Converting existing analysis to the EDP resource model

Converting existing analysis to the EDP resource model*

Okwudire, C.G.U.

Technische Universiteit Eindhoven (TU/e),
Den Dolech 2, 5612 AZ Eindhoven,
The Netherlands
c.g.u.okwudire@student.tue.nl

Bril, R.J.

Technische Universiteit Eindhoven (TU/e),
Den Dolech 2, 5612 AZ Eindhoven,
The Netherlands
r.j.bril@tue.nl

Abstract

In (hard) real-time embedded systems, it is necessary to guarantee that tasks always meet their deadlines i.e. results should neither be too early nor too late. In the context of fixed-priority systems, this is usually done by performing schedulability analysis in which the (best-case and) worst-case response-time of each task is computed and compared with its (best-case) worst-case deadline to determine schedulability. Resource reservation has been proposed as a means to provide temporal isolation between applications. Building upon this notion, hierarchical scheduling frameworks for different resource models have been proffered in the literature with complementary schedulability conditions. Unfortunately, these novel ideas do not directly allow for the reuse of existing results, but rather favor derivations from first principles. In this document, we investigate a means to reuse existing results from non-hierarchical scheduling theory by modeling the unavailability of a resource in a two-level hierarchical framework using two fictive tasks with highest priorities. We show that this novel method using our *unavailability model* not only allows for unifying the analysis but can also be easily applied in determining linear response-time upper bounds. For the latter, we also consider approaches for obtaining tighter bounds for harmonic tasks.

Acknowledgements

*This document recapitulates and builds on the results presented in [6].

Contents

List of Figures 4

List of Tables 5

1 Introduction 6
- 1.1 Context and motivation . . . 6
- 1.2 Problem statement . . . 7
- 1.3 Approach . . . 7
- 1.4 Contributions . . . 7
- 1.5 Organization of the document . . . 8

2 Real-time scheduling models 8
- 2.1 A basic model for FPPS . . . 8
- 2.2 A periodic server model for budgets . . . 9

3 Recapitulation of analysis for FPPS 10
- 3.1 Worst-case response-time analysis . . . 10
- 3.2 Best-case response-time analysis . . . 11
- 3.3 Jitter analysis . . . 12

4 Recapitulation of analysis for two-level H-FPPS 14
- 4.1 Worst-case response-time analysis of tasks . . . 15
- 4.2 Worst-case available capacity analysis . . . 16
- 4.3 Periodic resource model . . . 17
- 4.4 Explicit-deadline periodic (EDP) resource model . . . 17
- 4.5 Other models . . . 18
- 4.6 Best-case response-time analysis of tasks . . . 19
- 4.7 Best-case available capacity analysis . . . 19

5 Response-time analysis by modeling resource unavailability 20
- 5.1 Modeling unavailability of a budget . . . 21
- 5.2 Assumptions for resource provisioning to $\mathcal{A}_\alpha$ . . . 22
- 5.3 Worst-case response-time analysis . . . 22
- 5.4 Best-case response-time analysis . . . 22
- 5.5 Applying the response-time analysis to an example . . . 23

6 Response-time upper bounds 24
- 6.1 Existing analysis for FPPS . . . 25
- 6.2 Applying the existing analysis to H-FPPS using the linear supply bound function $lsbf_\Omega(t)$ . . . 26
- 6.3 An alternative approach based on unavailability model . . . 27

7 Improving response-time upper bounds 30
- 7.1 Improved closed-form (worst-case) response-time upper bounds for tasks having the same period and scheduled on a shared resource . . . 32
  - 7.1.1 Tangent of combination approach $\perp$ $(\sum_{FPPS})$ . . . 32
  - 7.1.2 Comparison of results . . . 33
- 7.2 Improved closed-form (worst-case) response-time upper bounds for harmonic tasks scheduled on a shared FPPS resource . . . 34
  - 7.2.1 Tangent of combination approach $\perp$ $(\sum_{FPPS})$ . . . 35
  - 7.2.2 Comparison of results . . . 36

8 Conclusions 36

9 Future work **37**

Acknowledgements **37**

References **38**

A Improved and/or faster schedulability analysis using the unavailability model **41**

