Response Time Analysis of Multiframe Mixed-Criticality Systems

Ishfaq Hussain
CISTER Research Center and ISEP/IPP
Porto, Portugal
hussa@isep.ipp.pt

Muhammad Ali Awan
CISTER Research Centre and ISEP/IPP
Porto, Portugal
awa@isep.ipp.pt

Pedro F. Souto
University of Porto, FEUP-Faculty of Engineering and CISTER Research Centre, Porto, Portugal
pfs@fe.up.pt

Konstantinos Bletsas
CISTER Research Centre and ISEP/IPP
Porto, Portugal
KOBLE@isep.ipp.pt

Benny Akesson
ESI (TNO), Eindhoven
University of Amsterdam
Amsterdam, the Netherlands
benny.akesson@tno.nl

Eduardo Tovar
CISTER Research Centre and ISEP/IPP
Porto, Portugal
emt@isep.ipp.pt

ABSTRACT

The well-known model of Vestal aims to avoid excessive pessimism in the quantification of the processing requirements of mixed-criticality systems, while still guaranteeing the timeliness of higher-criticality functions. This can bring important savings in system costs, and indirectly help meet size, weight and power constraints. This efficiency is promoted via the use of multiple worst-case execution time (WCET) estimates for the same task, with each such estimate characterised by a confidence associated with a different criticality level. However, even this approach can be very pessimistic when the WCET of successive instances of the same task can vary greatly according to a known pattern, as in MP3 and MPEG codecs or the processing of ADVB video streams.

In this paper, we present a schedulability analysis for the multiframe mixed-criticality model, which allows tasks to have multiple, periodically repeating, WCETs in the same mode of operation. Our work extends both the analysis techniques for Static Mixed-Cricality scheduling (SMC) and Adaptive Mixed-Criticality scheduling (AMC), on one hand, and the schedulability analysis for multiframe task systems on the other. Our proposed worst-case response time (WCRT) analysis for multiframe mixed-criticality systems is considerably less pessimistic than applying the SMC, AMC-rtb and AMC-max tests obliviously to the WCET variation patterns. Experimental evaluation with synthetic task sets demonstrates up to 63.8% higher scheduling success ratio (in absolute terms) compared to the best of the frame-oblivious tests.

ACM Reference Format:
Ishfaq Hussain, Muhammad Ali Awan, Pedro F. Souto, Konstantinos Bletsas, Benny Akesson, and Eduardo Tovar. 2019. Response Time Analysis of Multiframe Mixed-Criticality Systems. In *27th International Conference on Real-Time Networks and Systems (RTNS 2019), November 6–8, 2019, Toulouse, France*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3356401.3356405

RTNS 2019, November 6–8, 2019, Toulouse, France

ACM ISBN 978-1-4503-7223-7/19/11…$15.00
https://doi.org/10.1145/3356401.3356405

1 INTRODUCTION

Recent trends in many real-time embedded domains (e.g., automotive and avionics) favour mixed-criticality systems, where computing tasks of different criticalities co-exist on the same processor. A task's criticality is a measure of the severity of the consequences of that task failing, in conjunction with the probability of such a failure. Accordingly, tasks of higher criticalities are developed according to stricter (and costlier) methodologies, and the same holds for the techniques for estimating their worst-case execution times (WCETs). Additionally, scheduling arrangements have to ensure that, even when sharing system resources, a misbehavior of a lower-criticality application cannot affect the timing behavior of a higher-criticality application. One arrangement that ensures that, while also promoting efficient platform utilization, is the mixed-criticality task model of Vestal. The two most established variants of that model are the static variant with execution monitoring [5] and the adaptive mode-based variant [7].

In the simpler static variant of Vestal's model, each task has a criticality level and a WCET estimate for each criticality level lower or equal to its own. At run-time, the execution times of all tasks are monitored and any job that exceeds the WCET estimate corresponding to the degree of confidence appropriate to its criticality level is killed. However, future jobs of the same task will still arrive as normal. For schedulability analysis purposes, the classic worst-case response time (WCRT) recurrence is employed [18], assuming, for all tasks, WCET estimates with the degree of confidence appropriate for the lowest criticality level of the interfering task and of the task under analysis.

In the simplest case, the adaptive model involves two criticality levels and two modes (L and H) of operation. In L-mode, all tasks are present and WCETs are assumed for them which are *probably*, but not *provably*, safe. In case any task executes for its assumed WCET estimate without completing, a mode change is triggered. Then, low-criticality tasks are discarded, and provably safe, but potentially very pessimistic, WCETs are assumed for the remaining tasks. In each mode, all tasks present must provably meet their

deadlines, assuming the respective WCET estimates for that mode. This arrangement considerably mitigates the inefficiency in platform utilization (and commensurate over-engineering) that results from the overwhelming pessimism in the derivation of provably safe WCET estimates for high-criticality tasks.

Both of the above scheduling arrangements promote efficient processor utilization. However, there is another source of inefficient resource use, which the present paper intends to remedy. Namely, when the WCETs of successive jobs of the same task vary greatly by design, according to a known pattern. For such systems, assuming the maximum of the WCETs for all instances of the task (even the ones for which we *know* it is an overestimation) would grossly inflate the processor requirements. For example, in MPEG codecs [19], different kinds of frames (P, I or B) appear in a repeating pattern, with very different worst-case processing requirements. Moreover, Avionics Digital Video Bus (ADVB) [1] frames are transmitted uncompressed, to minimize encoding/encoding delays, but the fact that distinct types of ADVB frames exist (e.g., data, audio or video) implies different end-node processing requirements for each type. In other real-time industrial applications [4, 20] small amounts of data are collected periodically and then summarized and stored in batch after N periods, in an operation that involves costlier processing than that in the preceding periods. The *multiframe task model*, invented by Mok and Chen [22], and its analysis provide a way for efficiently dealing with patterned WCET variations in the single-criticality scheduling. However, until now this model and its existing analysis did not consider the scheduling of mixed-criticality tasks.

Under the multiframe model, a task with N frames is described by N different frame WCETs that repeat, in round-robin manner, in the sequence of its jobs. The well-known Liu and Layland task model [21] then becomes a special case, where the frame size of all tasks is one. The schedulability analysis for the multiframe model leverages the information about the pattern of frame WCET variation and achieves greater accuracy, compared to the frame-agnostic application of analysis for the Liu and Layland task model.

The present work combines the mixed-criticality model of Vestal with the multiframe model, in order to achieve similar improvements in the schedulability testing of mixed-criticality systems whose tasks' WCETs vary according to known patterns. Our three main contributions are the following:

(1) The multiframe task-model is combined with the mixed-criticality Vestal model. It is termed the *multiframe Vestal model* for mixed-criticality systems, in its static and mode-based variants.
(2) Based on the principles of mixed-criticality scheduling and established schedulability tests (SMC, AMC-rtb and AMC-max), we developed *adaptive multiframe mixed-criticality schedulability analyses*, SMMC, AMMC-rtb and AMMC-max, for fixed-priority-scheduled mixed-criticality tasks deployed on a uniprocessor hardware platform.
(3) In experiments with synthetic workloads, the proposed analyses are compared in terms of scheduling success ratio, against the frame-agnostic analyses for the corresponding variants of the Vestal model.

The paper is organised in eight sections. Related work is discussed in Section 2. The system model is presented in Section 3. Section 4 discusses some important background results. The schedulability analysis techniques for the static and mode-based variants of the multiframe Vestal model for mixed-criticality systems are presented in Sections 5 and 6, respectively. Section 7 provides an experimental evaluation, using synthetic task sets, of the schedulability performance of the proposed analyses against those devised for the original (non-multiframe) Vestal model. Finally, Section 8 concludes this work.

