $\mathsf{dbf}_\Gamma \stackrel{\text{def}}{=} \sum_\Gamma \mathsf{dbf}_i$.

The feasibility region Φ_Γ for Γ is defined from the EDF schedulability condition and lsbf. $\forall\ t \in \mathrm{D} \ : \ \mathsf{dbf}(t) \leq \alpha \cdot (t - \Delta)$ means that $\forall\ t \in \mathrm{D} \ : \ \Delta \leq t - \frac{\mathsf{dbf}(t)}{\alpha}$, and $\Delta \leq \min_{t \in \mathrm{D}} \left\{ t - \frac{\mathsf{dbf}(t)}{\alpha} \right\}$.

III. Bounding with mixed criticality

In MC real-time systems, task parameters such as WCETs depend on criticality levels. Safety critical applications have to be assured against any possible execution condition, faults included. A way to do that is to consider WCETs large enough to account for such conditions. Instead, if the task is mission critical or non-critical, its WCET requirement would be smaller as the task demands less in terms of resource and assurance [30], [24], [17]. We restrict our modeling and analysis to two criticality levels: the high criticality HI and the low criticality LO. Nonetheless, our reasoning can generalize to any criticality level.

HI-criticality tasks, i.e. safety critical, can have two execution modes, HI-criticality mode represented with $C_i(\text{HI})$, and LO-criticality mode represented with $C_i(\text{LO})$. $C_i(\text{HI})$ models the HI-criticality behavior of the task τ_i (most critical), thus the worst possible conditions it can suffer [24]. $C_i(\text{LO})$ models the LO-criticality conditions for τ_i. It does not assure against faults, at least it does not against all of them [24] – worst-cases are not included. It has to be $C_i(\text{HI}) \geq C_i(\text{LO})$ [30].

The model of a HI-criticality task is:

$$\tau_i = ([C_i(\text{LO}), C_i(\text{HI})], T_i, D_i, \chi_i). \tag{1}$$

For it, there exist two resource request rbf, depending on the criticality mode active: $\mathsf{rbf}^{\text{HI}}_{\text{HI},i} = \max\left\{0, \left(\left\lceil \frac{t}{T_i} \right\rceil \cdot C_i(\text{HI})\right)\right\}$ which models the resource request in HI-criticality mode, and $\mathsf{rbf}^{\text{LO}}_{\text{HI},i} = \max\left\{0, \left(\left\lceil \frac{t}{T_i} \right\rceil \cdot C_i(\text{LO})\right)\right\}$ which models the resource request in LO-criticality mode. χ_i indicates the the task actual (at runtime) criticality mode, $\chi_i \in \{\text{HI}, \text{LO}\}$.

LO-criticality tasks are tasks that can only execute under LO-criticality mode. $C(\text{LO})_i$ is sufficient to model the task behavior. The LO-criticality task model is:

$$\tau_i = (C_i(\text{LO}), T_i, D_i), \tag{2}$$

with only the LO-criticality mode possible, and $\mathsf{rbf}_{\text{LO},i} = \max\left\{0, \left(\left\lceil \frac{t}{T_i} \right\rceil \cdot C(\text{LO})_i\right)\right\}$ models the resource request of the LO-criticality task τ_i.

The MC real-time application Γ composes of a HI-criticality part Γ_{HI} which includes all and only the HI-criticality tasks, and a LO-criticality part Γ_{LO} which includes all and only the LO-criticality tasks; $\Gamma = \Gamma_{\text{HI}} \cup \Gamma_{\text{LO}}$.

At runtime, there exist different possible combinations of tasks executing in their criticality modes. For example, there could exist combinations of only HI-criticality tasks executing in HI-criticality mode. There could also exist combinations where some HI-criticality tasks execute in HI-criticality mode, others executes in LO-criticality mode, and LO-criticality tasks executes as well. It is also possible to have all the HI-criticality task executing in LO-criticality mode together with some or all LO-criticality tasks. Each combination k is a scheduling that can happen at runtime, and has a criticality level associated χ^k resulting from the combination of the criticality mode applied in the scheduling/combination.

The system criticality level χ describes the combination of tasks and task modes at runtime, $\chi \in \{\chi^1, \chi^2, \ldots\}$. The purpose of this work is to model all the possible combinations, and apply them into schedulability analysis.

A. Bounding resource request

From the MC task modeling, Equation (1) and Equation (2), we define multiple bounds to the task set resource request. They represent execution conditions as combination of tasks and task modes that can happen at runtime.

i) $\mathsf{rbf}^{\text{HI}}_{\text{HI}}$ - is the resource request from **all and only** the HI-criticality tasks being in HI-criticality mode: $\mathsf{rbf}^{\text{HI}}_{\text{HI}} = \sum_{\forall \tau_i \in \Gamma_{\text{HI}}} \mathsf{rbf}^{\text{HI}}_{\text{HI},i}$;

ii) $\mathsf{rbf}^{\text{HI,LO},j}_{\text{HI}}$ - is the resource request from **all** the HI-criticality tasks in HI-criticality mode which are **combined** with LO-criticality tasks: $\mathsf{rbf}^{\text{HI,LO},j}_{\text{HI}} = \sum_{\forall \tau_i \in \Gamma_{\text{HI}}} \mathsf{rbf}^{\text{HI}}_{\text{HI},i} + \sum_{\text{some } \tau_k \in \Gamma_{\text{LO}}} \mathsf{rbf}_{\text{LO},k}$. $\overline{\mathsf{rbf}}^{\text{HI,LO}}_{\text{HI}} = \{\mathsf{rbf}^{\text{HI,LO},j}_{\text{HI}}\}$ is the set of those rbfs with index j specifying which LO-criticality tasks are added to the HI-criticality ones;

iii) $\mathsf{rbf}^{\text{HI-LO},r}_{\text{HI}}$ - is the resource request where **some** HI-criticality tasks in HI-criticality mode, the rest in LO-criticality mode, in **combination** with LO-criticality tasks: $\mathsf{rbf}^{\text{HI,LO},r}_{\text{HI}} = \sum_{\text{some } \tau_i \in \Gamma_{\text{HI}}} \mathsf{rbf}^{\text{HI}}_{\text{HI},i} + \sum_{\text{rest } \tau_j \in \Gamma_{\text{HI}}} \mathsf{rbf}^{\text{LO}}_{\text{HI},j} + \sum_{\text{some } \tau_k \in \Gamma_{\text{LO}}} \mathsf{rbf}_{\text{LO},k}$ The whole set of combinations is $\overline{\mathsf{rbf}}^{\text{HI-LO}}_{\text{HI}} = \{\mathsf{rbf}^{\text{HI-LO},r}_{\text{HI}}\}$ with r the index to the combinations specifying which LO-criticality tasks are applied and which HI-criticality tasks are in HI-criticality mode;