B Derivation of equations **44**
B.1 Worst- and best-case response-time analysis using unavailability model 44
B.2 Derivation of (linear) response-time upper bounds using $R_{i,\alpha}^{UB}(\Omega)$ 46
B.3 Derivation of (linear) response-time upper bounds using $R_i^{UB\dagger}(C)$ 47
B.4 Derivation of (linear) response-time upper bounds for tasks having the same period (Application $\mathcal{A}_4$) 47
B.4.1 Sum of tangents $\sum(\perp)$ approach 47
B.4.2 Tangent of combination$\perp(\sum_{FPPS})$ approach 48
B.5 Derivation of (linear) response-time upper bounds for harmonic tasks (Application $\mathcal{A}_5$) 48
B.5.1 Sum of tangents $\sum(\perp)$ approach 48
B.5.2 Tangent of combination$\perp(\sum_{FPPS})$ approach 49

List of Figures

1 Timelines showing FPPS worst- and best-case response-time analysis by construction. 11
2 Timelines showing FPPS worst- and best-case response-time analysis with activation jitter. 14
3 Worst-case assumptions for available capacity analysis of a budget for different resource models. 16
4 Worst-case available capacity $WC^{\beta}(t)$ for periodic and EDP resource models. . . . 18
5 Best-case assumptions and the corresponding best-case available capacity $BC^{\beta}(t)$ for the EDP resource model. 20
6 Worst- and best-case supply of an EDP resource (by construction) using the unavailability model. 21
7 Timelines comparing worst-case response-time analysis of a task (a) on a shared EDP resource and (b) using the unavailability model. 24
8 Timelines comparing best-case response-time analysis of a task (a) on a shared EDP resource and (b) using the unavailability model. 24
9 Worst-case interference $I_j^O(t)$ of task τ_j and the corresponding upper bound $I_j^{UB}(t)$ assuming it is the only task in the system. 25
10 Worst-case interference $I_\dagger^O(t)$ due to the combined fictive task $\tau_\dagger$ and the corresponding upper bound $I_\dagger^{UB}(t)$. 28
11 Timeline for a critical instant of all tasks of application $\mathcal{A}_3$. 31
12 Timeline for a critical instant of all tasks of application $\mathcal{A}_4$. 32
13 Improving response-time upper bounds of lower priority tasks by combining same-period (higher priority) tasks. 33
14 Improving response-time upper bounds of lower priority tasks by combining harmonic-period (higher priority) tasks. 35

List of Tables

1 Task characteristics of $\mathcal{T}_1$ with their worst-case and best-case response-times. . . . 10
2 Characteristics of fictive tasks τ_{-1} and τ_0. 21
3 Characteristics of $\mathcal{T}_2^{\beta}$ (of application $\mathcal{A}_2$) with worst-case and best-case response-times of tasks. 23
4 Characteristics of β_2 provided to application $\mathcal{A}_2$. 23
5 (Upper bounds of) worst-case response-times for tasks $\tau_{1(,2)}$ and $\tau_{2(,2)}$ in application $\mathcal{A}_2$ ($\widehat{\mathcal{A}_2}$). 27
6 (Upper bounds of) worst-case response-times for tasks $\tau_{1(,3)}$, $\tau_{2(,3)}$ and $\tau_{3(,3)}$ in application $\mathcal{A}_3$ ($\widehat{\mathcal{A}_3}$). 30
7 (Upper bounds of) worst-case response-times for tasks τ_1, τ_2 and τ_3 in application $\mathcal{A}_4$. 32
8 (Upper bounds of) worst-case response-times for tasks τ_1, τ_2 and τ_3 in application $\mathcal{A}_5$. 34
9 (Upper bounds of) worst-case response-times for tasks τ_1 and τ_2 in application $\mathcal{A}_2$ ($\widehat{\mathcal{A}_2}$) based on proposed extension. 41
10 Characteristics of $\widehat{\mathcal{T}_2^{\beta}}$ with worst-case and best-case response-times of tasks. 44

1 Introduction

1.1 Context and motivation

Following the seminal work of Liu and Layland [27], many results have been achieved in the area of fixed-priority preemptive scheduling (FPPS) of real-time tasks, producing a considerable body of real-time analysis for scheduling of hard real-time tasks on a single, shared processor. Although, fixed-priority pre-emptive scheduling (FPPS) has been widely accepted and is currently the de-facto standard in industry for scheduling system with real-time constraints, this scheduling paradigm has the major drawback that temporary or permanent faults occurring in one application can hamper the execution of other applications. Such a situation is undesirable, particularly in hard real-time systems where failure of tasks in an application to meet their deadlines may have catastrophic consequences such as loss of life.