2 RELATED WORK

The two main variants (static [5] and adaptive mode-based [7]) of Vestal's original mixed-criticality model [24] characterize each task with multiple worst-case task execution time (WCET) estimates, with a corresponding degree of confidence associated with a different criticality level (up to the task's own criticality). This reflects the fact that, in the industry, the enormous costs of proving the safety of a WCET estimate (and the associated pessimism/overestimation) beyond doubt are justified only for high-criticality tasks [24]. For other tasks, less rigorously derived, WCET estimates are used – probably, but not provably, safe. Based on this model, several mixed-criticality scheduling arrangements have been devised for a variety of hardware platforms in the last decade. The majority of the work is summarized in a survey paper by Burns and Davis [11]. Here, we restrict ourselves to fixed-priority mixed-criticality scheduling algorithms on a uniprocessor platform.

Baruah and Burns [5] first proposed the use of execution-time monitoring, already supported in most embedded platforms. Any job by a lower-criticality task requiring more execution time than its most conservative WCET assumes, is terminated. Baruah, Burns and Davis [7] showed how standard worst-case response time analysis can be applied to this model (termed "Static Mixed-Criticality" or SMC), by assuming, for all tasks, WCET estimates associated with a criticality level never higher than that of the task under analysis. Baruah, Burns and Davis also [7] proposed an adaptive mixed-criticality (AMC) scheduling technique that adds modes of operation. When operating on a mode different from the highest, if any task exceeds its assumed WCET for that mode, the tasks whose criticality level is equal to the current mode of operation are discarded and the system switches to a next higher mode. Two schedulability tests for AMC were devised: AMC-rtb and the tighter, but more complex, AMC-max. The AMC-rtb schedulability test derives a simple-to-compute upper bound, while AMC-max checks all the possible mode-switch instants to derive the worst-case. The AMC-max test is more accurate but not exact. Audsley's priority assignment algorithm is also identified as being optimal for both SMC and AMC, in this work [7].

Asyaban and Kargahi [2] developed exact analysis for AMC at the cost of losing optimality in the priority ordering. The author also derived feasibility interval for mixed criticality periodic tasks with offset [2]. Fleming and Burns [14] extended the dual-criticality AMC schedulability analysis to an arbitrary number of criticality levels and showed that AMC-rtb approximates AMC-max reasonably well. The later AMC-IA schedulability test [15, 16] may

slightly outperform AMC-max in some cases. AMC-rtb and AMC-max were extended for task sets with arbitrary deadlines by Burns and Davis [12]. Zhao et al. [25] improve on AMC-max via a slightly different preemption model.

All aforementioned efforts have been developed in the context of a single-frame task model, where a single WCET of a task is used for all its jobs. These analyses are pessimistic for applications whose execution requirement may vary from one instance (job/frame) to another, but the variation follows a repetitive pattern. Such applications are better modelled with a multiframe task model, which is a generalization of the single-frame task model. In the multiframe task model, the schedulability analysis takes advantage of the varying execution requirement to improve the scheduling performance, potentially reducing system cost.

The multiframe task model was initially introduced by Mok and Chen [22] as a generalisation of Liu and Layland's task model. This model assumes that the WCETs of the successive jobs of the same task can vary with a repetitive pattern. The length of the pattern, in which each job may have a distinct WCET estimate is called the task's "frame size". The schedulability analysis of Mok and Chen [22] showed considerable improvements over the conservative schedulability analyses that use the highest task's WCET estimate as the WCET of its each job. Baruah et al. [8] improved the schedulability analysis of multiframe task model by considering the actual frame pattern rather than an accumulatively monotonic reordered pattern used by Mok an Chen [22]. The multiframe task model was further generalised by Baruah et al. [6] by considering additional task attributes, other than WCET, differing among frames of a task. However, the schedulability analysis for the multiframe task model has not been formulated for mixed-criticality systems. This prevents the benefits of that model from being exploited in the context of mixed-criticality scheduling. The present work therefore introduces schedulability analyses based on the principles of SMC, AMC-rtb and AMC-max, which eliminate the latent pessimism and improve the scheduling performance by leveraging the properties of the multiframe task model.

3 SYSTEM MODEL

In this section, we formalise the mixed-criticality multiframe task model, in both of its variants, static and adaptive mode-based.

Consider a set τ of n mixed-criticality sporadic tasks $(\tau_1, \ldots, \tau_n)$ on a uniprocessor system. Each task τ_i has a relative deadline D_i, a minimum inter-arrival time T_i, a criticality level k_i, which is either high (H) or low (L). For the sake of conciseness, we denote a low-criticality and a high-criticality task as L-task and H-task, respectively. Unlike jobs of conventional (i.e., single frame) tasks, successive jobs of the same multiframe task can differ in their WCETs, in a pattern that repeats after F_i successive jobs. Accordingly, any F_i successive jobs of τ_i are called a *superframe* and each of them is refered to as a frame. In any schedule, the k^{th}, $(k+F_i)^{th}$, $(k+2F_i)^{th}$, ... jobs of τ_i all have the same worst-case execution time behavior. However, even the same frame of the same task has multiple estimates for its WCET, in our model, with different degree of confidence. The semantics vary slightly, according to the particular variant of the model. The choice of scheduling algorithm is orthogonal to the above task model. In this paper, we assume fixed-priority scheduling, which is known as Adaptive Mixed-Criticality (AMC), in the context of the mode-based mixed-criticality scheduling. This means that each task has a fixed priority.

Table 1: Symbols used in the analysis

Symbols	Description
F_i	Number of frames of a task τ_i
$C^x_{i,j}$	WCET of j^{th} frame of τ_i in mode $x \in \{L, H\}$
$g^x(\tau_i, j)$	Cumulative worst-case execution requirement of j successive jobs of τ_i in mode $x \in \{L, H\}$, where $1 \leq j \leq N$
$G^x(\tau_i, t)$	Cumulative worst-case processor request of τ_i in mode x in any time interval of t time units
R^L_i	WCRT of τ_i in L-mode
R^H_i	WCRT of τ_i in H-mode
R^s_i	WCRT of τ_i, if caught in a mode change at time s

Table 1 summarises most symbols used in the worst-case response time analysis. Some other symbols are defined later in Section 6.

3.1 Static variant

In this variant, each job has a WCET estimate per criticality level less than or equal to its task's criticality level. Thus, job j of L-task τ_i has a single WCET estimate, $C^L_{i,j}$, i.e. a low-criticality WCET, or L-WCET for short. On the other hand, job k of H-task τ_i, has two WCET estimates, one L-WCET, $C^L_{i,k}$, and one high-criticality WCET, or H-WCET for short, $C^H_{i,k}$. L-WCETs can be optimistic. Whenever a job of some L-task τ_i, corresponding to its j^{th} frame, executes for its entire L-WCET, $C^L_{i,j}$, without completing, then that particular job is terminated. This does not suppress the arrival of future jobs of τ_i, subject to interarrival time constraints, and respecting the actual frame sequence of the task. Under this model variant, the system is schedulable if (i) no L-task misses its deadline as long as all jobs of all tasks, irrespective of their criticality, execute for up to their $C^L_{i,j}$, and also (ii) no H-task misses its deadline, as long as every job of every task τ_i executes for up to its $C^{k_i}_{i,j}$, where k_i is the criticality of τ_i, as defined earlier.

