HAL
open science

Exact Scheduling Analysis of Non-Accumulatively Monotonic Multiframe Tasks

A. Zuhily, Alan Burns

To cite this version:

A. Zuhily, Alan Burns. Exact Scheduling Analysis of Non-Accumulatively Monotonic Multiframe Tasks. 16th International Conference on Real-Time and Network Systems (RTNS 2008), Isabelle Puaut, Oct 2008, Rennes, France. inria-00336464

HAL Id: inria-00336464

https://inria.hal.science/inria-00336464v1

Submitted on 4 Nov 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Exact Scheduling Analysis of Non-Accumulatively Monotonic Multiframe Tasks

A. Zuhily and A. Burns
Real Time System Group,
Department of Computer Science, University of York,
UK.

Abstract

In this paper, we present the exact analysis of the worst case response time of the general multiframe (MF) task model executing on a uniprocessor according to the fixed priority scheduling scheme. The analysis is given in three steps. Firstly, we relax the restriction of Accumulatively Monotonic (AM) and present the basic response time analysis where we optimize the number of frames that have to be considered in such analysis; we show how they can be significantly reduced by eliminating non critical frames that are dominated by other frames. Secondly, we extend this analysis to be applicable to MF tasks with release jitter. Lastly, the basic analysis is improved to cope with arbitrary deadlines.

1 Introduction

The fundamental principle in the real-time *multiframe*, MF, task is that its worst-case execution time is different from one phase to another of its execution, for example, a task that executes with the worst-case execution times of 10ms and 5ms is said to have two frames. An example often found in industrial applications [3], is a periodic task that does a small amount of data collection in each period consuming a small execution time, but then summaries and stores this data every n cycles using a much more expensive algorithm that consumes a larger execution time. A further example is found within the MPEG coding standard where there are three types of video frames (usually represented by the letters I, P and B). The I frame usually takes much more decoding than the others, but may occur only every 10 frames. The assumption that all frames are I frames leads to poor utilization and the system could be theoretically unschedulable whilst practically it is schedulable. Presenting the decoder as a MF task allows the schedulability results to be optimised.

Mok and Chen [10, 11] were the first to introduce the MF concept as a generalisation of the classic Liu and Layland model [8]. They proposed a utilization based schedulability test, for fixed priority scheduling, under Rate Monotonic, RM, [8] priority assignment (the greater period the task has the lower priority it is assigned). They gave a utilization bound, assuming the execution time sequence of each MF task has a particular restriction that they called *Accumulatively Monotonic*, AM, (see Section 2). Subsequent papers (Section 3.1) have improved this utilization bound but their tests remain inexact (i.e. sufficient but not necessary). Whilst in this paper, we relax this AM restriction and present the exact scheduling analysis of MF tasks in terms of response time analysis.

In general, testing the schedulability of a set of MF tasks requires all possible phasings of the tasks to be examined; which leads to an exhaustive enumeration problem (i.e. an intractable problem). However, for a particular application, not all phases may need to be examined. We show how the *dominant* frames, that can give rise to the worst-case response times of lower priority tasks, can be identified and their usage reduce the processing required for the response time analysis. This analysis is then extended in this paper in two directions to be applicable to the MF tasks with: firstly, release jitter (RJ); and secondly arbitrary deadlines.

The paper is organised as follows. In the next section our system model and notation are introduced, while prior contributions are summarised in Section 3. Section 4 uses the notion of a *critical frame* (i.e. dominant frame) to reduce the scale of the scheduling problem; then the exact response time analysis is introduced for the general MF model. This analysis is developed to include release jitter of the tasks in Section 5 then to be applicable to the arbitrary deadlines scenario in Section 6. Conclusions are provided in Section 7.

2 System Model

Analysis in this paper considers a system that consists of N independent MF tasks. Each MF task τ_i consists, in its turn, of a sequence of n_i frames; where the frames are distinguished by their execution times. Frames are execut-

ing on a uniprocessor using the preemptive fixed priority scheduling policy. Priorities of MF tasks in the system are ordered consecutively with τ_1 having the highest priority in the system and τ_N the lowest priority. All frames in the same MF task have the same priority, are released with a fixed or minimum time interval T_i and have a relative deadline D_i.

Each frame in the MF task might have a worst case execution time that may be different from other frames in the same MF task. In other words, a MF task, τ_i, has n_i worst case execution times, $C_i^k; k = 0..n_i - 1$. Without loss of generality, we assume that the sequence of the execution time values is always in its shortest form; where **the shortest form of a sequence** is the shortest subsequence when repeated a number of times generates the original sequence. That is because, from the response time analysis point of view, the behaviour of the execution of a MF task whose execution times consist of repetitive subsequences is the same as the behaviour of the original sequence. For example, the execution behaviour of the MF task that is represented by the sequence $(8, 1, 4, 3, 8, 1, 4, 3)$ is the same as the execution behaviour of the subsequence $(8, 1, 4, 3)$. The extracted subsequence, $(8, 1, 4, 3)$, is referred to as the shortest form of the sequence $(8, 1, 4, 3, 8, 1, 4, 3)$.

To illustrate the problem of analysing the response time of MF tasks, Table 1 represents a simple system example with 2 tasks τ_1 and τ_2 where τ_1 is a MF task with 4 frames represented by the execution time values 8, 1, 4 and 3 and τ_2 has just one frame.

$task$	C	T	priority
τ_1	8, 1, 4, 3	10	1
τ_2	x	20	2

Table 1. System Example

Finding the worst case response time R_2 of τ_2, whatever its execution time is, requires finding the maximum amount of possible interference from τ_1. Table 2 shows values of interference that τ_1 generates from different initial frames (exe. seq. and inv. respectively stand for *execution time sequence* and *number of invocations*). It can be seen from Table 2 that the maximum amount of interference τ_1 generates, in the case of one invocation (i.e. 1 inv.), is when the frame, whose execution time is 8, is released first. While the maximum amount of interference, in the case of two invocations, is when the frame, whose execution time is 3, is released first. The maximum amount of interference, in the case of three invocations of τ_1 is when the frame, whose execution time is 4, is released first. Finally, in the case of four invocations of τ_1, the amount of interference τ_1 provides remains the same (i.e. 16 in this example) whatever frame is released first. Frames that could generate the maximum amount of interference will be called *critical frames*; which are, in this example, the frames whose execution times are 8, 4, and 3, but not 1 because any of the other frames can be considered as a critical frame on behalf of 1 (see Section 4.1). We consider the frame of a MF task τ_j as critical when it has two properties; firstly, it can generate the maximum amount of interference within a lower priority task for at least one of τ_j's invocations; and secondly there is no other frame in τ_j that generates greater or equal amount of interference for all possible numbers of τ_j's invocations.