To address this problem, the notion of *resource reservation* has been proposed [30]. Resource reservation provides *isolation* between applications, effectively guaranteeing temporal protection of an application against other malfunctioning applications. With applications consisting of one or more real-time tasks, resource reservation inherently requires multiple levels of scheduling, that is, a scheduling hierarchy [36]. In this hierarchical scheduling framework (HSF), we consider a set of independent applications that are executed on a shared resource, where each application consists of a set of independent, periodically released, hard real-time tasks. Furthermore, temporal isolation is achieved by allocation a dedicated budget to each application. In this document, we consider two-level hierarchical scheduling, with FPPS for tasks and budgets.

Analytic methods for hierarchical scheduling are a topic of current research [1, 10, 14, 26, 38, 31]. The periodic resource models proposed in [38], [16] to characterize periodic resource allocations to applications in the context of HSFs are complemented with *novel* schedulability conditions and methods to abstract timing requirements in the hierarchy of schedulers. However, given the significant amount of existing work on task scheduling on a single shared processor, it is desirable, if possible, to re-use existing results and avoid reinventing the wheel. Unfortunately, these novel conditions hamper reuse of existing schedulability results and encourage re-invention of results, with the risk of introducing errors. As an example, a recent paper [41] showed that the utilization bound for a periodic resource model under rate monotonic (RM) scheduling presented in [38] is optimistic. Such inadvertent errors will obviously be avoided if we can adapt the schedulability analysis of HSFs to the existing, thoroughly scrutinized analysis for single-level shared resources. As further motivation for the proposed approach of reuse, we remark that the same paper showed that by viewing the unavailability of the periodic resource as a deferrable server at highest priority, existing utilization bounds for systems with a deferrable server [23], [39] can be reused. Based on derived analysis for sporadic servers in the context of HSFs, it was already observed in [36] that the unavailability of a resource can be modeled as a fictive task. However, that view was not exploited to convert and unify schedulability analysis. Instead, the utilization bound presented in that paper is derived from first principles.

Although worst-case response-time analysis for a shared resource using FPPS has been addressed extensively in the literature, and many restrictions of the original scheduling model [27] have been lifted in subsequent work (e.g. [2, 32, 25, 24, 37, 40, 35, 17, 42, 18, 29], amongst others), the corresponding scheduling models for best-case response-time analysis [9, 19, 34, 7] considerably fall behind. The divergence for two-level hierarchical scheduling is even more pronounced given that, except for [28], we know of no other work which has addressed best-case response-time schedulability analysis for any of the resource models we consider notably, the periodic [38] and explicit-deadline periodic (EDP) [16] resource models. Ironically, the notion of *timely response* in real-time systems typically translates to both (best-case) *upper* bounds and (worst-case) *lower* bounds, i.e. tasks are required to be neither too early nor too late. In [7], Bril et al. identify some examples where best-case response-time analysis of tasks becomes particularly important

due to explicit requirements on response-time lower bounds. In an attempt to extend the existing best-case schedulability analysis, they present a conjecture for exact best-case response-times of periodically released, independent real-time tasks with *arbitrary deadlines* that are scheduled by means of FPPS.

By successfully reusing existing results for worst-case response-time analysis on a shared resource, we expect to be able to apply the same idea in performing best-case response analysis for tasks scheduled on a shared EDP resource, thereby making a novel contribution in the area of hierarchical schedulability analysis.

1.2 Problem statement

Against the backdrop presented in Section 1.1 above, we, therefore, aim at developing a framework which will enable the reuse of existing analysis for HSFs. In particular, we propose to unify the existing schedulability analysis for independent applications scheduled using FPPS on a single (shared) processor and the schedulability analysis for a corresponding two-level hierarchical framework in which the independent applications share a periodic resource (i.e. *virtual* processor) according to the EDP resource model [16] with FPPS.

1.3 Approach

Our approach to solving this problem will be to model the unavailability of the EDP resource using two *fictive tasks* executing at highest priority. We consolidate the work presented in [6] where the method was already outlined. Using sample task sets, we will show that this rather straightforward transformation reduces the problem to single-level FPPS for which we can then directly apply existing results. We further illustrate our method by applying it to the derivation of linear response-time upper bounds for FPPS [4, 15].