3.2 Adaptive mode-based variant

As in the mode-based variant of Vestal's model introduced by Baruah and Burns [7], the system operates in different modes – two in this paper, which is also the simplest case. The system boots in the default L-mode, where all tasks (i.e., both of low and high criticality) are present. For the L-mode, WCET estimates are assumed, for all task frames, that are most likely but not necessarily safe. If at any point in time, any job attempts to exceed its assumed WCET estimate for the L-mode, a mode change is triggered. This means that all low-criticality tasks (L-tasks) are discarded and only high-criticality tasks (H-tasks) are allowed to execute. The system is mixed-criticality-schedulable if (i) all tasks meet their deadlines in the L-mode, assuming that their jobs execute in accordance to the WCETs assumed for that mode; and (ii) all H-tasks (including any jobs thereof caught up in the mode change) meet their deadlines in the H-mode, assuming that their jobs can execute for as long as their respective H-mode WCET estimates. The latter are provably safe and typically very pessimistic.

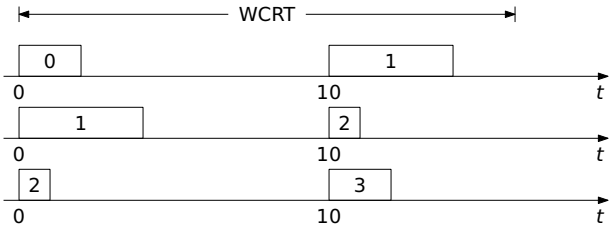

Figure 1: Interference of a multiframe task depends on the phasing of its jobs w.r.t the job under analysis, whose WCRT is shown. (The numbers inside the rectangles are the frame numbers of the interfering task.)

4 BACKGROUND

Our analysis builds on the worst-case response time analysis for the multiframe task model by Baruah et al. [8] and on the different mixed-criticality schedulability analysis techniques by Barauah, Burns and Davis [7]. In this section, we summarize those existing results.

4.1 Multiframe Task Model

To determine the schedulability of a multiframe task, it suffices to check if the WCRT of the frame with the largest WCET is smaller than the task deadline (assumed to be constrained, i.e., $D_i \leq T_i$). However, unlike the "single-frame" model, it is not enough to compute the number of job releases of each interfering task in the WCRT of the task under analysis. This is because each multiframe task τ_i is characterized by a vector of WCET $\left(C_{i,0}, C_{i,1}, \ldots, C_{i,(F_i-1)}\right)$, and the phasing of the released jobs affects the amount of interference. This is illustrated in Fig. 1, which shows the 3 possible phasings of jobs of a task τ_i with period $T_i = 10$ and WCET vector $(2, 4, 1)$, w.r.t to a job under analysis whose WCRT is 16 time units. During this interval there are at most 2 jobs of the interfering task, but the amount of interference depends on the first job in that sequence. As illustrated, this interference is worst, 6, if the first job in the sequence is job 0, $F_k, \ldots$. On the other hand, if the first job in the sequence is job 2, $2 + F_k, \ldots$, the interference is the least, 3.

To efficiently compute the interference exerted by a multiframe task, Baruah et al. [8] define the $g(\tau_i, k)$ function:

$$g(\tau_i, k) = \begin{cases} 0 & \text{if } k = 0 \\ max\left\{ \sum_{\ell=j}^{j+k-1} C_{i,(\ell \bmod F_i)} : 0 \leq j < F_i \right\} & \text{if } 1 \leq k \leq F_i \\ q \cdot g(\tau_i, F_i) + g(\tau_i, r) & \text{otherwise} \\ \quad \text{where } q = k \text{ div } F_i \text{ and } r = k \bmod F_i & \end{cases} \tag{1}$$

which bounds the cumulative WCET of any sequence of k jobs of task τ_i. This function is then used to define function $G(\tau_i, t)$:

$$G(\tau_i, t) = g\left(\tau_i, \left\lceil \frac{t}{T_k} \right\rceil\right) \tag{2}$$

which bounds the cumulative WCET of task τ_i over any time interval of duration t. Finally, the worst-case response time recurrence is:

$$R_i = g(\tau_i, 1) + \sum_{\tau_j \in hp(i)} G\left(\tau_j, R_i\right) \tag{3}$$

4.2 Static Mixed Criticality (SMC)

In the SMC task model [5], each task has a criticality level and it may have a different WCET estimate for each criticality level less than or equal to its criticality level. The conservativeness of those estimates increases with the criticality level. When the job of a task exceeds the WCET estimate corresponding to its own criticality level, it is terminated. With just two criticality levels, this ensures that the interference of any job of a low-criticality task does not exceed C_i^L.

Thus, assuming only two criticality levels, the WCRT recurrence for a single-frame low-criticality task under SMC is

$$R_i = C_i^L + \sum_{\tau_j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j^L \tag{4}$$

and for a high-criticality task it is

$$R_i = C_i^H + \sum_{\tau_j \in hpL(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j^L + \sum_{\tau_j \in hpH(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j^H \tag{5}$$

4.3 Adaptive Mixed Criticality (AMC)

Baruah, Burns and Davis [7] proposed a fixed-priority uniprocessor adaptive mixed-criticality scheduling algorithm (AMC). In [7], two sufficient schedulability analyses (AMC-rtb and AMC-max) are devised for the response time of job of a H-task that is affected by a mode change in the AMC model. These analyses are briefly described below.

4.3.1 AMC-rtb. In this analysis, the WCRT recurrence of a job of task τ_i that is affected by a mode change is:

$$R_i^* = C_i + \sum_{\tau_j \in hpH(i)} \left\lceil \frac{R_i^*}{T_j} \right\rceil C_j^H + \sum_{\tau_j \in hpL(i)} \left\lceil \frac{R_i^L}{T_j} \right\rceil C_j^L \tag{6}$$

where $hpL(i)$ and $hpH(i)$ are the sets of higher-priority low- and high-criticality tasks, respectively, for τ_i.

This analysis makes two conservative assumptions. First, that the number of L-jobs for each interfering L-task is maximum: R_i^L is the latest time the mode change may occur. Second, that all jobs of each interfering H-task take their H-WCET, C_j^H. However, these two assumptions cannot simultaneously hold. The big advantage of this analysis is that it is independent of the instant when the mode change occurs.

4.3.2 AMC-max. This analysis reduces the pessimism by taking into account the instant s, relative to the release of the job under analysis, at which the mode change occurs. This allows for a more accurate account of the number of jobs of an interfering L-task τ_j and therefore of its interference:

$$IL_j(s) = \left\lceil \frac{s}{T_j} \right\rceil C_j^L \tag{7}$$

By taking into account s, it is also possible to be less conservative in the estimate of the number of jobs of a H-task τ_k that complete

after the mode change.

$$M(k,s,t) = min\left\{\left\lceil \frac{t-s-(T_k-D_k)}{T_k}\right\rceil + 1, \left\lceil \frac{t}{T_k}\right\rceil\right\} \quad (8)$$

The number of jobs of a H-task that complete before a mode change is obtained by subtracting this value from an upper bound on the total number of jobs of task τ_k that may be released in time interval t, i.e. $\lceil t/T_k \rceil$. This allows the interference of H-task τ_k to be safely upper-bounded because $C_k^L \leq C_k^H$. Therefore, the interference of H-task τ_k in a time interval of duration t, when the mode change occurs s time units after the beginning of that interval, is given by:

$$IH_k(s,t) = M(k,s,t)\cdot C_k^H + \left(\left\lceil \frac{t}{T_k}\right\rceil - M(k,s,t)\right) C_k^L \quad (9)$$

Thus, the WCRT recurrence used to compute the response time of a job of a H-task τ_i that is affected by a mode change is:

$$R_i^{(n)}(s) = C_i^H + \sum_{\tau_j \in hpL(i)} IL_j(s) + \sum_{\tau_k \in hpH(i)} IH_k(s, R_i^{(n-1)}(s)) \quad (10)$$

The solution of this recurrence, $R_i(s)$ depends on the mode change instant, s. Thus, the response time of τ_i when affected by a mode change is given by:

$$R_i^* = max\{R_i(s) : 0 < s \leq R_i^L\}$$

However, Baruah, Burns and Davis [6] argue that it is enough to compute $R_i(s)$ only for values of s that correspond to the release of jobs of higher-priority L-tasks, when the first job of all these tasks is released at the same time as the job of the task under analysis and these tasks' jobs arrive as soon as possible.