frame Location	exe. seq.	1 inv	2 inv	3 inv	4 inv
0	8, 1, 4, 3	**8**	9	13	**16**
1	1, 4, 3, 8	1	5	8	**16**
2	4, 3, 8, 1	4	7	**15**	**16**
3	3, 8, 1, 4	3	**11**	12	**16**

Table 2. Possible Interference From τ_1

To calculate the amount of interference a frame generates within the response time of a lower priority task, we have to know the relative number of invocations (interference) the MF task is producing within this response time. For this reason we define a *cumulative function* for the frame whose location is x in the MF task τ_j to represent the amount of interference this frame generates. Definition 1 introduces this cumulative function.

Definition 1 *Given a MF task τ_j with n_j execution times $(C_j^0, C_j^1, .., C_j^{(n_j-1)})$. The cumulative function (ξ_j) of the frame whose location is x for a given number of τ_j's invocations, k, is the amount of interference that the MF task generates starting from frame x and proceeding for that number of invocations; and is given by Equation (1)*

$$\xi_j^x(k) = \sum_{f=x}^{x+k-1} C_j^{f \bmod n_j} \tag{1}$$

where $x = 0, .., n_j - 1,\ and\ k = 1, 2, ...$ For example, the value of $\xi_1^0(2)$ for the task τ_1 in Table 1 is 9.

From a scheduling point of view, a frame in a MF task is considered critical when it can give rise to the maximum interference for some lower priority tasks and so can lead to the worst case response time of that task. The maximum interference is generated by a frame of a MF task when its cumulative function is greater than the cumulative function of any other frame of the same MF task; for at least one possible number of interference. The following definition formally introduces the critical frame of a MF task.

Definition 2 The frame whose location is x in the MF task τ_j whose execution time sequence is in its shortest form, is critical *if and only if* $\exists\, k = 1, 2, .., n_j - 1, \forall y \neq x$:

$$\xi_j^x(k) > \xi_j^y(k) \tag{2}$$

For example, the first frame (i. e. the frame with location 0) of the MF task $(8, 1, 4, 3)$ is a critical frame because $\exists\, k = 1, \forall y \neq 0; \xi^0(1) > \xi^y(1)$.

We call the frame whose execution time is maximum the *Peak frame*:

Definition 3 The Peak Frame of a MF Task *is one of its frames with the maximum execution time.*

Note from Definition 2 that having the execution time sequence in its shortest form means that if we have more than one peak frame then at least one of the peak frames must be a critical frame; otherwise the execution time sequence is not in its shortest form. For example, the MF task τ where $C = (8, 1, 4, 3, 8)$ has two peak frames with locations 0 and 4 whose execution times are both 8; the first peak is not critical but the other is critical.

Mok and Chen[10] force one of the peak frames to be the only critical frame of the MF by giving a restriction on its execution times. This restriction is called Accumulatively Monotonic, AM, where one of the peak frames of the AM multiframe task is the only frame that generates the maximum amount of interference for all possible number of interference (invocations). Informally, all execution times of the AM multiframe task are dominated by one execution time value. Equation (3) represents this restriction.

$$\sum_{f=m}^{m+k} C^{(f \bmod n)} \geq \sum_{f=y}^{y+k} C^{(f \bmod n)} \tag{3}$$

$$\forall y, k = 0, 1, .., n-1$$

Where C^m is one of the maximum execution times $(C^0, C^1, .., C^{(n-1)})$ that satisfies Equation (3). For example, for the execution time sequence $(3, 8, 7, 3)$, $m = 1$ and $C^1 = 8$.

In this paper, we relax this AM restriction and present the response time analysis of the MF tasks in general taking into account the critical frame concept.

3 Related Work

Because we are concerned with response time scheduling analysis of the MF tasks, previous contributions must be covered within two fields: the schedulability analysis of MF tasks, and response time analysis.

3.1 Schedulability of MF Tasks

As the research into scheduling analysis started from the *utilization* point of view, we introduce here the beginning of this research. Liu and Layland [8] and Serlin[12], with the RM priority assignment algorithm, introduced a sufficient but not necessary scheduling test for a restricted model. The test was based upon the least upper bound of the processor utilization factor; which is given by $\sum_{i=1}^{i=N} \frac{C_i}{T_i}$. The test based on the criterion that a task set is schedulable if its processor utilization is less than a given upper bound; which is $N(2^{\frac{1}{N}} - 1)$ or 0.69 for big N. This upper bound is extended by different researchers to serve the MF model.

Although Mok and Chen [10, 11] were the first who studied the scheduling of the MF tasks, Han [4] gave another scheduling test, under RM priority assignment, that is better than Mok's test in the sense that AM multiframe task set with peak utilization[1] larger than their upper bound is not schedulable using Mok and Chen's utilization bound but can be found schedulable by Han's test.

Takada et al. [13] investigated the schedulability of MF tasks and gave a scheduling condition of the MF model, under the fixed priority scheme, showing that the time complexity of the schedulability decision becomes at least $\prod_i {n_i}^2$. They introduced a sufficient feasibility decision algorithm using a maximum interference function. However, in our paper we are interested in presenting an exact way of analysing the schedulability of the general MF tasks and optimizing the number of frames, of a MF task, that are needed for checking the schedulability of a lower priority MF task.

Traor et al. [14] mentioned in their paper that the MF model was a particular case of tasks with offset (transactions), so they assume that their offset analysis can be applied to the MF model. However, we assume in our model that the MF model is different from the general transaction model because the offset model fails to utilise the fact that all jobs from the same MF task have the same period.

Kuo et al. [7] gave another improved utilization bound for the scheduling test of systems with AM multiframe tasks. The main idea of the test is to merge tasks with harmonic periods to reduce the number of tasks that has to be considered in the schedulability test, and then apply Mok's bound to the merged tasks.