1.4 Contributions

The main contributions of this work are as follows:

First, we propose a method for converting existing analysis to the EDP model by viewing the *unavailability* of the budget as two *fictive, highest priority tasks.* By doing so, the analytic results for tasks under FPPS can be directly applied to tasks of independent applications under two-level hierarchical FPPS frameworks (H-FPPS). We provide equations for calculating the worst-case and best-case response-times of tasks in this model taking into account activation jitter and (specific) phasing of the fictive tasks relative to the defined critical/optimal instants. We show by means of an example that our approach not only yields accurate results (i.e. identical to results obtained using the EDP model directly) but also simplifies the analysis since no auxiliary (inverse) functions are needed and the classical FPPS equations for best- and worst-case analysis can be directly reused.

To further illustrate the applicability and simplicity of our approach, we determine (linear) response-time upper bounds of tasks in a two-level HSF by first converting the task set to an FPPS shared resource model and subsequently applying the approach in [15]. We prove that, again, the results obtained by our method are correct and coincide with those obtained using the *linear supply lower bound function* $lsbf_\Omega(t)$. Thus, our unavailability model serves as an alternative approach for solving the problem with no significant overhead.

Finally, we show that for higher priority tasks having the same or harmonic periods, response-time upper bounds of the *lower priority tasks* can be improved by first summing the interference due to these higher priority tasks before taking their (linear) demand upper bound. Subsequently, we

derive a closed-form response-time upper bound for this approach and show that it can, but not necessarily always will, improve on the existing bound presented in [15].

1.5 Organization of the document

The rest of this document is organized as follows. We begin with an overview of a real-time scheduling model for FPPS in Section 2. Next, we recapitulate the analysis of tasks under FPPS in Section 3. In Section 4, we review existing approaches for two-level H-FPPS with emphasis on the periodic and EDP resource models. We also present the notions of worst- and best-case available capacity which we use in the response-time analysis of *tasks* (schedulability analysis of *servers* is not elaborated in this document). Following this, we present our unavailability model for two-level H-FPPS in Section 5 including analyses of worst- and best-case response-times. In Section 6 we extend the linear response-time upper bounds of tasks under FPPS with deadlines less than or equal to periods presented in [15] and show how this analysis can be applied to two-level H-FPPS using the model proposed in Section 5. In Section 7, we consider a special subset of tasks namely tasks having harmonic periods for which we show how tighter response-time upper bounds can be obtained without hampering the linear time upper bound. Finally, we conclude the document and indicate possible directions for future work in Sections 8 and **??** respectively.

2 Real-time scheduling models

2.1 A basic model for FPPS

We assume a single processor and a set $\mathcal{T}$ of n periodically released, independent tasks $\tau_1, \tau_2, \ldots, \tau_n$ with unique, fixed priorities. At any moment in time, the processor executes the highest priority task that has work pending.

Each task τ_i is characterized by a (*release* or *activation*) *period* $T_i \in \mathbb{R}^+$, a *worst-case computation time* $WC_i \in \mathbb{R}^+$, a *best-case computation time* $BC_i \in \mathbb{R}^+$, where $BC_i \leq WC_i$, a *phasing* $\varphi_i \in \mathbb{R}$, a (*relative*) *worst-case deadline* $WD_i \in \mathbb{R}^+$, and a (*relative*) *best-case deadline* $BD_i \in \mathbb{R}^+ \cup \{0\}$, where $BD_i \leq WD_i$. The set of phasings φ_i is termed the phasing φ of the task set $\mathcal{T}$ i.e. $\varphi = \{\varphi_1, \varphi_2, \ldots, \varphi_n\}$. Also, the deadlines BD_i and WD_i are relative to the activations. For ease of presentation, we assume, in all the examples, that the worst-case and best-case computation times are identical, i.e. $WC_i = BC_i$, the best-case deadline is equal to zero, i.e. $BD_i = 0$, and we simply denote the computation time and worst-case deadline as C_i and D_i respectively.

An *activation time* is a time at which a task τ_i becomes ready for execution. Each activation of a task is termed a *job* of that task. The job of task τ_i that is activated at time φ_i and referred to as job zero. The activation of job k of τ_i therefore takes place at time $a_{ik} = \varphi_i + kT_i$, $k \in \mathbb{Z}$. The (*absolute*) *deadline* of job k of τ_i takes place at $d_{ik} = a_{ik} + D_i$. The *finalization time* f_{ik} of job k of τ_i is the time at which τ_i ends the execution of that job. The *response-time* R_{ik} of job k of τ_i is defined as the length of the time span between the activation time of that job and its finalization time, i.e. $R_{ik} = f_{ik} - a_{ik}$.