5 ANALYSIS FOR STATIC MULTIFRAME MIXED CRITICALITY SYSTEMS (SMMC)

In this section, we extend SMC analysis to the multiframe model. Like in the multiframe model, in SMC there is a single mode of operation, however tasks may have different levels of criticality and different WCET-estimates, one per criticality-level less or equal to it's own. Therefore, depending on the level of criticality of the task under analysis, different WCET-estimates are used in the WCRT-recurrence, see (4) and (5). For example, the WCRT of a H-task, (5), uses the L-WCET estimate to compute the interference by L-tasks and the H-WCET estimate to compute the interference by H-tasks. Therefore, we need to define additional $g()/G()$ functions that take into account the criticality level of the WCET estimate.

Let $g^L(\tau_i,k)$ be the cumulative L-WCET of any sequence of k jobs of task τ_i. This function is defined as [8]'s $g()$ function, see (1), except that for job j it uses its L-WCET, $C_{i,j}^L$ rather than its criticality-oblivious WCET, C_i:

$$g^L(\tau_i,k) = \begin{cases} 0 & \text{if } k = 0 \\ max\left\{\sum_{\ell=j}^{j+k-1} C^L_{i,(\ell \bmod F_i)} : 0 \leq j < F_i\right\} & \text{if } 1 \leq k \leq F_i \\ q\cdot g^L(\tau_i,F_i) + g^L(\tau_i,r) & \text{otherwise} \\ \quad \text{where } q = k \text{ div } F_i \text{ and } r = k \bmod F_i & \end{cases} \quad (11)$$

Analogously, we define $g^H(\tau_i,k)$, the cumulative H-WCET of any sequence of k jobs of task τ_i. Furthermore, we define the corresponding $G()$ functions, see (2):

$$G^L(\tau_i,t) = g^L\left(\tau_i, \left\lceil \frac{t}{T_i}\right\rceil\right) \quad (12)$$

$$G^H(\tau_i,t) = g^H\left(\tau_i, \left\lceil \frac{t}{T_i}\right\rceil\right) \quad (13)$$

With these definitions, it is straightforward to generalize the SMC's WCRT recurrences to the multiframe model. For L-tasks, the recurrence becomes similar to that for the single-criticality task model (3):

$$R_i = g^L(\tau_i,1) + \sum_{\tau_j \in hp(i)} G^L(\tau_j, R_i) \quad (14)$$

Indeed, like in the original multiframe analysis [8], it suffices to compute the response time for the frame with largest L-WCET, $g^L(\tau_i,1)$. Furthermore, each of the terms in the summation, $G^L(\tau_j,R_i)$, is an upper-bound of the interference by jobs of higher-priority task τ_j in the response time R_i of the taks under analysis, τ_i. This is because it represents the cumulative L-WCET, i.e. computed with same level of confidence as that of $g^L(\tau_i,1)$, of any sequence of τ_j's jobs in time interval, R_i.

However, for H-tasks, the WCRT-recurrence must use the $g()/G()$ function corresponding to the criticality level of each task:

$$R_i = g^H(\tau_i,1) + \sum_{\tau_j \in hpL(i)} G^L(\tau_j,R_i) + \sum_{\tau_k \in hpH(i)} G^H(\tau_k,R_i) \quad (15)$$

Again, it is enough to compute the response time of the job with the largest H-WCET, $g^H(\tau_i,1)$. The interference by higher-priority H-task τ_k is bound by $G^H(\tau_k,R_i)$, the cumulative H-WCET of a sequence of jobs of τ_k in the response time interval, R_i, of the task under analysis, τ_i. The interference by higher-priority L-task τ_j is bound by $G^L(\tau_j,R_i)$, because in SMC the system terminates a job of an L-task as soon as its execution time exceeds the respective L-WCET estimate.

6 ANALYSIS FOR ADAPTIVE MULTIFRAME MIXED CRITICALITY SYSTEMS (AMMC)

In this section, we develop a response time analysis for the adaptive mode-based variant of the multiframe mixed-criticality task model by combining the Multiframe Response Time Analysis in [8] with the AMC analysis in [7]. The idea is straightforward; we use AMC to count the number of jobs of each interfering task, and use Multiframe $g()/G()$ functions, or similar functions, to compute the cumulative WCET of these jobs and therefore the WCRT of the different tasks.

As usual, we need to check the schedulability in:

- **L-mode:** This can be done for all tasks using Baruah's WCRT recurrence, see (3) , and replacing $g()$ and $G()$ with $g^L()$ and $G^L()$, respectively.
- **H-mode:** Again, we can reuse Baruah's WCRT recurrence and replace g and $G()$ with $g^H()$ and $G^H()$, respectively, for all H-tasks.
- **Mode change:** In this case, we must consider, for each H-task, the response time of each job in one of its superframes when

it is caught by a mode change, i.e. when it is released before the mode change but completes only after.

We now focus on the response time of jobs caught by a mode change. Whereas in the single frame model all jobs have the same parameters, in the multiframe model, different jobs have different parameters. Therefore, for each task under analysis, τ_i, we need to:

(1) analyse the WCRT of each job in a superframe of τ_i;
(2) consider the phasing of the interfering tasks, i.e., which job is the first in the sequence.

Figure 1 illustrates the need for the latter. With respect to (1), consider two different jobs j and k ($j \neq k$) of a superframe of τ_i, it may be the case that $C^L_{i,j} > C^L_{i,k}$ and $C^H_{i,j} < C^H_{i,k}$. Thus, the schedulability of job j cannot be inferred from the response time of job k, nor vice-versa. Of course, it may not be necessary to compute the response time of all jobs. E.g., if $C^L_{i,j} > C^L_{i,k}$ and $C^H_{i,j} \geq C^H_{i,k}$, clearly it is enough to analyse the response time of job j of τ_i. Task τ_i is schedulable, if the WCRT of all jobs in one of its superframes does not exceed D_i.

In each of the following subsections, we describe how to generalise each of the AMC analyses to the multiframe model.

6.1 AMMC-rtb: AMC-rtb to multiframe tasks

As summarized in Section 4.3.1, AMC-rtb considers that each higher-priority L-task may interfere up to R^L_i. In the multiframe model, for each higher-priority L-task τ_k, we need to consider the cumulative WCET of a sequence of τ_k's jobs. Thus, the interference of a higher priority L-task, τ_k, on job j of H-task τ_i is given by:

$$G^L(\tau_k, R^L_{i,j})$$

where $R^L_{i,j}$ is the WCRT in L-mode of job j of task τ_k. With respect to each higher-priority H-task, τ_k, AMC-rtb assumes that every job of τ_k executes for its H-mode WCET (H-WCET for short) over the entire WCRT of task τ_i's job j when it is caught by a mode change, $R^*_{i,j}$. Thus the interference of τ_k, on that job is given by:

$$G^H(\tau_k, R^*_{i,j})$$

Accordingly, the WCRT of job j of a H-task τ_i released before the mode change but completing after the mode change is upper-bounded by:

$$R^*_{i,j} = g^H(\tau_i, 1) + \sum_{k \in hpL(i)} G^L(\tau_k, R^L_{i,j}) + \sum_{k \in hpH(i)} G^H(\tau_k, R^*_{i,j}) \tag{16}$$

The WCRT of H-task τ_i is therefore the maximum of the WCRT of all frames in a superframe of τ_i:

$$R^*_i = max\{R^*_{i,j} : 0 \leq j < F_i\}$$

6.2 AMMC-max: AMC-max to multiframe tasks

AMC-max reduces the pessimism in AMC-rtb by explicitly considering the mode change instant, s, as summarized in Section 4.3.2. In particular, rather than assuming that all jobs of a H-task complete after the mode change, AMC-max provides a tighter upper-bound of the number of jobs of a H-task that complete after the mode change. As a consequence, AMC-max may underestimate the number of jobs of that task that complete before the mode change. (This is also done in AMC-rtb, where the number of jobs of a H-task that complete before the mode change is assumed to be 0.) Thus, critical for the safety of AMC-max is that the cumulative WCET of a task in a time interval comprising the mode change instant does not decrease, if the number of jobs of that task that complete after the mode change is overestimated.