More recently, Lu et al. [9] improved Kuo's utilization test and presented new scheduling conditions for AM multiframe tasks within the utilization domain and assuming RM priority assignment. The improvement is that they used Kuo's method to merge the tasks and then they applied their

[1] The peak utilization is the summation of all utilizations of the peak frames in the system

[2] $\prod_i n_i$ means the product of all numbers of frames over all tasks in the system

test to the merged tasks. The schedulability status, under their approach, depends on the total peak utilization, U, of the AM multiframe tasks being less than a defined upper bound called the Conditional Bound function, CB.

Although Lu's analysis improves previous results, it still remains inexact as well as being applicable only to the AM model; while the response time analysis is exact and tractable for the AM model. We show in [16] that even for the AM model, the exact response time schedulability analysis is always better than Lu's analysis.

As can be seen from the above contributions, all published approaches are inexact since all of them are either in the utilization domain or are only sufficient – as well as most of them being only applicable to the AM model. In this paper, we provide exact scheduling analysis for general MF tasks within the response time domain. In the following, we cover contributions in the response time domain.

3.2 Standard Response Time Analysis

For schedulability analysis under fixed priority preemptive scheduling, each task is considered as schedulable if it finishes its execution before its deadline. To clarify, a task τ_i is schedulable if its worst case response time R_i (which measures the maximum time from when the task is released until it finishes its execution) is less than or equal to its deadline. Usually, the response time of a task represents two kinds of execution: execution of the task itself and execution of other tasks in the system that is presented as 'interference' from higher priority tasks.

The scheduling research into the response time analysis begins when Joseph and Pandya [6] followed by Audsley et al. [1] introduced a formula, Equation (4), for finding the worst case response time of a task τ_i having specific restrictions and assuming Liu and Layland's critical instant [8]. Where Liu and Layland's critical instant, and so the worst case response time, of a task is when that task is released simultaneously with all higher priority tasks.

$$R_i = C_i + \sum_{j=1}^{i-1} \lceil \frac{R_i}{T_j} \rceil C_j \tag{4}$$

where $\sum_{j=1}^{i-1} \lceil \frac{R_i}{T_j} \rceil C_j$ is the amount of interference from tasks whose priorities are higher than the priority of τ_i.

To solve Equation (4), a recurrence relation is used as in Equation (5); where $l = 0, 1, 2, ...$ and $r_i^0 = C_i$. The smallest non-negative solution of Equation (5) represents the worst case response time of τ_i. In other words, the worst case response time is obtained when it is found that $r_i^{l+1} = r_i^l (= R_i$ for the smallest value of $l)$. However, in the case that r_i^{l+1} becomes greater than the deadline of the task, τ_i is not guaranteed to meet its deadline, so we say that the task is unschedulable.

$$r_i^{l+1} = C_i + \sum_{j=1}^{i-1} \lceil \frac{r_i^l}{T_j} \rceil C_j \tag{5}$$

In terms of MF tasks, Equation (4) assumes that C_j is constant for all frames in a MF task, so the critical instant of the task is not affected by changing the start frame of the higher priority tasks – because all frames in the standard task generate the same amount of interference. In this paper, the restriction of having constant C is relaxed; so, this standard response time formula requires modification.

Baruah et al. [2] used the response time analysis to give a tractable but sufficient schedulability test for a system of MF tasks. They merged the execution time sequences of the MF tasks taking into account the maximum amount of interference that higher priority MF tasks provide. Then, they applied the fixed point algorithm to determine the worst case response time of the peak frame of the lower priority MF task considering the merged execution time sequences of the higher priority MF tasks. Although this analysis is within response time domain, the test is inexact while in our contribution we provide exact response time analysis.

4 Exact Analysis of General MF Tasks

In this section, the two restrictions of having constant execution times and AM multiframe tasks are relaxed. Initially the worst case response time analysis of a MF task τ_i requires checking all possible combinations of all frames of the MF tasks whose priorities are higher than τ_i; which means we have to consider $\prod_{j=1}^{i-1} n_j$ different combinations of the frames [13]. However, the critical frame concept (see Section 2) leads to the requirement of only checking the critical frames of the MF tasks whose priorities are higher than τ_i's. We show in [15], by evaluation, that the number of critical frames is mostly located in the range $[45\%, 60\%]$ of the original number of frames; which reduces the number of required combinations for finding the worst case response time of τ_i. So, to present the exact worst case response time analysis we follow two steps: in the first step, we identify the critical frames; while in the second step, we present the exact response time formula depending on these critical frames.

4.1 Identifying Critical Frames

To identify the critical frames of a MF task, we follow a scenario where we first identify the non-critical frames then consider the remaining frames of this MF task as critical.

To identify the non-criticality of the yth frame[3] of the MF task τ_j, we invert Definition2 so we say that y^{th} frame

[3]For simplicity of presentation, we call the frame whose location is y the yth frame with the knowledge that y's values starts from 0

is not critical if there is another x^{th} frame whose cumulative function is always greater than or equal to the cumulative function of the yth one; for all τ_j's numbers of invocations. However, we sufficiently consider only $n_j - 1$ invocations of τ_j because the amount of the generated interference from any of τ_j's frames increases with a fixed rate after $n_j - 1$ invocations. To symbolize the definition of the non-critical frame of a MF task τ_j having n_j execution times $(C_j^0, C_j^1, .., C_j^{(n_j-1)})$ within its shortest form, the yth frame is definitely not critical if $\exists x \in \{0, .., n_j - 1\}$ *and* $x \neq y$; where Equation 6 is satisfied $\forall k = 1, 2, .., n_j - 1$.

$$\xi_j^x(k) \geq \xi_j^y(k). \tag{6}$$

The criterion of the non-criticality of a frame that is represented by Equation (6) means that the amount of interference the yth frame generates is never more than the amount of interference the xth frame generates, so the yth frame is never critical. We call the yth frame, in this case, a *dominated frame* and the xth frame the *dominant frame*. So, applying this criterion on all frames of a MF task judges the non-critical frames and therefore the remaining frames of the MF task are critical. One successful application of this criterion results in the frames with the minimum execution times are never critical, the following theorem proves that.