iv) $\mathsf{rbf}^{\text{LO},k}_{\text{HI}}$ - is the resource request where **all** the HI-criticality tasks are in LO-criticality mode and the **combination** with LO-criticality tasks: $\mathsf{rbf}^{\text{LO},k}_{\text{HI}} = \sum_{\forall \tau_i \in \Gamma_{\text{HI}}} \mathsf{rbf}^{\text{HI}}_{\text{HI},i} + \sum_{\text{some } \tau_k \in \Gamma_{\text{LO}}} \mathsf{rbf}_{\text{LO},k}$. $\overline{\mathsf{rbf}}^{\text{LO}}_{\text{HI}} = \{\mathsf{rbf}^{\text{LO},k}_{\text{HI}}\}$ is the set of such

combinations, with k the index to represent which LO-criticality tasks are added;

$\mathsf{rbf}_{\mathrm{LO}}$ - is the resource request from **only** LO-criticality tasks: $\mathsf{rbf}_{\mathrm{LO}} = \sum_{\forall \tau_i \in \Gamma_{\mathrm{LO}}} \mathsf{rbf}_{\mathrm{LO},i}$.

The resource requests of all the combinations between tasks and task modes can be grouped as:

$$\overline{\mathsf{rbf}} \stackrel{def}{=} \{\mathsf{rbf}_{\mathrm{HI}}^{\mathrm{HI}}, \overline{\mathsf{rbf}}_{\mathrm{HI}}^{\mathrm{HI,LO}}, \overline{\mathsf{rbf}}_{\mathrm{HI}}^{\mathrm{HI-LO}}, \overline{\mathsf{rbf}}_{\mathrm{HI}}^{\mathrm{LO}}, \mathsf{rbf}_{\mathrm{LO}}\}. \quad (3)$$

To each resource request there is a system criticality level χ^k associated, $\chi^k \in \overline{\chi}$.

B. MC Bounding for FP and EDF

With the MC model there exist a set of level-i workload bounds, each obtained with the combination of HI- and LO-criticality tasks in their respective modes. Only the tasks higher priority than τ_i are combined for the level-i workload, and χ_i^k is the criticality level associated of the level-i workloads combination. There exist: i) $\mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{HI}}$ - **only** the HI-criticality tasks all in HI-criticality mode. To it, there is $\chi_{\mathrm{HI},i}^{\mathrm{HI}} = \mathrm{HI}$ representing its criticality level; ii) $\overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{HI,LO}}$ - the HI-criticality tasks **all** in HI-criticality mode **combined** with LO-criticality tasks. To each $\mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{HI,LO},k} \in \overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{HI,LO}}$, $\chi_{\mathrm{HI},i}^{\mathrm{HI,LO},k}$ represents its criticality level; iii) $\overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{HI-LO}}$ - **some** of the HI-criticality tasks in HI-criticality mode (the rest is in LO-criticality mode) **combined** with LO-criticality tasks. To each $\mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{HI-LO},j} \in \overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{HI-LO}}$, $\chi_{\mathrm{HI},i}^{\mathrm{HI-LO},j}$ represents its criticality level; iv) $\overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{LO}}$ - **all** the HI-criticality tasks in LO-criticality mode **combined** with LO-criticality modes. To each $\mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{LO},r} \in \overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{LO}}$, $\chi_{\mathrm{HI},i}^{\mathrm{LO},r}$ represents its criticality level; v) $\mathsf{wbf}_{\mathrm{LO},i}$ - **only** LO-criticality tasks. To it, there is $\chi_{\mathrm{LO},i} = \mathrm{LO}$ representing its criticality level.

All the combination of the level-i workload are grouped as:

$$\overline{\mathsf{wbf}}_{\mathrm{HI},i} \stackrel{def}{=} \{\mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{HI}}, \overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{HI,LO}}, \overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{HI-LO}}, \overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{LO}}, \mathsf{wbf}_{\mathrm{LO},i}\}, \quad (4)$$

with the criticality levels for the level-i as:

$$\overline{\chi}_i \stackrel{def}{=} \{\chi_{\mathrm{HI},i}^{\mathrm{HI}}, \overline{\chi}_{\mathrm{HI},i}^{\mathrm{HI,LO}}, \overline{\chi}_{\mathrm{HI},i}^{\mathrm{HI-LO}}, \overline{\chi}_{\mathrm{HI},i}^{\mathrm{LO}}, \chi_{\mathrm{LO},i}\}. \quad (5)$$

The bounds in Equation (4) can be ordered in increasing order, $\mathsf{wbf}^j \leq \mathsf{wbf}^{j+1}$; the set $\overline{\chi}_i$ from Equation (5) is ordered accordingly and such that χ_i^j is for wbf_i^j and χ_i^{j+1} is for wbf_i^{j+1}.

In case of $\mathrm{HI}-\mathrm{LO}$, instead of enlisting all the combinations it is possible to define 'envelope' bounds depending on the number of HI-criticality tasks that are in HI-criticality mode at the same time: $\mathsf{wbf}_{\mathrm{HI},i}^{*\mathrm{HI-LO},k} \stackrel{def}{=} \max_j\{\mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{HI-LO},j}\}$. k is the number of HI-criticality tasks in HI-criticality mode considered for the combination. $\overline{\mathsf{wbf}}_{\mathrm{HI},i}^{*\mathrm{HI-LO}}$ collects all those envelopes, for all k, and can be applied into $\overline{\mathsf{wbf}}$ instead of $\overline{\mathsf{wbf}}_{\mathrm{HI},i}^{\mathrm{HI-LO}}$. This would reduce the number of possible combinations and criticality levels, in turn reducing the complexity of the modeling.