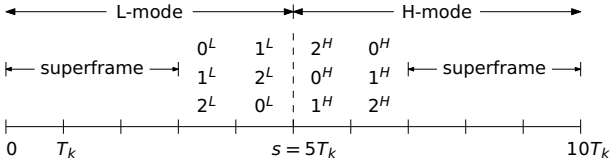

Figure 2: Interference of a H-task depends on the phasing of its jobs.

So that we can safely apply AMC-max accounting of H-jobs that complete after the mode change to the multiframe model, the following must hold:

Lemma 6.1. *Consider a sequence of successive jobs by a multiframe task τ_i with a fixed fixed length n such that the last $n^H < n$ of these jobs terminate after the mode change. Let n'^H ($n \geq n'^H \geq n^H$) be an overestimation of n^H. Then assuming the termination of the last n'^H jobs after the mode change does not decrease the cumulative WCET.*

Proof. The proof is by induction on the number of jobs that terminate after the mode switch.

Base step: $n'^H = n^H$ and therefore the two job sequences are the same and so are their cumulative WCET.

Induction step. Consider a sequence of these n jobs such that the last h ($n > h \geq n^H$) terminate after the mode switch. Let j be the last job in that sequence that terminates before the mode change (such a job exists because $h < n$). If we assume that job j terminates after the mode change the difference between the cumulative WCET of the new and the previous sequences is $C^H_{i,j} - C^L_{i,j}$. Because by assumption $C^H_{i,j} \geq C^L_{i,j}$ this difference is non negative, and therefore, by the induction step assumption, the cumulative WCET is not smaller than that of the sequence with n^H jobs terminating after the mode change. □

Thus, for multiframe tasks (just as for single-frame tasks), it is safe for the analysis to overestimate, in a given sequence of jobs by an interfering task, the number of jobs that execute for their H-WCET. In particular, this means that we can use AMC-max's upper-bound on the number of jobs that complete after the mode change, (8), and use it to derive the number of jobs that complete before the mode change to be used in the WCRT recurrence.

The challenge is that the phasing of the interfering task affects the amount of interference. The difference with respect to Baruah's multiframe analysis is that now there may be two subsequences, one before the mode change and another after the mode change. Figure 2 illustrates this. It shows an interfering task τ_k with 3 frames. The time interval under consideration is 10 times T_k and we assume that the mode change occurs exactly at the middle of

that interval. Thus, in this interval, there are 10 releases of jobs of τ_k, 5 before the mode change and another 5 after the mode change. Hence there is a superframe plus 2 frames, both before and after the mode change. The cumulative WCET of a super-frame is independent of the initial frame in the superframe. Therefore, we need only to find the worst-case cumulative WCET of a sequence of 4 frames, such that the two first frames take their L-WCET, whereas the other 2 frames take their H-WCET. One of the following frame sequences leads to the worst-case cumulative execution time: $(0^L, 1^L, 2^H, 0^H), (1^L, 2^L, 0^H, 1^H), (2^L, 0^L, 1^H, 2^H)$, where we denote by i^L (i^H) a frame whose L-WCET (H-WCET) is equal to the L-WCET (H-WCET) of frame i.

Formally, we define function $g^*(\tau_i, \ell^L, \ell^H)$ that computes the cumulative WCET of a sequence of jobs of H-task τ_i, which is composed by a subsequence of ℓ^L jobs that take their L-WCET followed by a subsequence of ℓ^H that take their H-WCET:

$$g^*(\tau_i, \ell^L, \ell^H) = \begin{cases} g^H(\tau_i, \ell^H) & \text{if } \ell^L = 0 \\ g^L(\tau_i, \ell^L) & \text{if } \ell^L \neq 0 \wedge \ell^H = 0 \\ max\left\{ \sum_{k=j}^{j+\ell^L-1} C^L_{i,(k \bmod F_i)} + \sum_{k=j+\ell^L}^{j+\ell^L+\ell^H-1} C^H_{i,(k \bmod F_i)} : 0 \le j < F_i \right\} & \text{if } 1 \le \ell^L < F_i \wedge 1 \le \ell^H < F_i \\ q^L \cdot g^L(\tau_i, F_i) + g^*(\tau_i, r^L, r^H) + q^H \cdot g^H(\tau_i, F_i) & \text{otherwise} \end{cases} \tag{17}$$

where $q^L = (\ell^L \text{ div } F_i)$, $r^L = (\ell^L \bmod F_i)$, $q^H = (\ell^H \text{ div } F_i)$ and $r^H = (\ell^H \bmod F_i)$.

Thus the WCRT of job j of H-task τ_i is upper bounded by:

$$R^*_{i,j}(s) = g^H(\tau_i, 1) + \sum_{k \in hpL(i)} G^L(\tau_k, s) + \sum_{k \in hpH(i)} g^*(\tau_k, \ell^L(k, R^*_{i,j}(s), s), \ell^H(k, R^*_{i,j}(s), s)) \tag{18}$$

where:

$$\ell^H(k, R^*_{i,j}(s), s) = M(k, s, R^*_{i,j}(s))$$

$$\ell^L(k, R^*_{i,j}(s), s) = \left\lceil \frac{R^*_{i,j}}{T_k} \right\rceil - \ell^H(k, R^*_{i,j}(s), s)$$

Therefore:

$$R^*_{i,j} = max\{R^*_{i,j}(s) : 0 < s \le R^L_i\}$$

Like in AMC-max, to compute the WCRT of job j of task τ_i, we need to compute the response time only for the values of s that correspond to the release of jobs of higher-priority L-tasks, when the first job of each of these tasks is released at the same time as the job under analysis and the other jobs arrive as soon as possible. Like in AMMC-rtb, to compute the WCRT of task τ_i, we need to compute the response time of all jobs in a superframe and take the maximum of these values:

$$R^*_i = max\{R^*_{i,j} : 0 \le j < F_i\}$$

6.2.1 Implementation of the $g^*()$ function. [8] provides an efficient implementation $g()$ that relies on an array $M_k[F_k + 1]$ per task τ_k, such that $M_k[i] = g(\tau_k, i)$, for $0 \le i \le F_K$. This array is computed before initiating the response time analysis proper, and the complexity of its computation is $O(F_k^2)$. Analogously, in order to implement the $g^*()$ function, we define a 2-dimensional $(F_k+1)\times(F_k+1)$ matrix, M_k per H-task τ_k, where $M_k[i][j] = g^*(\tau_k, i, j)$ for $0 \le i \le F_k$ and $0 \le j \le F_k$. Note that element $M[0][j] = g^H(\tau_k, j)$, and therefore $M[0][F_k]$ is the H-WCET of a superframe of τ_k. Likewise, element $M[i][0] = g^L(\tau_k, i)$, and therefore $M[F_k][0]$ is the L-WCET of a superframe of τ_k. Furthermore, all other elements of the last column and of the last line are not needed and therefore need not be computed. Indeed, the value of $g^*(\tau_k, \ell, h)$ can be efficiently computed as follows from the elements of the M_k matrix:

$$(\ell \text{ div } F_k)\cdot M[F_k][0] + M[\ell \bmod F_k][h \bmod F_k] + (h \text{ div } F_k)\cdot M[0][F_k]$$

Computation of matrix M_k is very similar to the computation of vector M_k in [8]. The first step is to compute two $F_k \times F_k$ matrices L_k and H_k. These matrices are just like $F_k \times F_k$ matrix D_k in [8], except that L_k and H_k use respectively the L-WCET and H-WCET of τ_k's frames. More specifically element $L_k[i][j] = \sum_{\ell=i}^{i+j} C^L_{k,(\ell mod F_k)}$, i.e. element $L_k[i][j]$ is the the cumulative WCET of a sequence of $(i + 1)$ L-frames that starts with frame j. Algorithm 1 shows the code segment presented in [8] adapted to initialize the elements of matrix L_k; except for the notation, the only difference is the use of the L-WCET in lines 2 and 5. The initialization of H_k is identical. The complexity of this code segment is $O(F_k^2)$.