Theorem 4.1 *Given a MF task τ_i with n_i execution times, where $n_i > 1$, in its shortest form, the frames with the minimum value of execution times are never critical frames.*

Proof
The cumulative functions of the two frames: the frame with the minimum execution time and the subsequent frame to the minimum, are given by Equations (7) and (8):

$$\xi_i^{min}(k) = \sum_{j=min}^{min+k-1} C_i^{(j \ mod \ n_i)}, \tag{7}$$

$$\xi_i^{min+1}(k) = \sum_{j=min+1}^{min+k} C_i^{(j \ mod \ n_i)}; \tag{8}$$

where min is the location of the minimum execution time in its sequence.
For each $k = 1, 2, ..$, we subtract Equation (7) from Equation (8), so we get

$$\xi_i^{min+1}(k) - \xi_i^{min}(k) = C_i^{((min+k) \ mod \ n_i)} - C_i^{(min \ mod \ n_i)}.$$

As C_i^{min} is the minimum execution time of all frames, the right side of the equation is never negative so the left side of the equation is also never negative. So, $\xi_i^{min+1}(k) \geq \xi_i^{min}(k); \forall k = 1, 2, ..$; which means that each frame with the minimum execution time is always dominated by the frame it is followed by. □

Note that if a MF task has more than one minimum in its sequence of execution times, then each minimum frame is dominated by the following frame, which results that all minimums are not critical. Also, note that this theorem shows that in the worst case, when there is only one minimum frame for τ_i, there is a maximum of $n_i - 1$ critical frames since this minimum is definitely non-critical. So, for a MF task with at least two different execution times, the frames that have to be considered in the response time analysis are the frames whose execution times are not minimum.

Assume $\hat{L}_i$ to be the set of the critical frame locations. Then, from $\hat{L}_j$ we define $\hat{V}_i$ to represent the combinations of the higher priority MF tasks as the cartesian product of all sets of the critical frame locations for all tasks whose priorities are higher than τ_i. This cartesian product $\hat{V}_i$ is defined as follow. Let $\hat{V}_1 = \{\}$, $\hat{V}_2 = \hat{L}_1$ and for $i > 2$ define $\hat{V}_i$ to be the cartesian product of $\hat{L}_1$ ... $\hat{L}_{i-1}$. In other words, $\hat{V}_i = \hat{L}_1 \times \hat{L}_2 \times .. \times \hat{L}_{i-1}$.

4.2 Exact Analysis of MF tasks

This section improves the standard analysis that is given in Section 3.2 to cope with the Non-AM multiframe tasks. To start with, the critical instant of a MF task τ_i is known as the instant that leads to the worst case response time of τ_i. In comparison to the standard critical instant and as the critical frames of a MF task are the only frames that can lead to the worst case response time of lower priority MF tasks, we now identify the critical instant of a MF task τ_i as in Definition 4; where the peak frame of τ_i is the frame that generates the maximum amount of execution of τ_i assuming that $C_i^k \leq T_i; \forall k$.

Definition 4 The critical instant of a MF task τ_i *is the simultaneous release, of the peak frame of τ_i with the critical frames of higher priority MF tasks, that lead to the worst case response time of τ_i.*

Assuming the critical instant in Definition 4, the response time analysis of τ_i considers its peak frame and the previous reduced set of critical frames for each MF task whose priority is higher than the priority of τ_i. So, to find the worst case response time of τ_i we have to maximize its response time over all frame combinations of the higher priority MF tasks. Symbolically, the worst case response time of τ_i has to be maximized over all values in $\hat{V}_i$; which is given as $R_i = \max_{\tilde{v} \in \hat{V}_i}\{R_{i,\tilde{v}}\}$; where $\{R_{i,\tilde{v}}\}$ is the response time of τ_i that is relative to the simultaneous release which is presented as the combination $\tilde{v}$ from the cartesian product $\hat{V}_i$ and is given by Equation (9)

$$R_{i,\tilde{v}} = C_i^{m_i} + \sum_{j=1}^{i-1} \xi_j^{\tilde{v}_j} \ (\lceil \frac{R_{i,\tilde{v}}}{T_j} \rceil); \tag{9}$$

where m_i is the location of the peak frame of the task τ_i. $\tilde{v_j}$ is the jth element of the vector $\tilde{v}$ (i.e. τ_j's critical frame that is relative to the combination $\tilde{v}$).

Equation (9) is solved by forming a recurrence relation given by Equation (10).

$$r_{i,\tilde{v}}^{l+1} = C_i^{m_i} + \sum_{j=1}^{i-1} \xi_j^{\tilde{v_j}} \left(\lceil \frac{r_{i,\tilde{v}}^{l}}{T_j} \rceil\right) \tag{10}$$

where $r_{i,\tilde{v}}^{0} = C_i^{m_i}$ and $l = 0, 1, ..$ until $r_{i,\tilde{v}}^{l+1} = r_{i,\tilde{v}}^{l}$ or $r_{i,\tilde{v}}^{l+1}$ becomes greater than D_i.

Example

To illustrate the exact response time analysis of the non-AM multiframe tasks, Table 3 represents parameters of a simple system example with three tasks τ_1, τ_2 and τ_3. Within this example we investigate the schedulability of τ_3.

$task$	C	$T = D$	priority
τ_1	3, 4, 6, 8, 7, 5	10	high
τ_2	5, 6, 10, 7	40	medium
τ_3	1, 2, 3	60	low

Table 3. System Example

Without the critical frame concept, we have to evaluate the response time of τ_3 over all possible frames of the higher priority tasks τ_1 and τ_2; which means we had to do 24 evaluations. However, as dominated frames are never critical, we need only consider the critical frames of both τ_1 and τ_2, this reduces the number of needed evaluations.

In this example, τ_1 and τ_2 have less than $n_j - 1$ critical frames; for $j = 1, 2$; applying Equation (6) to τ_1 shows that the frame with the execution time 8 dominates both frames with the execution times 7 and 5. So, both frames with the execution times 7 and 5 are never critical and there is no need to check them to find the worst case response time of τ_3. The same argument is applied to the frame with the execution time 10 in τ_2; where it dominates the frame with the execution time 7. Tables 4 and 5 show the amount of interference each frame of each MF task τ_1 and τ_2 generates; which are represented by the function ξ, (1 inv. means $k = 1$ for $\xi_j^x(k)$, and so on 2 inv., 3 inv., ..). Note how the minimum frame is always dominated by the frame that it is followed by.

Therefore, to find the worst case response time of τ_3 we have to evaluate its response time over the critical frames of τ_1 and τ_2 which are presented by their locations $\hat{L_1}$ and $\hat{L_2}$ as follows $\hat{L_1} = \{1, 2, 3\}$ and $\hat{L_2} = \{1, 2\}$. So, $\hat{V_3} = \hat{L_1} \times \hat{L_2} = \{(1, 1), (1, 2), (2, 1), (2, 2), (3, 1), (3, 2)\}$.