We assume that we do not have control over the phasing φ, for instance since the tasks are released by external events; so we assume that any arbitrary phasing may occur. This assumption is common in real-time scheduling literature [20, 21, 27]. We also assume other standard basic assumptions [27], i.e. tasks are ready to run at the start of each period and do not suspend themselves, tasks will be preempted instantaneously when a higher priority task becomes ready to run, a job of task τ_i does not start before its previous job is completed, and the overhead of context switching and task scheduling is ignored. Finally, we assume that the deadlines are hard meaning that each job of a task must be completed at or before its worst-case deadline and at

or after its best-case deadline. For notational convenience, we assume that the tasks are given in order of decreasing priority, i.e. task τ_1 has highest priority and task τ_n has lowest priority.

Given these definitions and assumptions, we define the following derived notions: The *worst-case response-time* WR_i and the *best-case response-time* BR_i of a task τ_i are the largest and the smallest response-time of any of its jobs under arbitrary phasing, respectively, i.e.

$$WR_i \stackrel{\text{def}}{=} \sup_{\varphi,k} R_{ik}(\varphi) \quad \text{and} \quad BR_i \stackrel{\text{def}}{=} \inf_{\varphi,k} R_{ik}(\varphi). \tag{1}$$

Note that the response-time R_{ik} has been parametrized in these equations to denote its dependency on the phasing φ. A *critical instant* [27] and an *optimal* (or *favorable*) *instant* [9, 34] of a task are defined to be (hypothetical) instants that lead to the worst-case and the best-case response-time for that task, respectively. The (*processor*) *utilization factor* $U^{\mathcal{T}}$ of a task set $\mathcal{T}$ is the fraction of the processor time spent on the execution of that task set [27]. The fraction of processor time spent on executing a periodic task τ_i with a fixed period T_i and computation time C_i is therefore C_i/T_i, and is termed the *utilization factor* U_i^{τ} of task τ_i, i.e.

$$U_i^{\tau} \stackrel{\text{def}}{=} \frac{C_i}{T_i}. \tag{2}$$

The *cumulative utilization factor* $U_i^{\mathcal{T}}$ for periodic tasks τ_1 till τ_i with fixed computation times is the fraction of processor time spent on executing these tasks, and is given by

$$U_i^{\mathcal{T}} \stackrel{\text{def}}{=} \sum_{1\leq j\leq i} U_j^{\tau} = \sum_{1\leq j\leq i} \frac{C_j}{T_j}. \tag{3}$$

Therefore, $U^{\mathcal{T}}$ is equal to the cumulative utilization factor $U_n^{\mathcal{T}}$ for the n periodic tasks comprising $\mathcal{T}$.

Because we distinguish best-case and worst-case computation times in this document, we get best-case and worst-case versions of the various notions of utilization, i.e.

$$WU^{\mathcal{T}} \stackrel{\text{def}}{=} WU_n^{\mathcal{T}} = \sum_{1\leq j\leq n} WU_j^{\tau} = \sum_{1\leq j\leq n} \frac{WC_j}{T_j} \quad \text{and} \quad BU^{\mathcal{T}} \stackrel{\text{def}}{=} BU_n^{\mathcal{T}} = \sum_{1\leq j\leq n} BU_j^{\tau} = \sum_{1\leq j\leq n} \frac{BC_j}{T_j}. \tag{4}$$

Based on the notion worst-case response-time and the assumption that deadlines are hard, we conclude that a set $\mathcal{T}$ of n periodic tasks can be scheduled if and only if

$$\bigwedge_{i=1,\ldots,n} (BD_i \leq BR_i \wedge WR_i \leq WD_i)\,. \tag{5}$$

Equation (5) represents an *exact* schedulability condition for $\mathcal{T}$. A *necessary* schedulability condition for $\mathcal{T}$ is given by

$$WU^{\mathcal{T}} \leq 1. \tag{6}$$

2.2 A periodic server model for budgets

Assuming periodic servers as implementations for budgets, a periodic budget may simply be viewed as an artificial periodic task with a fixed computation time (i.e. $WC = BC$), and a worst-case (relative) deadline equal to the period (i.e. $D = T$). Because we also assume FPPS for budgets, all notions and assumptions of our scheduling model for tasks can be reused for budgets. We remark that the fact that budgets are allocated and provided to applications (and not to individual tasks, for instance) is not important for our model for budgets.