Algorithm 1 Computation of auxiliary matrix L_k.

```
// L_k is a F_k × F_k matrix, such that L_k[i][j] = Σ_{ℓ=i}^{i+j} C^L_{k,(ℓ mod F_k)}
1: for (j = 0; j < F_k; j++) // First line
2:     L_k[0][j] = C^L_{k,j};
3: for (i = 1; i < F_k; i++)
4:     for (j = 0; j < F_k; j++)
5:         L_k[i][j] = L_k[i − 1][j] + C^L_{i,(i+j)%F_k};
```

Algorithm 2 shows the code segment used for computing matrix M_k from matrices L_k and H_k. For the sake of readability, we use the $max(m, n)$ function, which returns the maximum of its two integer arguments. The algorithm uses the elements of L_k and H_k in a very similar way to the computation of the corresponding array in [8] from the elements of the D_k matrix. As a result the complexity for computing each element of the M_k matrix, $O(F_k)$, is the same as the complexity for computing each element of the array in [8]. However because the M_k matrix has $(F_k + 1)^2$ elements rather than $(F_k + 1)$ elements, the complexity of the computation of matrix M_k is $O(F_k^3)$ rather than $O(F_k^2)$. Note that all elements of the last row except for the first one are not computed, as they are not needed. The same for all elements of the last column except the first one.

7 EVALUATION

The proposed analyses are implemented in a Java tool [23] to evaluate their scheduling performance. This Java tool has two modules. The first module generates the synthetic workload for the specified input parameters. The second module implements the different

Algorithm 2 Computation of $(F_k+1)\times(F_k+1)$ matrix M_k. $M_k[i][j]$ is the cumulative WCET of any sequence of i L-jobs followed by j H-jobs of task τ_k

```
1:  M_k[0][0] = 0
    // First row is g^H: this is the algorithm in [8]
2:  for (h = 1; h <= F_k; h++) { // h is the length of the sequence
3:      M_k[0][h] = 0;
4:      for (i = 0; i < F_k; i++) // i is the first frame
5:          M_k[0][h] = max(M_k[0][h], H_k[h − 1][i]);
6:  }
    // First column is g^L: this is the algorithm in [8]
7:  for (ℓ = 1; ℓ <= F_k; ℓ++) { // ℓ is the length of the sequence
8:      M_k[ℓ][0] = 0;
9:      for (i = 0; i < F_k; i++) // i is the first frame
10:         M_k[ℓ][0] = max(M_k[ℓ][0], L_k[ℓ − 1][i]);
11: }
    // Remaining elements: based on the algorithm in [8]
12: for (ℓ = 1; ℓ < F_k; ℓ++) // ℓ is the length of the L-sequence
13:     for (h = 1; h < F_k; h++) { // h: H-sequence's length
14:         M_k[ℓ][h] = 0;
15:         for (i = 0; i < F_k; i++) // i: first frame in L-sequence
16:             M_k[ℓ][h] = max(M_k[ℓ][h],
                              L_k[ℓ − 1][i] + H_k[h − 1][(i + ℓ)%F_k]);
17:     }
```

schedulability analyses. The generation of the synthetic task sets is controlled through the following parameters.

- Task periods are generated in the range of 10 msec to 1 sec using a log-uniform distribution. We assume implicit deadlines ($D_i = T_i$), even though the proposed analyses hold for the more general constrained deadlines ($D_i \leq T_i$).
- The UUnifast algorithm [10, 13] is used to generate the L-mode utilization ($U^L_{i,1}$) for the first frame of a task τ_i in an unbiased way. The L-WCET of the first frame[1] of each task is then $C^L_{i,1} = T_i \times U^L_{i,1}$. The number of frames for each task is selected randomly with a uniform distribution within a range of $[1, \alpha]$, where $\alpha > 1$ is a user-defined integer parameter. The L-WCETs of other frames of a task are randomly selected within an interval of $[\beta \times C^L_{i,1}, C^L_{i,1}]$ with uniform distribution, where $\beta \in (0, 1]$ is a task generation parameter limiting the L-WCET variation among a task's frames.
- The user-defined fraction of H-tasks $\xi \in (0,1)$ in the task set.
- The H-WCET of the j^{th} frame of H-task τ_i ($C^H_{i,j}$) is derived by linearly scaling up that frame's L-WCET ($C^L_{i,j}$) with a user-defined factor of κ, i.e., $C^H_{i,j} = \kappa \times C^L_{i,j}$.

We used Audsley's priority assignment [3] for all schedulability tests. The target utilization is varied within a range of $(0, 1]$ with a step size of 0.1. Different random class objects are defined for utilization, minimum inter-arrival time, number of frames and L-WCET of each frame. These random class objects are seeded with different odd numbers and reused in successive replications [17]. For each set of input parameters, 1000 random task-sets are generated. The parameters used in this evaluation are summarized in Table 2.

[1]For convenience, without loss of generality (since shift-rotating the order of the frames results in an equivalent multiframe task), the first frame has the biggest L-WCET.

Table 2: Overview of Parameters

Parameters	Values	Default
H-WCET scale-up factor (κ)	$\{2 : 0.5 : 6\}$	3
Task-set size (n)	$\{8 : 4 : 32\}$	16
Fraction of H-tasks in τ (ξ)	$\{0.2 : 0.05 : 0.7\}$	0.4
Upper bound on # of frames (α)	$\{3 : 1 : 10\}$	5
Lower bound on L-WCET (β) variation	$\{0.1 : 0.1 : 0.8\}$	0.2
Inter-arrival time (T_i)	$10ms$ to $1s$	N/A

To avoid generating a huge number of plots, in each of our experiments, we only vary one parameter at a time, with the other parameters conforming to their default values, displayed in Table 2. The number of plots is further reduced by plotting weighted schedulability rather than schedulability success ratio for each possible combination of input parameters. The weighted schedulability representation condenses three dimensional-plots to two-dimensional plots [9, 11] by eliminating the axis of task set utilization. This performance metric gives more weight to task sets with higher utilization. Using notation from [11], let $S_y(\tau, p)$ represent the result (0 or 1) of the schedulability test y for a given task-set τ with an input parameter p. Then $W_y(p)$, the weighted schedulability for that test y as a function p, is presented in (19), where $U^L(\tau)$ is the nominal system utilization of the task set τ in the L-mode.

$$W_y(p) = \frac{\sum_{\forall \tau}(U^L(\tau) \cdot S_y(\tau, p))}{\sum_{\forall \tau} U^L(\tau)} \tag{19}$$

7.1 Results

We compared the weighted schedulability of the three proposed schedulability analyses (SMMC, AMMC-rtb, AMMC-max) against the existing ones (SMC, AMC-rtb and AMC-max). For the SMC, AMC-rtb and AMC-max tests, the multiframe task is transformed into an instance of the classic (i.e., single-frame) mixed-criticality task-model by pessimistically discarding frame information. In this transformation, for any task τ_i, its L-WCET and H-WCET are set to $C^L_i = \max_{j=0}^{F_i-1} C^L_{i,j}$ and $C^H_i = \max_{j=0}^{F_i-1} C^H_{i,j}$, respectively.