Now, for each combination $\tilde{v} \in \hat{V_3}$, we find the relative response time of τ_3 by applying Equation (10). For example, to find $R_{3,(1,1)}$, we do the following $r_{3,(1,1)}^{0} = 3$,

frame Location	1 inv.	2 inv.	3 inv.	4 inv.	5 inv.
0	3	7	13	21	28
1	**4**	**10**	**18**	**25**	**30**
2	**6**	**14**	**21**	**26**	**29**
3	**8**	**15**	**20**	**23**	**27**
4	7	12	15	19	25
5	5	8	12	18	26

Table 4. Cumulative functions of τ_1

Frame Location	1 inv.	2 inv.	3 inv.
0	5	11	21
1	**6**	**16**	**23**
2	**10**	**17**	**22**
3	7	12	18

Table 5. Cumulative functions of τ_2

$r_{3,(1,1)}^{1} = 3 + \sum_{j=1}^{2} \xi_j^{\tilde{v_j}} \left(\lceil \frac{3}{T_j} \rceil\right) = 3 + 4 + 6 = 13$,
$r_{3(1,1)}^{2} = 3 + \sum_{j=1}^{2} \xi_j^{\tilde{v_j}} \left(\lceil \frac{13}{T_j} \rceil\right) = 19$,
$r_{3,(1,1)}^{3} = 19 = r_{3(1,1)}^{2}$. So, $R_{3,(1,1)} = 19$.
Similarly we find all $R_{3,\tilde{v}}$ for all elements in $\hat{V_3}$ to get the values in Table 6. Therefore, $R_3 = \max\{19, 30, 29, 38, 39, 36\} = 39$. Note the maximum is over only 6 values instead of 24 and also note that the critical instant of τ_3 is the simultaneous release of τ_3 having execution time of 3, τ_2 having execution time of 10 and τ_1 having execution time of 6 (i. e. $\tilde{v} = (2, 2)$).

frame Location	0	1	2	3	4	5
0	-	-	-	-	-	-
1	-	19	30	29	-	-
2	-	38	**39**	36	-	-
3	-	-	-	-	-	-

Table 6. Possible Response Times of τ_3

As $R_3 < D_3$, so τ_3 is schedulable.

5 Analysis of MF Tasks with Release Jitter

In this section we extend the basic response time analysis of the non AM multiframe tasks to include the MF tasks that are subjected to release jitter; where *release jitter* of a task is defined as the maximum variation in the release times of the task's frames[5]. In other words, when τ_j is subjected to release jitter then it is not strictly periodic with period T_j; because its frames are released somewhere within time interval of length J_j and then every period T_j. Mathemati-

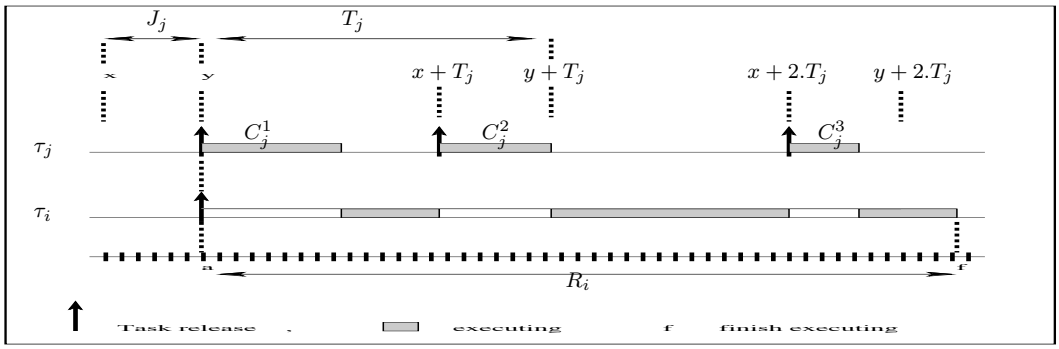

Figure 1. Illustration of release jitter problem

cally, let a_j^k be the kth frame of τ_j, then

$$k.T_j + x \leq a_j^k \leq k.T_j + y; \ \forall k = 0, 1, 2, .. \quad (11)$$

$$where \ J_j = y - x.$$

The first issue in analysing the worst case response time of the MF task τ_i is to identify its frame situation that leads to this worst case. Assume that the worst case situation of a lower priority task τ_i starts execution at time a, then a higher priority MF task τ_j preempts τ_i the most when τ_j and τ_i have three properties in their execution scenario. The first property is that the first frame of τ_j must be its critical frame that leads to the maximum interference within τ_i's execution. While the second property is that the first frame of τ_j takes place rightmost in its release jitter interval (i.e. J_j) and its next frames (i.e. second, third, ..) take place leftmost in its release jitter interval (i.e. from Equation (11) $a_j^0 = y$ and $a_j^k = x + k.T_j; k = 1, 2, ...$).The third property is that τ_i and τ_j start their first execution simultaneously (i.e. $a_j^0 = y = a_i^0 = a$) because this simultaneous release, of the peak frame of τ_i and critical frames of the higher priority MF tasks, provides the maximum amount of preemption time for τ_i. Figure 5 illustrates this worst case scenario.

The second issue in the analysis is to check if having release jitter (RJ) of a non AM multiframe task τ_j affects its critical frame set that was discussed in the previous section. In fact, RJ could affect the number of interference τ_j generates on a lower priority task whilst the critical frame set stays the same. To clarify, assume k_1 is the number of interference τ_j generates on a lower priority task having no RJ of τ_j and x_1 is the critical frame that is relative to k_1; this k_1 could increase to k_2 when τ_j is subjected to RJ and hence the critical frame could be changed relatively to x_2 which copes with this k_2. However, both of x_1 and x_2 are from the critical frame set because this set depends on the maximum amount of interference that τ_j generates for each of its possible number of interference. So even if the number of interference is increased by RJ, the relative critical frame will be one of the critical frame set. Therefore the critical frame set stays the same as explained in Section 4.

The following example explains how the critical frame could be changed in the case of RJ, but is still one of the critical frame set. Suppose a system with three MF tasks in Table 7, the critical frame locations of τ_1 and τ_2 are $\{1, 2, 3, 4\}$ and $\{1, 2, 3\}$ respectively.