Under EDF, the different combinations are represented with dbfs. There exist: i) $\mathsf{dbf}_{\mathrm{HI}}^{\mathrm{HI}}$ - **only** the HI-criticality tasks all in HI-criticality mode; $\chi_{\mathrm{HI},i}^{\mathrm{HI}} = \mathrm{HI}$ represents its criticality level; ii) $\overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{HI,LO}}$ - the HI-criticality tasks **all** in HI-criticality mode **combined** with LO-criticality tasks. To each $\mathsf{dbf}_{\mathrm{HI}}^{\mathrm{HI,LO},k} \in \overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{HI,LO}}$, $\chi_{\mathrm{HI}}^{\mathrm{HI,LO},k}$ associated to, is its criticality level; iii) $\overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{HI-LO}}$ - **some** of the HI-criticality tasks in HI-criticality mode (the rest is in LO-criticality mode) **combined** with LO-criticality tasks. To each $\mathsf{dbf}_{\mathrm{HI}}^{\mathrm{HI-LO},j} \in \overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{HI-LO}}$, $\chi_{\mathrm{HI}}^{\mathrm{HI-LO},j}$ is its criticality level; iv) $\overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{LO}}$ - **all** the HI-criticality tasks in LO-criticality mode **combined** with LO-criticality modes. To each $\mathsf{dbf}_{\mathrm{HI}}^{\mathrm{LO},r} \in \overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{LO}}$, $\chi_{\mathrm{HI}}^{\mathrm{LO},r}$ is its criticality level; v) $\mathsf{dbf}_{\mathrm{LO}}$ - **only** LO-criticality tasks; $\chi_{\mathrm{LO},i} = \mathrm{LO}$ representing its criticality level. The set of all those dbfs is:

$$\overline{\mathsf{dbf}}_{\mathrm{HI}} \stackrel{def}{=} \{\mathsf{dbf}_{\mathrm{HI}}^{\mathrm{HI}}, \overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{HI}}, \overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{HI-LO}}, \overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{LO}}, \mathsf{dbf}_{\mathrm{LO}}\}, \quad (6)$$

and the set of criticality levels is:

$$\overline{\chi} \stackrel{def}{=} \{\chi_{\mathrm{HI}}^{\mathrm{HI}}, \overline{\chi}_{\mathrm{HI}}^{\mathrm{HI,LO}}, \overline{\chi}_{\mathrm{HI}}^{\mathrm{HI-LO}}, \overline{\chi}_{\mathrm{HI}}^{\mathrm{LO}}, \chi_{\mathrm{LO}}\}. \quad (7)$$

The bounds in Equation (6) can be ordered in increasing order, $\mathsf{dbf}^i \leq \mathsf{dbf}^{i+1}$; the set $\overline{\chi}$ from Equation (7) is ordered accordingly and such that χ^j is for dbf^j and χ^{j+1} is for dbf^{j+1}.

Within the $\mathrm{HI}-\mathrm{LO}$ case, there can be defined bounds such that: $\mathsf{dbf}^{*\mathrm{HI-LO},k} \stackrel{def}{=} \max_j\{\mathsf{dbf}^{\mathrm{HI-LO},j}\}$. k the number of HI-criticality tasks in HI-criticality mode; $\overline{\mathsf{dbf}}_{\mathrm{HI}}^{*\mathrm{HI-LO}}$ collects them all and can be applied into $\overline{\mathsf{dbf}}$ instead of $\overline{\mathsf{dbf}}_{\mathrm{HI}}^{\mathrm{HI-LO}}$. This allows reducing the number of possible combinations and criticality levels, in turn reducing the complexity of the modeling.

Figure 2 illustrates an example of some level-i bounds $\mathsf{wbf}_i \in \overline{\mathsf{wbf}}$, while Figure 3 is an example of some demand bounds $\mathsf{dbf} \in \overline{\mathsf{dbf}}$ from different task mode combinations. For each bound there is associated a criticality level. In the figures there are represented few bounds (3 LO and a HI) which can be compared among them and with the available resource sbf. It is: $\mathsf{wbf}_{\mathrm{LO},i}^{\mathrm{HI},r} \leq \mathsf{wbf}_{\mathrm{LO},i}^{\mathrm{HI},t} \leq \mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{HI},k} \leq \mathsf{wbf}_{\mathrm{HI},i}^{\mathrm{LO},s}$ and $\mathsf{dbf}_{\mathrm{LO}}^{\mathrm{HI},m} \leq \mathsf{dbf}_{\mathrm{LO}}^{\mathrm{HI},n} \leq \mathsf{dbf}_{\mathrm{HI}}^{\mathrm{HI},j} \leq \mathsf{dbf}_{\mathrm{HI}}^{\mathrm{LO},v}$.

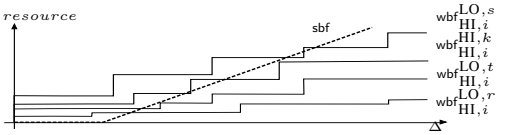


Fig. 2: level-i wbfs from MC executions under FP. Some wbf_is compared with the resource provisioning sbf.

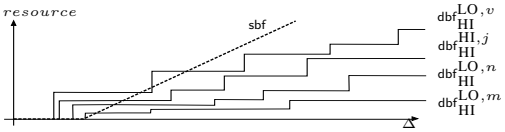


Fig. 3: dbfs from MC executions under EDF. Some dbfs compared with the resource provisioning sbf.

IV. Scheduling with mixed criticality

We propose two schedulability analyses based on FP and EDF that apply MC tasks Equation (1) and Equation (2). They are off-line analyses which account for all the criticality mode combinations that can happen at runtime. They embeds criticality levels into schedulability conditions.

These analyses focus on finding which are the criticality levels (criticality mode combinations) that can be assured schedulable. They also allow for evaluating the resource applied to execute Γ_{HI}, and thus the remaining resource is left to execute Γ_{LO} without harming HI-criticality tasks' executions.

A. FP and EDF scheduling with mixed criticality

To guarantee schedulability, the resource provisioning sbf is compared with resource requests (workloads) in case of FP, or with the resource demand in case for EDF. Figure 2 and Figure 3 illustrate the comparison between bounds and some available sbf. The way to compare depends on the scheduling policy.