Hence, we assume a single processor and a set $\mathcal{B}$ of n periodically released, independent budgets $\beta_1, \beta_2, \ldots, \beta_n$ with fixed, unique priorities. Note that we assume that the *capacity* of a budget (i.e. the equivalent of a computation time of a task) is fixed, hence a budget is always entirely consumed, and never discarded or suspended. Depletion of capacity can be done by idling the capacity away [14], by soft tasks of the application associated with the budget [14], or by means of 'in-the-place-of' spare-capacity provision to other applications [5]. However, since we only consider hard real-time tasks in this document, we assume that any remaining capacity is idled away [14].

We assume a one-to-one relationship between budgets and applications, i.e. budget β_α is associated with application $\mathcal{A}_\alpha$ with $1 \leq \alpha \leq m$. An application $\mathcal{A}_\alpha$ is assumed to consist of n_α tasks. A *necessary* schedulability condition for $\mathcal{A}_\alpha$ with an associated budget β_α [8] is now given by

$$WU_\alpha^{\mathcal{A}} \leq U_\alpha^{\beta}, \tag{7}$$

where $WU_\alpha^{\mathcal{A}}$ and U_α^{β} denote the (worst-case cumulative) utilization factor of $\mathcal{A}_\alpha$ and β_α, respectively.

In the remainder of this document, we assume that task sets and budget sets satisfy the necessary worst-case schedulability condition as expressed by Equation (6). Moreover, we assume that applications satisfy the necessary schedulability condition as expressed by Equation (7). Furthermore, we will distinguish tasks on a shared resource from those on a shared period resource by appending a subscript to the latter to denote the application to which a task belongs e.g. $\tau_{i,\alpha}$ denotes task i of application $\mathcal{A}_\alpha$ whose computation time, period and worst-case response-time are given by $C_{i,\alpha}$, $T_{i,\alpha}$ and $WR_{i,\alpha}$ respectively. We will use $\mathcal{T}_\alpha^\beta$ to denote the task set associated with application $\mathcal{A}_\alpha$ and scheduled on budget β_α. Moreover, we will use the superscript β to distinguish the characteristics of budgets from those of tasks e.g. C_α^β denotes the capacity of budget β_α. Also, whenever we desire to explicitly mention that application $\mathcal{A}_\alpha$ comprises n_α tasks, we employ the notation $\mathcal{A}_{\alpha,n_\alpha}$. The same also applies to notions of utilization factor.

3 Recapitulation of analysis for FPPS

In this section we recapitulate analysis of tasks under FPPS, based on [2, 9, 20, 27, 34], amongst others. We start with worst-case and best-case response-time analysis, using an example task set $\mathcal{T}_1$ with characteristics as given in Table 1 for illustration purposes. The response-time analysis is exact under arbitrary phasing, but typically pessimistic under a specific phasing as shown in [18]. We subsequently recapitulate finalization jitter and activation jitter.

task	$T = D$	C	WR	BR
τ_1	3	1	1	1
τ_2	5	2	3	2
τ_3	18	3	14	7

Table 1: Task characteristics of $\mathcal{T}_1$ with their worst-case and best-case response-times.

3.1 Worst-case response-time analysis

A *critical instant* of a task τ_i is assumed when τ_i is simultaneously released with all tasks with a higher priority than τ_i [27]. Figure 1(a) shows a timeline of $\mathcal{T}_1$ with critical instants for all tasks. As can be observed from the figure, the critical instant is the same for all the tasks and

coincides with their simultaneous release (cf. *optimal instant* in Section 3.2 which may vary per task). Because the worst-case response-times of all tasks are smaller than their deadlines, we conclude from this timeline that $\mathcal{T}_1$ is schedulable.