$task$	C	T	priority
τ_1	3, 4, 6, 7, 8, 6, 8	10	high
τ_2	5, 6, 7, 10	40	medium
τ_3	1, 2, 3	60	low

Table 7. System Example

Firstly, we present all possible response times of τ_3 assuming there is no RJ for any tasks in the system. Table 8 presents all possible response times that are relative to all critical frames of τ_1 and τ_2. So, we see from the table that the worst case response time of τ_3 is 50 and the relative critical frames of τ_1 and τ_2 are the execution times whose locations are 3 and 3; which represent the execution times 7 and 10 respectively.

frame Location	0	1	2	**3**	4	5	6
0	-	-	-	-	-	-	-
1	-	19	30	30	34	-	-
2	-	20	37	39	35	-	-
3	-	30	40	**50**	38	-	-

Table 8. Responses of τ_3 when no RJ

Now, assume τ_1 has RJ of 1; then this RJ gives rise to extra interference from τ_1 and therefore its critical frame is changed to include this extra interference. To find out how the critical frame is changed, Table 9 presents all possible response times of τ_3 (calculated by applying Equation (12)) that are relative to all critical frames of τ_1 and τ_2. We see

from this table that the worst case response time of τ_3 is 56 and critical frames of τ_1 and τ_2 are 6 and 10. While according to the frames of 7 and 10 of τ_1 and τ_2, the relative response time of τ_3 in the case of RJ of τ_1 (i.e. $J_1 = 1$) is $R_3 = 54$. So, the specific critical frame could be changed when RJ exists but is still one of the critical frame set of the relative MF task.

frame Location	0	1	**2**	3	4	5	6
0	-	-	-	-	-	-	-
1	-	19	36	38	34	-	-
2	-	27	37	39	35	-	-
3	-	38	**56**	54	38	-	-

Table 9. Responses of τ_3 when $J_1 = 1$

So, analysis of the worst case response time of τ_i has to be maximized over all combinations of the critical frames of the higher priority MF tasks. Symbolically, recall from Section 4, $\hat{L_j}$ and $\hat{V_i}$ as the locations of the critical frames of τ_j and the cartesian product of $\hat{L_j}$; where τ_j represents the MF task whose priority is higher than τ_i, then what is left is to find the formula that represents $R_{i,\tilde{v}}$ for a specific combination, $\tilde{v}$, of the critical frames of all higher priority MF tasks.

Theorem 5.1 *Given a real time system consisting of N non-AM multiframe tasks τ_j; $j = 1, 2, .., N$, each MF task τ_j has a maximum release jitter equals J_j and has its peak frame at position m_j; the worst case response time of τ_i is given by the smallest non-negative solution to Equation (12):*

$$R_{i,\tilde{v}} = C_i^{m_i} + \sum_{j=1}^{i-1} \xi_j^{\tilde{v_j}} \left(\lceil \frac{R_{i,\tilde{v}} + J_j}{T_j} \rceil\right) \qquad (12)$$

where $\tilde{v_j}$ is the jth element of the vector $\tilde{v}$ that represents one of the combinations of the critical frames of the MF tasks whose priorities are higher than τ_i.

Proof Assume I is the maximum amount of interference from tasks whose priorities are higher than τ_i, then $I = \sum_{j=1}^{i-1} I_j$; where I_j is the maximum amount of interference from the higher priority MF task τ_j. We divide this amount into two parts $I_j = C_j^{\tilde{v_j}} + I_j^{rest}$; where $C_j^{\tilde{v_j}}$ is the first interference that τ_j provides within $(T_j - J_j)$ while I_j^{rest} is the amount of interference that τ_j provides within $R_{i,\tilde{v}} - (T_j - J_j)$ starting from the frame that follows the first one. So, I_j^{rest} is given by: $I_j^{rest} = \xi_j^{(\tilde{v_j}+1)}(\lceil \frac{R_{i,\tilde{v}} - (T_j - J_j)}{T_j} \rceil)$. Therefore,
$I_j = C_j^{\tilde{v_j}} + \xi_j^{(\tilde{v_j}+1)}(\lceil \frac{R_{i,\tilde{v}} - (T_j - J_j)}{T_j} \rceil)$

$I_j = \xi_j^{\tilde{v_j}}(\lceil \frac{R_{i,\tilde{v}} - (T_j - J_j)}{T_j} \rceil + 1)$ because the cumulative function starts from a previous frame so an extra interference has been added,

$I_j = \xi_j^{\tilde{v_j}}(\lceil \frac{R_i - (T_j - J_j)}{T_j} + 1 \rceil)$ because we add an integer to the ceiling function so we can move this integer into the ceiling function,

$$I_j = \xi_j^{\tilde{v_j}}(\lceil \frac{R_i - (T_j - J_j)}{T_j} + \frac{T_j}{T_j} \rceil) = \xi_j^{\tilde{v_j}}(\lceil \frac{R_i + J_j}{T_j} \rceil).$$

So, the maximum amount of interference from tasks that have higher priority than the MF task τ_i, assuming $C_j^{\tilde{v_j}}$ is the first execution time of τ_j (remember that $\tilde{v_j}$ is location of a critical frame, of τ_j, that is relative to the combination $\tilde{v}$), is given by $I = \sum_{j=1}^{i-1} \xi_j^{\tilde{v_j}}(\lceil \frac{R_i + J_j}{T_j} \rceil)$; and therefore, the worst case response time of the task τ_i is given by the smallest non negative solution to Equation (12).□

Solving Equation (12) is given by forming a recurrence equation given by $r_{i,\tilde{v}}^{l+1} = C_i^{m_i} + \sum_{j=1}^{i-1} \xi_j^{\tilde{v_j}} (\lceil \frac{r_{i,\tilde{v}}^{l} + J_j}{T_j} \rceil)$ where $r_{i,\tilde{v}}^{0} = C_i^{m_i}$ and $l = 0, 1, ..$ till $r_{i,\tilde{v}}^{l+1} = r_{i,\tilde{v}}^{l} = R_{i,\tilde{v}}$. However, if $r_{i,\tilde{v}}^{l+1} > D_i - J_i$, we say that τ_i is not schedulable; whilst when $R_{i,\tilde{v}} < D_i - J_i$ then τ_i is schedulable.

6 Analysis of MF Tasks with Arbitrary Deadliness

In this section, we discuss the criticality issues of the non-AM multiframe task when the deadline of the MF task becomes arbitrary. Then we extend the response time analysis of the non-AM multiframe tasks to be applicable to the arbitrary deadlines scenario.