Figure 3 presents the effect of varying the H-WCET scale-up factor (κ) on all analyses. A higher value for κ increases the H-mode processor requirement of high-criticality tasks, hence the weighted schedulability of all analyses decreases accordingly. The multiframe-based analyses perform better than the conventional AMC-max, AMC-rtb and SMC analyses by leveraging frame information. As expected, AMC-max outperforms AMC-rtb and AMMC-max outperforms AMMC-rtb, albeit by very small margins in both cases (less than 3%). The difference between adaptive and static variants comes from the fact that static analyses do not drop L-tasks, this makes the WCRT of H-tasks computed with SMC more likely to exceed their deadlines. SMMC performs better at low values of κ compared to SMC, AMC-rtb and AMC-max, but the difference among them decreases with an increase in κ, because task sets become harder to schedule. A higher κ reduces the overall schedulability ratio for each analysis, therefore, the differences in performance between the multiframe- and single-frame-based analyses also decrease.

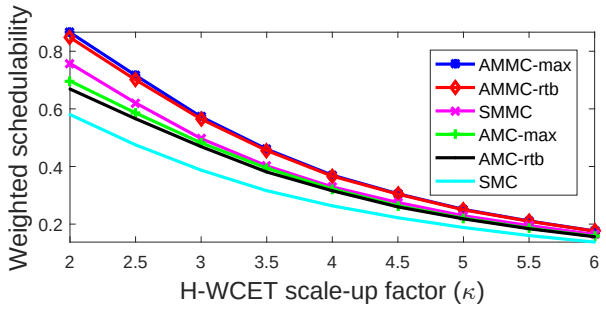

Figure 3: Effect of H-WCET variation on different analysis

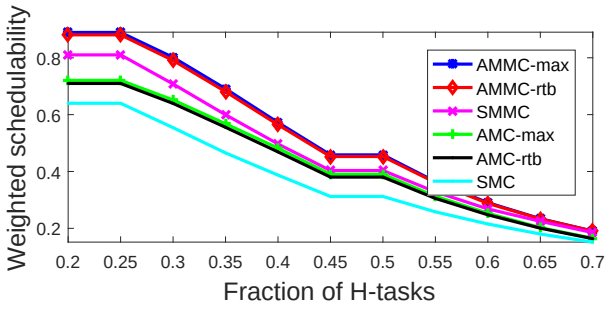

Figure 4: Weighted Schedulability vs. H-task share

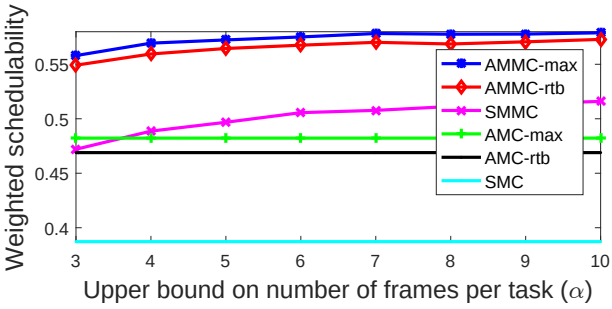

Figure 5: Weighted schedulability vs. max. number of frames in a task

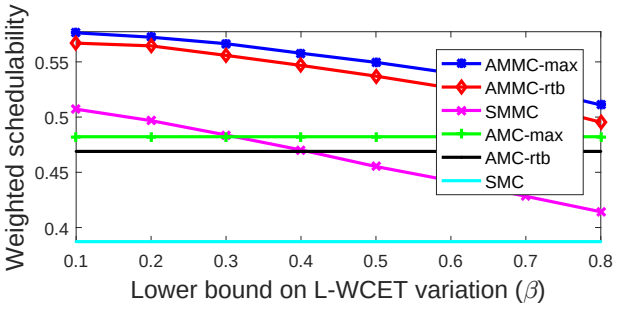

Figure 6: Comparison of different analysis techniques for L-WCET variation

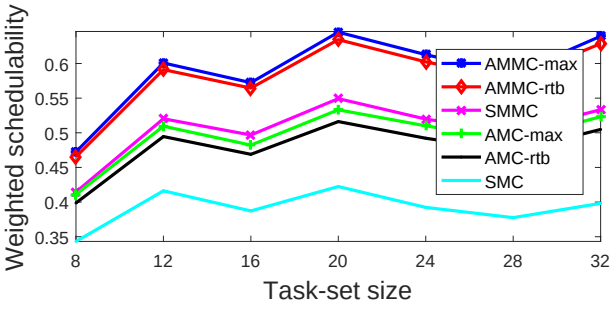

Figure 7: Effect of task set size variation over weighted schedulabaility

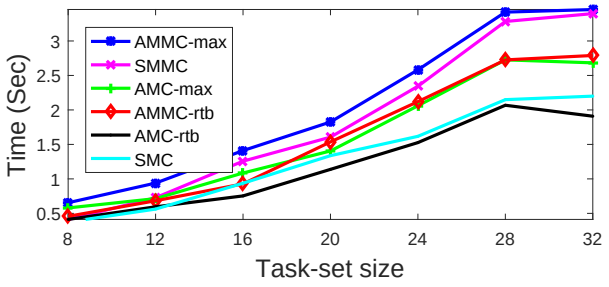

Figure 8: Impact of task set size variation on the average running time

A higher fraction of H-tasks (ξ) increases the processor requirement of the system in H-mode. Hence, the weighted schedulability of all analyses decreases (Figure 4). Their absolute difference in terms of weighted schedulability also decreases, as the number of feasible task-sets decreases with a higher fraction of H-tasks.

The potential for improvements in weighted schedulability for SMMC, AMMC-rtb and AMMC-max over single-frame-based analyses increases when the number of frames per task is higher. A larger value of upper bound for the number of frames per tasks results in more frames per task on average as well. With more

Table 3: Maximum absolute difference in schedulability success ratio of each multiframe analysis from its corresponding single-frame analysis.

	Varied parameters				
Analyses	κ	$\lvert\tau\rvert$	ξ	α	β
AMMC-max	52.4%	22.3%	**63.8%**	17.8%	16.9%
AMMC-rtb	46.9%	22.5%	**61.6%**	16.5%	15.4%
SMMC	35.1%	31.9%	**47.7%**	24.4%	22.6%

frames in a task, the probability of having low-WCET frames increases, which in turn magnifies the improvement from leveraging the frame information (or, equivalently, magnifies the pessimism when disregarding it). Hence, the weighted schedulability of the multiframe-based analyses improves when the upper bound on the number of frames per task is higher, as shown in Figure 5. The classic SMC, AMC-rtb and AMC-max are insensitive to this parameter, as these analyses assume the maximum estimates for L-WCET and H-WCET over all frames. For low α, leveraging frame information does not compensate for the pessimism of SMMC, causing it to under perform compared to AMC-rtb and AMC-max.

The WCET variation limiting parameter β defines a lower bound for the ratio between the smallest frame L-WCET and the greatest frame WCET of a given task. A higher value of β decreases this range, and consequently, increases the average execution requirement of task's frames, all other things remaining equal. Hence, the weighted schedulability of the multiframe-based analyses decreases with an increase in β (Figure 6). One important observation is that the absolute difference among the weighted schedulability of AMMC-max, AMMC-rtb and SMMC increases with an increase in β. The pessimism in AMMC-rtb against AMMC-max becomes a major differentiator when the potential of gaining from the multiframe properties becomes limited. The same behavior is shown by SMMC against SMC. Similarly to the bound on the maximum number of frames per task, β has no effect on SMC, AMC-rtb and AMC-max, as the maximum of the WCET estimates over all frames are assumed in each mode.