Theorem 1 (FP schedulability with MC): Considering a mixed criticality task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ of n tasks ordered with decreasing priority, i.e., τ_1 is assigned the highest priority whereas τ_n is assigned the lowest priority. Γ composes of HI-criticality tasks Γ_{HI} defined as in Equation (1), and LO-criticality tasks Γ_{LO} defined as in Equation (2). $\forall \tau_i \in \Gamma$, $hp(i) = \{\tau_j, \tau_k, \ldots, \tau_i\}$ is the set of tasks with priority higher or equal to τ_i; tasks in $hp(i)$ belongs to Γ_{HI} and Γ_{LO}. The level-i workloads are: $\overline{\mathsf{wbf}}_i = \{\mathsf{wbf}^{\text{HI}}_{\text{HI},i}, \overline{\mathsf{wbf}}^{\text{HI,LO}}_{\text{HI},i}, \overline{\mathsf{wbf}}^{\text{HI-LO}}_{\text{HI},i}, \overline{\mathsf{wbf}}^{\text{LO}}_{\text{HI},i}, \mathsf{wbf}_{\text{LO},i}\}$, according to Equation (4), and $\overline{\chi}_i$ defines the set of criticality levels for the level-i workloads Equation (5). Γ is FP schedulable under resource provisioning sbf with system criticality level $\chi = min_i\{\chi_i\}$, χ_i being the level-i criticality level in $\overline{\chi}_i$, if for all $i \in \{1, 2, \ldots, n\}$ $\exists t_0 \in \text{schedP}_i$ such that:

$$\mathsf{wbf}^{\chi_i}_i(t_0) \leq \mathsf{sbf}(t_0); \tag{8}$$

schedP_i is the set of deadlines of all $\tau_j \in hp(i)$.

Proof: The schedulability of each task τ_i in Γ is guaranteed with the largest level-i workload which is smaller than sbf [11]; χ_i corresponding to the largest schedulable level-i, is the schedulability criticality level for τ_i. The task set is schedulable if all tasks are schedulable, and the criticality level is the minimum among the schedulable criticality levels that satisfy all the conditions, $\chi = min_i\{\chi_i\}$. ∎

Equation (8) in Theorem 1 defines the FP schedulability conditions which apply MC models. It proposes different degree of schedulability for MC tasks.

Theorem 2 (EDF schedulability with MC): Considering a mixed criticality task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ with HI-criticality tasks Γ_{HI} defined as in Equation (1) and LO-criticality tasks Γ_{LO} defined as in Equation (2), $\Gamma = \Gamma_{\text{HI}} \cup \Gamma_{\text{LO}}$. For Γ, the ordered demand bound functions are $\overline{\mathsf{dbf}} = \{\mathsf{dbf}^{\text{HI}}_{\text{HI}}, \overline{\mathsf{dbf}}^{\text{HI,LO}}_{\text{HI}}, \overline{\mathsf{dbf}}^{\text{HI-LO}}_{\text{HI}}, \overline{\mathsf{dbf}}^{\text{LO}}_{\text{HI}}, \mathsf{dbf}_{\text{LO}}\}$, Equation (6), with $\overline{\chi}$ defining the ordered set of levels of criticality Equation (7). Γ is EDF schedulable under resource provisioning sbf with system criticality level χ if $\forall t_0 \in D$:

$$\mathsf{dbf}^{\chi}(t_0) \leq \mathsf{sbf}(t_0); \tag{9}$$

$\chi \in \overline{\chi}$ and $\mathsf{dbf}^{\chi} \in \overline{\mathsf{dbf}}_{\text{HI}}$.

Proof: For Γ, with HI-criticality tasks combined with LO-criticality tasks, the largest demand bound function $\mathsf{dbf}^{\chi}_{\text{HI}} \in \overline{\mathsf{dbf}}$ which is smaller than sbf assures schedulability for the tasks combination that it represents, [6]. χ describes the criticality level of the application up to which, EDF schedulability is guaranteed. ∎

Equation (9) in Theorem 2 defines EDF schedulability conditions which apply the MC models. It proposes different degree of schedulability for MC tasks.

B. Feasibility regions with mixed criticality

The FP scheduling condition parametrized with χ_i, Equation (8), translates into comparing feasibility regions and points within the (α, Δ)-space. A feasibility region Φ^{χ_i} is defined such that: $\Delta \leq \min_i \max_{t \in SchedP_i} \left\{ t - \frac{\mathsf{wbf}^{\chi_j}_i(t)}{\alpha} \right\}$. It has associated the criticality level χ_i such that all the wbf_i, for all i applied, are from the same criticality level χ_i, Theorem 1.

For EDF it is the same with the scheduling condition in Equation (9). A feasibility region Φ^{χ} is defined such that: $\Delta \leq \min_{t \in \text{D}} \left\{ t - \frac{\mathsf{dbf}^{\chi}(t)}{\alpha} \right\}$, and is parametrized with χ.

To both FP and EDF, there exist a set $\overline{\Phi}$ of feasibility regions for the possible criticality levels resulting from the mode combinations – one set per scheduling policy. It is:

$$\overline{\Phi} \stackrel{def}{=} \{\Phi^{\text{HI}}_{\text{HI}}, \overline{\Phi}^{\text{HI,LO}}_{\text{HI}}, \overline{\Phi}^{\text{HI-LO}}_{\text{HI}}, \overline{\Phi}^{\text{LO}}_{\text{HI}}, \Phi_{\text{LO}}\}, \tag{10}$$

with the criticality level associated:

$$\overline{\chi} \stackrel{def}{=} \{\chi^{\text{HI}}_{\text{HI}}, \overline{\chi}^{\text{HI,LO}}_{\text{HI}}, \overline{\chi}^{\text{HI-LO}}_{\text{HI}}, \overline{\chi}^{\text{LO}}_{\text{HI}}, \chi_{\text{LO}}\}. \tag{11}$$

$\overline{\Phi}$ can be made of envelope bounds, with $\overline{\Phi}^{*\text{HI-LO}}_{\text{HI}}$ instead of $\overline{\Phi}^{\text{HI-LO}}_{\text{HI}}$. The feasibility regions in Equation (10) can be ordered in increasing order (from the small region to the larger), with the consequent ordering of the criticality levels in Equation (11).