The first issue to study in this scenario is the critical instant of the MF task; where we always present it as the instant that leads to the worst case response time. When no interference from the task itself is allowed, the critical instant of a MF task τ_i is the simultaneous release of its peak frame with the critical frames of the higher priority MF tasks that lead to the worst case response time of τ_i. However, when the deadline of a MF task extends beyond its period, there has been a possibility of having interference from the task itself within its busy period as well as the interference from the higher priority MF tasks; assuming that a busy period of a frame of a MF task is from when this frame starts its execution until finishing this execution. So, arbitrary deadlines may lead to the situation of analysing all critical frames of the analysed MF task instead of analysing only its peak frame. In other words, the critical instant of a non-AM multiframe task τ_i within arbitrary deadlines is the simultaneous release, that leads to the worst case response time of τ_i, of the critical frames of both τ_i and all higher priority MF tasks.

The second issue is to analyse the interference from the

task itself which requires the redefinition of $\hat{V}_i$ that was given in Section 4 to include the analysed MF task τ_i as well as the MF tasks whose priorities are higher than τ_i's. So, $\hat{V}_i$ now represents the cartesian product of all $\hat{L}_j$ for $j = 1, .., i$. Therefore, the response time of τ_i has to be analysed for each combination, that is presented as a vector in $\hat{V}_i$, of the critical frames of τ_i and higher priority MF tasks.

To explain in more details, the worst case response time of τ_i in the arbitrary deadlines scenario is analysed as a double maximization in two steps. The first step is to find, for each combination in $\hat{V}_i$, the busy periods that are relative to the possible number of interference from τ_i and then maximize those busy periods over all possible number of interference. The second step is to maximize the results in the first step, for each combination in $\hat{V}_i$ over all those combinations in $\hat{V}_i$.

Now, to completely analyse the interference from the analysed task τ_i itself, we introduce q as the number of execution of τ_i ($q = 1, 2, ..$), so the amount of execution that τ_i provides for q number of its execution is presented by its cumulative function $\xi_i^{\tilde{v}_i}(q)$; where τ_i is released with the critical frame $\tilde{v}_i$.

Therefore, as an illustration of the first step of the analysis we present $r_{i,\tilde{v}}(q)$, for $\tilde{v} \in \hat{V}_i$, as the time from the first execution of τ_i until its qth execution. $r_{i,\tilde{v}}(q)$is given by Equation (13).

$$r_{i,\tilde{v}}(q) = \xi_i^{\tilde{v}_i}(q) + \sum_{j=1}^{i-1} \xi_j^{\tilde{v}_j}(\lceil \frac{r_{i,\tilde{v}}(q)}{T_j} \rceil). \qquad (13)$$

So, the qth busy period of τ_i is given by the following Equation $w_{i,\tilde{v}}(q) = r_{i,\tilde{v}}(q) - (q-1)T_i$.. We keep finding busy periods until τ_i stops interfering with itself. In other words, above iteration over increasing values of q can stop when $w_{i,\tilde{v}}(q) \leq qT_i$. This is because satisfying the restriction means that τ_i has finished its execution within the period that τ_i is released in; and no further interference from τ_i itself will occur.

As we have already identified all busy periods we need for the analysis, we now complete the first step of the response time analysis by finding their maximum busy period that represents the worst case busy period $w_{i,\tilde{v}}$ that is relative to the combination $\tilde{v} \in \hat{V}_i$.

$$w_{i,\tilde{v}} = \max_{q=1,2,..} w_{i,\tilde{v}}(q). \qquad (14)$$

Therefore, the second step, which is the last step of the analysis, is to find the worst case response time of τ_i by maximizing $w_{i,\tilde{v}}$ over all possible combinations of the critical frames that are presented by $\tilde{v}$ as in the following equation (i.e. Equation (15))

$$R_i = \max_{\tilde{v} \in \hat{V}} \{w_{i,\tilde{v}}\} \qquad (15)$$

Thus, the schedulability test of τ_i within the arbitrary deadlines scenario is as follows: τ_i is schedulable if its worst case response time, that is calculated by Equation (15), is less than or equals to its deadline (i.e. $R_i \leq D_i$).

Example

Assume a system with three independent tasks τ_1, τ_2 and τ_3 with the parameters given in Table 10. To identify the schedulability status of τ_3, we find its worst case response time. So, we need to evaluate the response time over all possible critical frames of the MF tasks τ_1, τ_2 and τ_3.

$task$	C	T	D	priority
τ_1	5, 3, 4, 6, 8, 7	10	10	1
τ_2	6, 10, 7, 5	40	40	2
τ_3	6, 7, 8	50	60	3

Table 10. Attributes of tasks in the system

Using analysis in Section 4, locations of the critical frames $\hat{L}_j$; $j = 1, ..3$ are found as follows $\hat{L}_1 = \{2, 3, 4\}$, $\hat{L}_2 = \{0, 1\}$ and $\hat{L}_3 = \{1, 2\}$. So, the cartesian product $\hat{V}_3$ of $\hat{L}_j$; $j = 1, ..3$ is found by $\hat{V}_3 = \{(2,0,1), (2,0,2), (2,1,1), (2,1,2), (3,0,1), (3,0,2), (3,1,1), (3,1,2), (4,0,1), (4,0,2), (4,1,1), (4,1,2)\}$.
Now, we apply the analysis in this section for each $\tilde{v} \in \hat{V}_3$ on two steps. The first step is to find $w_{3,\tilde{v}}$ by applying Equations (13) and (14). So, $r_{3,(2,0,1)}(q) = \xi_3^1(q) + \sum_{j=1}^{2} \xi_j^{\tilde{v}_j}(\lceil \frac{r_{3,(2,0,1)}(q)}{T_j} \rceil)$
$q = 1$, then $r_{3,(2,0,1)}(1) = 7 + \sum_{j=1}^{2} \xi_j^{\tilde{v}_j}(\lceil \frac{r_{3,(2,0,1)}(1)}{T_j} \rceil)$.
By solving this iterative equation, we find that $r_{3,(2,0,1)}(1) = 38$. So, $w_{3,(2,0,1)}(1) = 38 - 0(50) = 38 \leq T_3$. Therefore, no need to increase q's values any more. Thus, $w_{3,(2,0,1)} = 38$.
Similarly, we find $r_{3,(2,0,2)}(1) = 8 + \sum_{j=1}^{2} \xi_j^{\tilde{v}_j}(\lceil \frac{r_{3,(2,0,2)}(1)}{T_j} \rceil) = 39$. So, $w_{3,(2,0,2)}(1) = 39$. In the same way, we find all $w_{3,\tilde{v}}$ using $r_{3,\tilde{v}}(q)$ to get the results in Table 11. Where we calculate all possible worst case busy periods that are relative to all critical frames of τ_3 and higher priority MF tasks. Note from Table 11 that values of q increases to 2 for the combinations $(2, 1, 1)$, $(2, 1, 2)$, and $(4, 1, 2)$ as the relative $w_{3,\tilde{v}}(1)$ is greater than T_3.