The effect of variation in the number of tasks on the weighted schedulability is presented in Figure 7. The important observation is that, similarly to single-criticality systems, the weighted schedulability improves with larger task set sizes, as more low-utilization tasks are easier to schedule compared to fewer high-utilization tasks. The fact that during task generation, the number of H-tasks is rounded up to the nearest integer as needed (i.e., when the target fraction of H-tasks would result in a non-integer number) explains the saw-tooth shape of the weighted schedulability plots of each analysis.

To quantify the benefits in terms of *non-weighted* schedulability success ratio, Table 3 shows the maximum absolute difference in non-weighted schedulability success ratio of each multiframe analysis over its corresponding single-frame analysis. In the best case, the AMMC-max, AMMC-rtb and SMMC analyses achieve up to 63.8%, 61.6% and 47.7%, higher schedulability successes ratio, respectively, compared to their corresponding single-frame analyses. The bold values in Table 3 shows the highest difference achieved by each analysis among all experiments.

Table 4: Average running time (in seconds) of each analysis with default parameters.

AMMC-max	AMMC-rtb	SMMC	AMC-max	AMC-rtb	SMC
1.41	0.9282	1.257	1.0854	0.7533	0.9370

Finally, we experimentally explore the running time of each analysis. The platform used for these experiments has 32 GB RAM and 12 Intel Xeon ES-2420 v2 cores with maximum frequency of 2.20 GHz and ran Linux Mint 18.2 Sylvia. Except the task set size, the running time is virtually insensitive to other parameters. Figure 8 presents the average running times of all analyses for different task-set sizes in seconds. AMMC-max takes longer than other analyses as it requires more computation while computing the processor requirement in each iteration. As expected, the running times of all tests increase with an increase in task-set size as the feasibility testing has to be performed for each task. For the default parameters, the running time of all analyses is presented in Table 4. For both single and multiframe analyses, AM(M)C-max is slowest and AM(M)C-rtb is fastest. Multiframe analyses are always slower than single-frame analyses.

8 CONCLUSIONS

For uniprocessor platforms, we extended the mode-based mixed-criticality model of Vestal, which assumed a single worst-case execution time per task per mode. The extension accommodate multiframe tasks that have different WCETs for their jobs, in a repeating patterns. For this extended model, we formulated schedulability analyses (SMMC, AMMC-rtb and AMMC-max) that leverage the frame information, resulting in greater accuracy over the state-of-the-art mixed-criticality schedulability tests (SMC, AMC-rtb and AMC-max) that, pessimistically, have to discard frame information and use a single WCET per task per mode, in order to be applied. Experimental evaluation with synthetic task sets confirm the benefits (up to 63.8% in schedulability success ratio) of the new model and its analysis. In the future, we intend to consider a partitioned multicore arrangement and incorporate the effects of memory stalls under memory access regulation into the schedulability analysis to make it more realistic and more applicable to real-world systems.

ACKNOWLEDGEMENTS

This work was partially supported by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UID/CEC/04234). This research is supported by the Netherlands Organization for Applied Scientific Research TNO.

REFERENCES

[1] AERONAUTICAL RADIO, INC. 2013. ARINC specification 818-2 Avionics Digital Video Bus (ADVB) High Data Rate (818-2 ed.). AERONAUTICAL RADIO, INC.

[2] Sedigheh Asyaban and Mehdi Kargahi. 2018. An exact schedulability test for fixed-priority preemptive mixed-criticality real-time systems. 54, 1 (01 Jan 2018), 32–90.

[3] N. C. Audsley. 2001. On Priority Assignment in Fixed Priority Scheduling. 79, 1 (2001), 39–44.

[4] CM Bailey, A Burns, AJ Wellings, and CH Forsyth. 1995. Keynote paper: A performance analysis of a hard real-time system. Control Engineering Practice

3, 4 (1995), 447–464.
[5] Sanjoy Baruah and Alan Burns. 2011. Implementing mixed criticality systems in Ada. In 16th Ada-Europe Conference. 174–188.
[6] Sanjoy Baruah, Deji Chen, Sergey Gorinsky, and Aloysius Mok. 1999. Generalized Multiframe Tasks. 17, 1 (01 Jul 1999), 5–22.
[7] S. K. Baruah, A. Burns, and R. I. Davis. 2011. Response-Time Analysis for Mixed Criticality Systems. 34–43.
[8] S. K. Baruah and A. Mok. 1999. Static-priority scheduling of multiframe tasks. 38–45.
[9] Andrea Bastoni, Björn Brandenburg, and James Anderson. 2010. Cache-related preemption and migration delays: Empirical approximation and impact on schedulability. Proceedings of OSPERT (2010), 33–44.
[10] E. Bini and G.C. Buttazzo. 2009. Measuring the Performance of Schedulability tests. 30, 1-2 (2009), 129–154.
[11] A. Burns and R.I. Davis. 2014. Adaptive Mixed Criticality Scheduling with Deferred Preemption. 21–30.
[12] Alan Burns and Robert Ian Davis. 2017. Response Time Analysis for Mixed Criticality Systems with Arbitrary Deadlines. In 5th International Workshop on Mixed Criticality Systems (WMC 2017). York.
[13] Robert I. Davis and Alan Burns. 2009. Priority Assignment for Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Real-Time Systems. 398–409.
[14] T. Fleming and A. Burns. 2013. Extending Mixed Criticality Scheduling. In Proc. WMC, RTSS. 7–12.
[15] Tom Fleming, Huang-Ming Huang, Alan Burns, Chris Gill, Sanjoy Baruah, and Chenyang Lu. 2017. Corrections to and Discussion of "Implementation and Evaluation of Mixed-criticality Scheduling Approaches for Sporadic Tasks". 16, 3 (2017), 77:1–77:4.
[16] Huang-Ming Huang, Christopher Gill, and Chenyang Lu. 2014. Implementation and Evaluation of Mixed-criticality Scheduling Approaches for Sporadic Tasks. 13, 4s, Article 126 (April 2014), 25 pages.
[17] Raj Jain. 1991. The art of computer systems performance analysis - techniques for experimental design, measurement, simulation, and modeling. Wiley. I–XXVII, 1–685 pages.
[18] M. Joseph and P. Pandya. 1986. Finding Response Times in a Real-Time System. 29, 5 (1986), 390–395.
[19] Didier Le Gall. 1991. MPEG: A Video Compression Standard for Multimedia Applications. Commun. ACM 34, 4 (April 1991), 46–58. https://doi.org/10.1145/103085.103090
[20] John P. Lehoczky, Lui Sha, and Y. Ding. 1989. The Rate Monotonic Scheduling Algorithm: Exact Characterization and Average Case Behavior. 166–171.
[21] C. Liu and J. Layland. 1973. Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. J. ACM 20 (1973), 46–61.
[22] A. K. Mok and D. Chen. 1997. A multiframe model for real-time tasks. 23, 10 (Oct 1997), 635–645.
[23] Borislav Nikolic, Muhammad Ali Awan, and Stefan M. Petters. 2011. SPARTS: Simulator for Power Aware and Real-Time Systems. Changsha, China, 999–1004.
[24] S. Vestal. 2007. Preemptive Scheduling of Multi-criticality Systems with Varying Degrees of Execution Time Assurance.
[25] Q. Zhao, Z. Gu, and H. Zeng. 2013. PT-AMC: Integrating Preemption Thresholds into Mixed-Criticality Scheduling. 141–146. https://doi.org/10.7873/DATE.2013.042