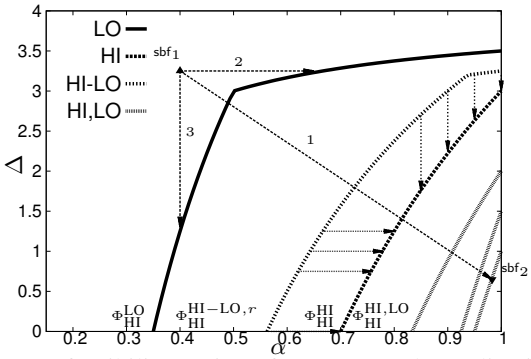


Fig. 4: FP feasibility regions for an example application Γ.

Figure 4 details some feasibility regions for FP scheduling with mixed criticality. Here it is possible comparing regions between them (*ordering between regions* $\Phi^k \leq \Phi^j$), and compare each region with the available resource sbf (sbf *is inside* Φ^j *thus* χ^j *is guaranteed*). LO-criticality conditions (LO) are more prone to schedulability since they require less computational resource – larger feasibility region. The more HI-criticality tasks are scheduled in HI-criticality mode or the more LO-criticality tasks are scheduled together with HI-criticality tasks, the larger is the resource required to schedule – smaller feasibility regions.

C. Sensitivity analysis with mixed criticality

We intend to use sensitivity analysis to investigate multiple elements which can impact the design of MC real-time systems. In particular, we apply sensitivity analysis with schedulability conditions parametrized with criticality levels, Theorem 1 and Theorem 2. Our proposal is illustrated with three questions.

Q1) *Which is the criticality level that can be assured with the available resource provisioning?* This is a critical question for MC scheduling as it focuses on how enhancing computational resource usage by scheduling both HI-criticality and LO-criticality tasks together. With the (α, Δ)-space representation, the sensitivity analysis answers Q1 finding the combinations that can be scheduled for a given resource. Considering the (k, m) formalization, k LO-criticality tasks out of m total task executing, Q1 becomes seeking how many LO-criticality tasks can be executed together with $m - k$ HI-criticality task in HI-criticality modes. k is the parameter to be studied in order to find the largest value that can be guaranteed with the available resource. With the MC modeling proposed in combination with the (α, Δ) representation, this can be solved seeking for the largest feasibility region that include the sbf given. It is exploring an index in $\overline{\Phi}$ seeking for regions.

Q2) *What is the cost to guarantee schedulable a certain criticality level?* The cost being in terms of computational resource. The sensitivity analysis can be used to define what is the resource change necessary to guarantee the schedulability up to a specific criticality level. This is very helpful in defining and evaluating trade-offs between resource and criticality/schedulability. The Euclidean distance $dist(\mathsf{sbf}_2, \mathsf{sbf}_1)$ between two points in the (α, Δ)-space, defined as:

$$dist(\mathsf{sbf}_2, \mathsf{sbf}_1) \stackrel{def}{=} (\delta\alpha = \alpha_2 - \alpha_1, \delta\Delta = \Delta_2 - \Delta_1), \quad (12)$$

quantifies the distance between two resource provisioning $\mathsf{sbf}_2 - \mathsf{sbf}_1 = (\alpha_2 - \alpha_1, \Delta_2 - \Delta_1)$. The cost here is the resource provisioning change necessary to move from sbf_1 to sbf_2. To note that in order to increase the resource provisioning, α has to increase and Δ has to decrease. There exist also the distance between a point and a feasibility region, $dist(\mathsf{sbf}_1, \Phi^j)$. We define it as:

$$dist(\Phi^j, \mathsf{sbf}_k) \stackrel{def}{=} (\pm_{\geq 0/<0} min|\delta\alpha|, \pm_{\geq 0/<0} min|\delta\Delta|). \quad (13)$$

Metric (13) quantifies the resource change to guarantee schedulable the configuration represented by Φ^j. With all positive δs, the sign of the minimum between the absolute values $|\cdot|$ is $+$; with all negative δs, the sign is $-$. αs and Δs for Φ^j are taken from the region border, and the min is for the δs between sbf^k and all those points.

Q3) *What is the cost to change a system criticality mode?* With this, we intend the possibility in the (α, Δ)-space to quantify the computational resource difference between two criticality levels. The sensitivity analysis quantifies that difference as distance between the two regions which is defined as:

$$dist(\Phi^k, \Phi^j) \stackrel{def}{=} (\pm_{\geq 0/<0} min|\delta\alpha|, \pm_{\geq 0/<0} min|\delta\Delta|). \quad (14)$$

Metric (14) is applied at iso-parameter, which means computing the $\delta\Delta$ with the same α, and $\delta\alpha$ with the same Δ. With all positive δs, the sign of the minimum between the absolute values $|\cdot|$ is $+$; with all negative δs, the sign is $-$; with both negative and positive δs, the the min is 0 as the intersection between the regions. The αs and the Δs are taken from the regions border. Metric (13) and Metric (14) are computed differently to signal the resource difference that exist between the two cases.

Figure 4 is an example of sensitivity analysis for evaluating the resource necessary to guarantee schedulability – Q1 and Q2 with Metric (12) and Metric (13). There are 6 regions grouped in 4 different classes: LO for only LO-criticality modes combined, HI for only HI-criticality modes combined, HI − LO for some HI-criticality tasks in HI-criticality modes combined with some LO-criticality tasks, HI, LO for all HI-criticality tasks in HI-criticality mode combined with LO-criticality tasks. There are three cases which define three resource provisioning changes from an initial resource sbf_1. With sbf_1 available is not possible guarantee any of the schedulability level represented, since sbf_1 in not included in any of those feasibility regions.

Change 1: change of sbf_1 to sbf_2 to guarantee schedulability of HI, LO, the most demanding case among the represented ones. $dist(\mathsf{sbf}_2, \mathsf{sbf}_1) = (\delta\alpha = 0.985 - 0.4 = 0.585, \delta\Delta = 0.65 - 3.25 = -2.6)$, Metric (12). Change 2 iso-Δ: change of sbf_1 to LO configuration schedulability by modifying only α Metric (13), $dist(\Phi_{\text{LO}}, \mathsf{sbf}_1) = (\delta\alpha = 0.65 - 0.4 = 0.25, \delta\Delta = 0)$. Change 3 iso-$\alpha$: change of sbf_1 to LO configuration schedulability by modifying only Δ Metric (13), $dist(\Phi_{\text{LO}}, \mathsf{sbf}_1) = (\delta\alpha = 0, \delta\Delta = 1.3 - 3.25 = -1.95)$. The difference between change 2 and change 3 is in terms of changing either α's or Δ's. Distances where one dimension is 0 are advantaged, since the resource change necessary is easier to apply – less constraints.