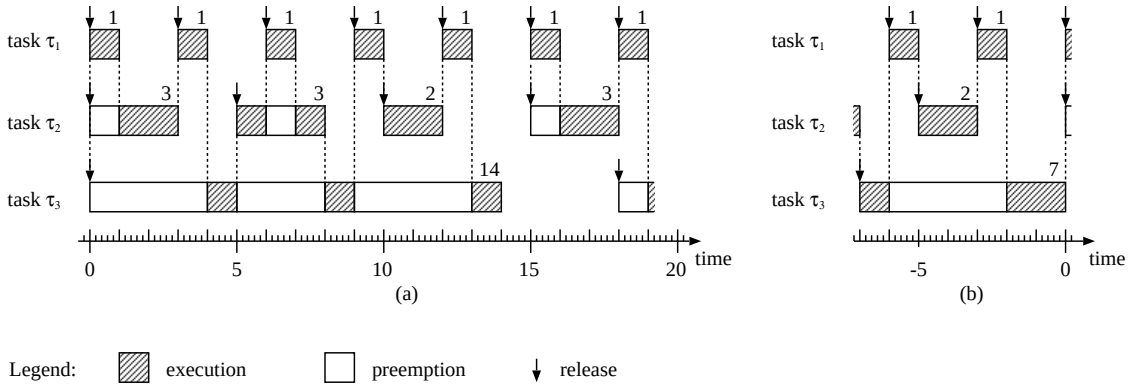

Figure 1: Timelines of $\mathcal{T}_1$ with (a) critical instants for all tasks and (b) an optimal instant for task τ_3. The numbers to the top right corner of the boxes denote the response-time of the respective jobs.

From this notion of critical instant, it has been derived in [20] that the *worst-case response-time* WR_i of task τ_i is given by the smallest value $x \in \mathbb{R}^+$ satisfying

$$x = WC_i + \sum_{1 \leq j < i} \left\lceil \frac{x}{T_j} \right\rceil WC_j. \tag{8}$$

Such a smallest value exists for task τ_i if and only if $WU_{i-1}^{\mathcal{T}} < 1$; see, for example, [5]. Because we assume $WU^{\mathcal{T}} \leq 1$ and $WC_i > 0$ for all tasks, $WU_{i-1}^{\mathcal{T}} < 1$ holds for all tasks. To calculate WR_i, we can use an iterative procedure based on recurrence relationships [2].

$$\begin{aligned} wr_i^{(0)} &= WC_i \\ wr_i^{(l+1)} &= WC_i + \sum_{1 \leq j < i} \left\lceil \frac{wr_i^{(l)}}{T_j} \right\rceil WC_j, \quad l = 0, 1, \ldots \end{aligned}$$

The procedure is stopped when the same value is found for two successive iterations or when the deadline is exceeded. In the latter case, task τ_i is not schedulable.

3.2 Best-case response-time analysis

An *optimal instant* of a task τ_i is assumed when the completion of τ_i coincides with the simultaneous release of all tasks with a higher priority than τ_i [9, 34]. Figure 1(b) shows a timeline of $\mathcal{T}_1$ with an optimal instant for task τ_3. Unlike the critical instant which was the same for all tasks, we observe the optimal instant may vary from one task to another. This follows directly from the definition of an optimal instant.

The *best-case response-time* BR_i of task τ_i is given by the largest value $x \in \mathbb{R}^+$ satisfying

$$x = BC_i + \sum_{1 \leq j < i} \left(\left\lceil \frac{x}{T_j} \right\rceil - 1 \right) BC_j. \tag{9}$$

Such a largest value exists for task τ_i if and only if $BU_{i-1}^{\mathcal{T}} < 1$; see [5]. Because $BU^{\mathcal{T}} \leq WU^{\mathcal{T}}$

by definition, and we assume $WU^{\mathcal{T}} \leq 1$ and $BC_i > 0$ for all tasks, the relation $BU^{\mathcal{T}}_{i-1} < 1$ trivially holds for all tasks. To calculate BR_i, we can use an iterative procedure based on recurrence relationships [2].

$$\begin{aligned} br_i^{(0)} &= WR_i \\ br_i^{(l+1)} &= BC_i + \sum_{1 \leq j < i} \left(\left\lceil \frac{br_i^{(l)}}{T_j} \right\rceil - 1 \right) BC_j, \quad l = 0, 1, \ldots \end{aligned}$$

The procedure is stopped when the same value is found for two successive iterations or when the deadline is exceeded. In the latter case, task τ_i is not schedulable.