The second step is to find the worst case response time of τ_3, R_3, as the maximum of all possible busy periods (i.e. $w_{3,\tilde{v}}$'s column in Table 11). Thus, $R_3 = max\{38, 39, 57, 58, 39, 40, 46, 47, 36, 37, 40, 58\} = 58 < D_3$, so τ_3 is schedulable.

$\tilde{v}$	q	$r_{3,\tilde{v}}(q)$	$w_{3,\tilde{v}}(q)$	$w_{3,\tilde{v}}$
$(2,0,1)$	1	38	38	38
$(2,0,2)$	1	39	39	39
$(2,1,1)$	1	57	$57 > T_3$	
$(2,1,1)$	2	69	19	$max\{57,19\} = 57$
$(2,1,2)$	1	58	$58 > T_3$	
$(2,1,2)$	2	68	18	$max\{58,18\} = 58$
$(3,0,1)$	1	39	39	39
$(3,0,2)$	1	40	40	40
$(3,1,1)$	1	46	46	46
$(3,1,2)$	1	47	47	47
$(4,0,1)$	1	36	36	36
$(4,0,2)$	1	37	37	37
$(4,1,1)$	1	40	40	40
$(4,1,2)$	1	58	$58 > T_3$	
$(4,1,2)$	2	79	29	$max\{58,29\} = 58$

Table 11. Possible Busy Periods

7 Conclusion

In this paper we have addressed the exact response time schedulability analysis of Non-AM multiframe tasks for fixed priority preemptive systems considering a defined concept of the critical frame. Also, we have deduced the concept of dominated frames and show that such frames are non-critical, they can be safely discarded when computing worst-case response times of lower priority tasks. This provides an improvement in the response time scheduling in terms of having a maximum of $(n-1)$ critical frames, in the worst case, for a MF task of n frames; and for most MF tasks they will have significantly less than $(n-1)$ critical frames. So, the upper bound of the number of combinations that need to be examined to determine the worst-case response time is "$\prod_i (n_i - 1)$" in the worst case, which is lower than was previously published, and can be less when the dominated frames are ignored. Furthermore, we have shown that frames with the minimum execution time are always non-critical.

In addition, we have extended the exact response time analysis of the MF tasks in two directions; one is to include the MF tasks subjected to release jitter, and the other is to include tasks with arbitrary deadlines. Although the two extensions of release jitter and arbitrary deadliness can be composed, due to space restrictions the composed analysis is not presented in this paper.

Acknowledgment

With many thanks and regards to Rob Davis who helped in the early stages of this paper and to Damascus University who sponsored Ms Zuhily during her research.

References

[1] N. C. Audsley, A. Burns, M. Richardson, K. W. Tindell, and A. J. Wellings. Applying New Scheduling Theory to Static Priority Preemptive Scheduling. *Software Engineering Journal*, 8(5):284–292, September 1993.

[2] S. K. Baruah, D. Chen, and A. Mok. Static-priority scheduling of multiframe tasks. In *proceedings* 11th *Euromicro Conference on Real-Time Systems*, pages 38–45, June 1999.

[3] A. Burns, A. J. Wellings, C. Forsyth, and C. Bailey. A performance analysis of a hard real-time system. *Control Engineering Practice*, 3(4):447–464, 1995.

[4] C. J. Han. A better polynomial-time schedulability test for real-time multiframe tasks. In *proceedings of 19th IEEE Real-Time Systems Symposium*, pages 104–113, December 1998.

[5] M. Joseph. *Real Time Systems Specification, Verification and Analysis*. Prentice Hall Inc., 1st edition, 1996.

[6] M. Joseph and P. Pandya. Finding Response Times in a Real-Time system. *The Computer Journal*, 29(5):390–395, October 1986.

[7] T. Kuo, L. Chang, Y. Liu, and K. Lin. Efficient on-line schedulability tests for real-time systems. *IEEE Trans. on Software Engineering*, 29(8), 2003.

[8] C. L. Liu and J. W. Layland. Scheduling Algorithm for Multiprogrammimg in a Hard Real-Time Environment. *Journal of the Association for Computing Machinery*, 20(1):46–61, January 1973.

[9] W. C. Lu, K. J. Lin, H. W. Wei, and W. K. Shih. New schedulability conditions for real-time multiframe tasks. In *19th Euromicro Conference on Real Time Systems, (ECRTS07)*, Pisa, Italy, July 4-6 2007.

[10] A. K. Mok and D. Chen. A multiframe model for real time tasks. In *proceedings of IEEE International Real Time System Symposium*, pages 22–29, December 1996.

[11] A. K. Mok and D. Chen. A multiframe model for real-time tasks. *IEEE Trans. on Software Engineering*, 23(10):635–645, Oct 1997.

[12] O. Serlin. Scheduling of time critical processes. In *AFIPS Spring Computing Conference*, 1972.

[13] H. Takada and K. Sakamura. Schedulability of generalized multiframe task sets under static priority assignment. In *Real Time Computing Systems and Applications*, pages 80–86, 1997.

[14] K. Traor, E. Grolleau, A. Rahni, and M. Richard. Response-time analysis of tasks with offsets. In *11th IEEE International Conference on Emerging Technologies and Factory Automation, (ETFA 2006)*, Prague, Czech Republic, September 2006.

[15] A. Zuhily. Exact response time analysis for multiframe tasks. Technical Report YCS 410, University ofYork, 2007.

[16] A. Zuhily and A. Burns. Exact response time scheduling analysis of accumulatively monotonic multiframe real time tasks. In *5th International Colloquium on Theoretical Aspects of Computing (ICTAC)*, pages 410–424, 2008.