Figure 4 presents also an example of cost evaluation with sensitivity analysis, Q3 and Metric (14). From HI to one HI − LO configuration k, it is: $dist(\Phi^{\text{HI}}, \Phi^{\text{HI-LO},k}) = (0.17, -0.2)$. It quantifies the resource difference between Φ^{HI} and $\Phi^{\text{HI-LO},k}$, equivalently the resource increase necessary to schedule $\Phi^{\text{HI-LO},k}$ from a schedulable Φ^{HI}. This translates also into the resource necessary to add LO-criticality tasks into the scheduling.

V. Evaluation

The case study here is to apply our MC modeling, our schedulability analysis , and our sensitivity analysis.

The MC real-time application Γ considered is a robotic application which combines HI-criticality tasks and LO-criticality tasks. It composes of a total of 14 tasks, 10 HI-criticality tasks and 4 LO-criticality tasks, which implements the main functionalities that a robot could have e.g., drivers, slam, navigation, control. It is inspired by MAUVE project https://forge.onera.fr/projects/mauve, to which τ_9 and τ_{10} are added for two extra safety critical functionalities while LO-criticality tasks represents functionally important, but non-critical, jobs

such as image processing. Table I recaps it and the parameters for each task with the specificity of the MC task model considered, Equation (1) and Equation (2). Tasks are assumed with $T_i = D_i$. We note that Γ is made from harmonic tasks, thus the utilization criteria[1] could be applied for schedulability analysis [13]. Instead, we use Theorem 1 and Theorem 2 in combination with the (α, Δ)-space. The utilization is used only for consideration on how to partition Γ.

	T/Δ	C/α
τ_1 - "drivers" (HI-criticality)	50	$(5, 10)$
τ_2 - "control" (HI-criticality)	100	$(4, 8)$
τ_3 - "guidance" (HI-criticality)	100	$(1, 3)$
τ_4 - "laser" (HI-criticality)	200	$(5, 10)$
τ_5 - "tracking" (HI-criticality)	50	$(10, 15)$
τ_6 - "camera" (HI-criticality)	200	$(1, 3)$
τ_7 - "SLAM" (HI-criticality)	50	$(10, 20)$
τ_8 - "navigation" (HI-criticality)	100	$(4, 8)$
τ_9 - "crit1" (HI-criticality)	100	$(15, 25)$
τ_{10} - "crit2" (HI-criticality)	100	$(15, 20)$
τ_{11} - "no-crit1" (LO-criticality)	200	15
τ_{12} - "no-crit2" (LO-criticality)	200	25
τ_{13} - "no-crit3" (LO-criticality)	200	40
τ_{14} - "no-crit4" (LO-criticality)	200	20
$\mathsf{sbf}_1^1 = \mathsf{sbf}_2^1$	25	0.6
$\mathsf{sbf}_1^2 = \mathsf{sbf}_2^2$	12	0.75
$\mathsf{sbf}_1^3 = \mathsf{sbf}_2^3$	7	0.9
$\mathsf{sbf}_1^4 = \mathsf{sbf}_2^4$	0.3	0.99

TABLE I: Task sets and resource supply.

Resource partitioning. For Γ_{HI}, and the tasks in HI-criticality mode, it is: $U_{\text{HI}}^{\text{HI}} = \frac{333}{200}$. With the tasks in LO-criticality mode, it is: $U_{\text{HI}}^{\text{LO}} = \frac{165}{200}$. For Γ_{LO} it is $U_{\text{LO}} = \frac{100}{200}$. The worst-case total utilization is $U_{\text{HI}}^{\text{HI}} + U_{\text{LO}} = \frac{433}{200}$, while the best case total utilization is: $U_{\text{HI}}^{\text{LO}} + U_{\text{LO}} = \frac{265}{200}$. In order to guarantee the scheduling of some combinations, at least two resource partitions would be required, and for each have $U_j \leq 1$.

The industry approach to MC would consist of separating tasks by their criticality levels: HI-criticality tasks separated by LO-criticality tasks. This would end up into 3 different resource partitions, two for HI-criticality tasks, since one would not be enough to guarantee the schedulability being $U_{\text{HI}}^{\text{HI}} = \frac{333}{200}$, and one for LO-criticality tasks $U_{\text{LO}} = \frac{100}{200}$. This partitioning choice is not optimal in terms of resource usage since it has large waist of computational resource, approximately $\frac{167}{200} + \frac{100}{200}$.

Other partitioning solutions could be applied with equivalent guarantees on scheduling both HI- and LO-criticality tasks. We propose the following based on two partitions with almost evenly distributed utilizations. Partition P_1, is such that $P_1 = \{\tau_1, \tau_2, \tau_3, \tau_5, \tau_{10}, \tau_{11}, \tau_{13}\}$ with $U_{1,\text{HI}}^{\text{HI}} = \frac{162}{200}$, $U_{1,\text{HI}}^{\text{LO}} = \frac{100}{200}$, and $U_{1,\text{LO}} = \frac{55}{200}$. Partition P_2, $P_2 = \{\tau_4, \tau_6, \tau_7, \tau_8, \tau_9, \tau_{12}, \tau_{14}\}$, with $U_{2,\text{HI}}^{\text{HI}} = \frac{159}{200}$, $U_{2,\text{HI}}^{\text{LO}} = \frac{84}{200}$, and $U_{2,\text{LO}} = \frac{45}{200}$.

The tasks in P_1 are scheduled under EDF, while those in P_2 are scheduled under FP. The resource partitioning can be seen as partitioned multi-core scheduling. For each partition, we envision 4 possible resource provisioning: $\mathsf{sbf}_1^1 = \mathsf{sbf}_2^1 = (0.6, 25)$, $\mathsf{sbf}_1^2 = \mathsf{sbf}_2^2 = (0.75, 12)$, $\mathsf{sbf}_1^3 = \mathsf{sbf}_2^3 = 0.9, 7)$, and $\mathsf{sbf}_1^4 = \mathsf{sbf}_2^4 = (0.99, 0.3)$. In Table I the resource provisioning details are illustrated.

P_1 **sensitivity analysis.** For P_1 there are: i) 5 dbfs (and Φs) from HI, HI, LO, 1 (τ_{11}), HI, LO, 2 (τ_{13}), HI, LO, 3 ($\tau_{11} + \tau_{13}$), and LO; ii) 95 possible dbfs (and Φs) from HI − LO combinations. 15 are from only one HI-criticality task in HI-criticality mode, and can be reduced to three $\mathsf{dbf}^{*\text{HI}-\text{LO}-1}$ with the envelop bounding; 30 are from only two HI-criticality tasks in HI-criticality mode, and can be reduced to three bounds $\mathsf{dbf}^{*\text{HI}-\text{LO}-2}$; and so on. Figure 5 represents the set of feasibility regions for P_1 compared with the four possible resource available. The cases LO + LOs represents the bounds with all HI-criticality tasks in LO-criticality modes combined with LO-criticality tasks, while $only$LO is for only HI-criticality tasks in LO-criticality mode. The case with only LO-criticality tasks scheduled is not depicted.

With sbf_1^1 it is not possible to guarantee any of the criticality combinations for P_1; with sbf_1^2 it is possible to guarantee $only$LO and LO + LOs with only τ_{11} added. To note that it is not possible to guarantee schedulable the combination with all HI-criticality tasks in HI-criticality mode together with all LO-criticality tasks, since $U_{1,\text{HI}}^{\text{HI}} + U_{1,\text{LO}} > 1$. This is illustrated in the (α, Δ)-space representation with only two feasibility regions $\Phi_{\text{HI}}^{\text{HI,LO}}$ and not three.

The costs to change resource provisioning are $dist(\mathsf{sbf}_1^2, \mathsf{sbf}_1^1) = (0.15, -13)$, $dist(\mathsf{sbf}_1^3, \mathsf{sbf}_1^2) = (0.3, -18)$, $dist(\mathsf{sbf}_1^4, \mathsf{sbf}_1^3) = (0.39, -24.7)$, $dist(\mathsf{sbf}_1^3, \mathsf{sbf}_1^2 = (0.15, -5)$, $dist(\mathsf{sbf}_1^4, \mathsf{sbf}_1^2) = (0.24, -11.7)$, and $dist(\mathsf{sbf}_1^4, \mathsf{sbf}_1^3) = (0.09, -6.7)$. The sensitivity analysis quantifies all those costs with Metric (12). For example, while designing the system and deciding to change resource provisioning in order to guarantee HI, LO cases, sbf_1^3 would be necessary which make the need for and increase of 0.15 of α and decreasing of 5 of Δ from an initial sbf_1^1.

As another example of sensitivity analysis for Q2, the cost for including a second LO-criticality task to all HI-criticality tasks in LO-criticality mode (LO + LOs) would be: $dist(\Phi_{\text{HI}}^{\text{LO},2}, \Phi_{\text{HI}}^{\text{LO},1}) = (0.17, -11)$. Instead, including all three LO-criticality tasks to all HI-criticality tasks in LO-criticality mode (LO + LOs) it costs $dist(\Phi_{\text{HI}}^{\text{LO},3}, \Phi_{\text{HI}}^{\text{LO},1}) = (0.2, -11)$.

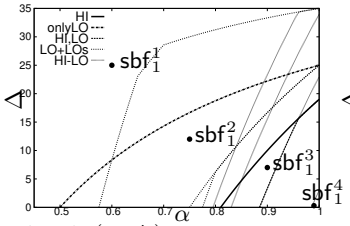


Fig. 5: (α, Δ)-space representation for P_1 under EDF.

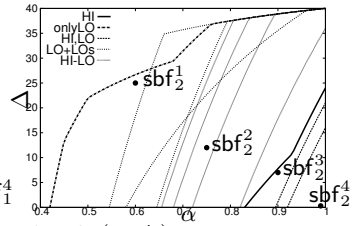


Fig. 6: (α, Δ)-space representation for P_2 under FP.

P_2 **sensitivity analysis.** Figure 6 illustrates the feasibility regions for the different task mode combinations in P_2, cases HI, HI, LO, HI − LO and LO. In particular, for HI − LO there are represented the regions from $\mathsf{wbf}_{\text{HI},i}^{*\text{HI}-\text{LO}}$. With sbf_1^2, it

[1]The task utilization U_i is the ration between the computation time and the period, $U_i = C_i/T_i$. The application utilization U is given as the summation between task utilizations, $U = \sum_i U_i$. The utilization criteria with harmonic tasks and deadlines equal to the periods, is such that schedulability is guaranteed iff $U \leq 1$ [13].

is possible to schedule the $only$LO case, all the LO + LOs cases, and some HI − LO cases. The schedulability of all the criticality levels can be achieved only with sbf_4. sbf_2^2 and sbf_3^2 allow the schedulability of intermediate combinations. To note that with sbf_4 there is also some resource margin for eventually including new tasks. That margin can be quantified with $dist(\mathsf{sbf}_2^4, \Phi_{\text{HI}}^{\text{HI}-\text{LO}}) = (-0.7, 16)$ as resource reduction. In P_2, an example of Metric (14) applied to evaluate the difference between scheduling HI − LO (four HI-criticality modes and all the LO-criticality tasks) and scheduling HI − LO (three HI-criticality modes and all the LO-criticality tasks) is $dist(\Phi_{\text{HI}}^{*\text{HI}-\text{LO},4}, \Phi_{\text{HI}}^{*\text{HI}-\text{LO},3}) = (0.1, -4)$. In order to schedule four HI-criticality modes from three HI-criticality, the resource has to be increased by $\delta\alpha = 0.1$ and $\delta\Delta = -4$.

VI. Conclusion

We have developed MC models with workloads and demand bound functions that bound criticality mode combinations and define multiple system criticality levels. The schedulability analyses we proposed make use of the MC models and apply them to FP and EDF. In there, the scheduling conditions are parametrized with the criticality levels: combinations are guaranteed to be schedulable or not, depending on the available computational resource sbf. We also formalized the (α, Δ)-space for the MC problem. In there, MC models and MC scheduling conditions translate into feasibility regions where criticality level is guaranteed to be schedulable if the resource availability sbf belongs to the feasibility region. The sensitivity analysis is applied to evaluate the MC schedulability conditions, and the costs necessary to guarantee some combinations and not others. The sensitivity analysis identified trade-offs between criticality levels and resource provisioning which can be handy while designing systems.

Future work will focus on improving resource usage and optimally combining HI-criticality tasks with LO-criticality tasks. In particular, it will be developed policies to explore the proposed trade-offs. The policies will be implemented to define the best resource provisioning changes with respect to the criticality levels to be assured.

References

[1] L. Abeni and G. Buttazzo. Integrating multimedia applications in hard real-time systems. In *Proceedings of the 19th IEEE Real-Time Systems Symposium*, pages 4–13, dec 1998.

[2] K. Agrawal and S. Baruah. Intractability issues in mixed-criticality scheduling. In *30th Euromicro Conference on Real-Time Systems, ECRTS*, pages 11:1–11:21, 2018.

[3] S. Baruah. Schedulability analysis of mixed-criticality systems with multiple frequency specifications. In *Embedded Software (EMSOFT), 2016 International Conference on*, pages 1–10. IEEE, 2016.

[4] S. Baruah and Z. Guo. Mixed-criticality scheduling upon varying-speed processors. In *Real-Time Systems Symposium (RTSS), IEEE 34th*, pages 68–77. IEEE, 2013.

[5] S. Baruah and Z. Guo. Scheduling mixed-criticality implicit-deadline sporadic task systems upon a varying-speed processor. In *Real-Time Systems Symposium (RTSS)*, pages 31–40. IEEE, 2014.

[6] S. K. Baruah. Dynamic and static-priority scheduling of recurring real-time tasks. In *Real-Time System*, pages 93–128, 2003.

[7] S. K. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, N. Megow, and L. Stougie. Scheduling real-time mixed-criticality jobs. In *Proceedings of the 35th international conference on Mathematical foundations of computer science*, MFCS, pages 90–101, 2010.

[8] S. K. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, S. van der Ster, and L. Stougie. Preemptive uniprocessor scheduling of mixed-criticality sporadic task systems. *J. ACM*, 62(2):14:1–14:33, 2015.

[9] S. K. Baruah, A. Burns, and R. I. Davis. Response-time analysis for mixed criticality systems. In *Proceedings of the 32nd IEEE Real-Time Systems Symposium, RTSS*, pages 34–43, 2011.

[10] S. K. Baruah, A. Burns, and Z. Guo. Scheduling mixed-criticality systems to guarantee some service under all non-erroneous behaviors. In *28th Euromicro Conference on Real-Time Systems, ECRTS*, pages 131–138, 2016.

[11] E. Bini and G. C. Buttazzo. Schedulability analysis of periodic fixed priority systems. *IEEE Transactions on Computers 53 (11), pp. 1462-1473*, pages 1462–1473, 2004.

[12] E. Bini, M. D. Natale, and G. C. Buttazzo. Sensitivity analysis for fixed-priority real-time systems. *Real-Time Systems*, 39(1-3), 2008.

[13] G. C. Buttazzo. Rate monotonic vs. EDF: judgment day. *Real-Time Systems*, 29(1):5–26, 2005.

[14] R. I. Davis and A. Burns. Response time upper bounds for fixed priority real-time systems. In *Proceedings of the 29th IEEE Real-Time Systems Symposium, (RTSS)*, pages 407–418, 2008.

[15] R. I. Davis and A. Burns. Mixed criticality systems - a review. *Technical report, Department of Computer Science, Unviersity of York*, 2016.

[16] D. de Niz, K. Lakshmanan, and R. Rajkumar. On the scheduling of mixed-criticality real-time task sets. In *Real-Time Systems Symposium, RTSS. 30th IEEE*, pages 291 –300, 2009.

[17] R. Ernst, A. Burns, L. Thiele, and J. L. Rhun. Mixed critical system design and analysis. In *Proceedings of the 12th International Conference on Embedded Software, EMSOFT*, pages 247–248, 2012.

[18] R. Ernst and M. D. Natale. Mixed criticality systems - A history of misconceptions? *IEEE Design & Test*, 33(5):65–74, 2016.

[19] T. Fleming, H. Huang, A. Burns, C. D. Gill, S. Baruah, and C. Lu. Corrections to and discussion of "implementation and evaluation of mixed-criticality scheduling approaches for sporadic tasks". *ACM Trans. Embedded Comput. Syst.*, 16(3):77:1–77:4, 2017.

[20] L. George and J. Hermant. Characterization of the space of feasible worst-case execution times for earliest-deadline-first scheduling. *JACIC*, 6(11):604–623, 2009.

[21] Z. Guo and S. Baruah. Mixed-criticality scheduling upon non-monitored varying-speed processors. In *Industrial Embedded Systems (SIES), 8th IEEE International Symposium on*, pages 161–167. IEEE, 2013.

[22] Z. Guo and S. Baruah. Mixed-criticality scheduling upon varying-speed multiprocessors. In *Dependable, Autonomic and Secure Computing (DASC), 12th International Conference on*, pages 237–244. IEEE, 2014.

[23] Z. Guo and S. Baruah. The concurrent consideration of uncertainty in wcets and processor speeds in mixed-criticality systems. In *Proceedings of the 23rd International Conference on Real Time and Networks Systems (RTNS)*, pages 247–256. ACM, 2015.

[24] Z. Guo, L. Santinelli, and K. Yang. EDF schedulability analysis on mixed-criticality systems with permitted failure probability. In *21th IEEE International Conference on Embedded and Real-Time Computing System and Applications*, 2015.

[25] J. Hermant and L. George. A c-space sensitivity analysis of earliest deadline first scheduling. In *Revue des Nouvelles Technologies de l'Information - RNTI issue from ISoLA 2007 Workshop On Leveraging Applications of Formal Methods, Verification and Validation*, 2007.

[26] G. Lipari and E. Bini. Resource partitioning among real-time applications. In *IEEE Euromicro Conference on Real-Time Systems ECRTS*, pages 151–158, 2003.

[27] L. Santinelli, G. C. Buttazzo, and E. Bini. Multi-moded resource reservations. In *17th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS*, pages 37–46, 2011.

[28] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *RTSS '03: Proceedings of the 24th IEEE International Real-Time Systems Symposium*, page 2. IEEE Computer Society, 2003.

[29] L. Thiele, S. Chakraborty, and M. Naedele. Real-time calculus for scheduling hard real-time systems. In *Proc. ISCAS*, volume 4, pages 101–104, 2000.

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