3.3 Jitter analysis

The *worst-case* (*absolute*) *finalization jitter* FJ_i of a periodically released task τ_i is defined as

$$FJ_i \stackrel{\text{def}}{=} \sup_{\varphi,k,l} (f_{ik}(\varphi) - f_{il}(\varphi) - (k-l)T_i). \tag{10}$$

For a strictly periodically released task τ_i, this can be rewritten to

$$\begin{aligned} FJ_i &= \sup_{\varphi,k,l} ((f_{ik}(\varphi) - (\varphi_i + kT_i) - (f_{il}(\varphi) - (\varphi_i + lT_i))) \\ &= \sup_{\varphi,k,l} ((f_{ik}(\varphi) - a_{ik}(\varphi_i)) - (f_{il}(\varphi) - a_{il}(\varphi_i))) \\ &= \sup_{\varphi,k,l} (R_{ik}(\varphi) - R_{il}(\varphi)). \end{aligned} \tag{11}$$

Because the largest and smallest response-times are not necessarily taken for the same phasing, an upper bound on FJ_i is given by

$$FJ_i \leq \sup_{\varphi,k} R_{ik}(\varphi) - \inf_{\varphi,l} R_{il}(\varphi) = \{(1)\}\ WR_i - BR_i. \tag{12}$$

For task τ_3 of $\mathcal{T}_1$, we find $FJ_3 \leq WR_3 - BR_3 = 14 - 7 = 7$.

We note that Equation (11) would also be a sensible definition for *worst-case* (*absolute*) *response jitter* RJ_i of a periodic task τ_i i.e.

$$RJ_i \stackrel{\text{def}}{=} \sup_{\varphi,k,l} (R_{ik}(\varphi) - R_{il}(\varphi)). \tag{13}$$

We observe that such a definition of the notion of worst-case (absolute) response jitter RJ_i differs from the notions of *absolute response-time jitter* RTJ_i^{abs} in [11] and *absolute finishing jitter* AFJ_i in [12]. In [11], absolute response-time jitter has been defined as

$$\text{RTJ}_i^{abs} \stackrel{\text{def}}{=} \max_k R_{ik} - \min_k R_{ik}. \tag{14}$$

The definition of absolute finishing jitter in [12] is similar. The main difference between these two notions and our notion is that [12, 11] assume a specific phasing whereas we assume arbitrary phasing.

Next to finalization jitter, there can also be *activation* (or *release*) *jitter*. In this case, the releases of jobs of task τ_i do not take place strictly periodically, with period T_i, but we assume they take place somewhere in an interval of length AJ_i that is repeated with period T_i. More specifically, the activation times satisfy

$$\sup_{k,l}(a_{ik}(\varphi_i) - a_{il}(\varphi_i) - (k-l)T_i) \leq AJ_i, \tag{15}$$

where φ_i now denotes the start of the interval of length AJ_i in which job zero of task τ_i is activated, rather than the time at which job zero is activated, i.e. $\varphi_i + kT_i \leq a_{ik} \leq \varphi_i + AJ_i + kT_i$. We now assume $D_i \leq T_i - AJ_i$, since otherwise there may be too little time between two successive releases to complete the job. In case of activation jitter, the analysis to derive worst-case and best-case response-times, as well as finalization jitter, is slightly altered. For worst-case and best-case response-times, we extend the analysis as described in [2] and [9, 34], respectively.

A *critical instant* of a task τ_i is assumed when τ_i is simultaneously released with all tasks with a higher priority than τ_i, all those tasks with a higher priority experience a maximum release delay at that simultaneous release and a minimum release delay at subsequent releases. The *worst-case response-time* WR_i of task τ_i of $\mathcal{T}$ with activation jitter AJ_i is given by the smallest value $x \in \mathbb{R}^+$ satisfying

$$x = WC_i + \sum_{1 \leq j < i} \left\lceil \frac{x + AJ_j}{T_j} \right\rceil WC_j. \tag{16}$$

Similar to the case without activation jitter, there exists a smallest value if and only if $WU^{\mathcal{T}}_{i-1} < 1$ and the recursive equation can than be solved by means of an iterative procedure, starting with a lower bound.

An *optimal instant* of a task τ_i is assumed when the completion of τ_i coincides with the simultaneous release of all tasks in $\mathcal{T}$ with a higher priority than τ_i, all those tasks with a higher priority experience a maximal release delay at that simultaneous release and a minimal release delay at previous releases. The *best-case response-time* BR_i of task τ_i is given by the largest value $x \in \mathbb{R}^+$ satisfying

$$x = BC_i + \sum_{1 \leq j < i} \left(\left\lceil \frac{x - AJ_j}{T_j} \right\rceil - 1 \right)^+ BC_j, \tag{17}$$

where the notation w^+ stands for $\max(w, 0)$. Similar to the case without activation jitter, there exists a largest value if $WU^{\mathcal{T}} \leq 1$ and the recursive equation can than be solved by means of an iterative procedure, starting with an upper bound.

With activation jitter, we now derive from Equation (10)

$$\begin{aligned} FJ_i &= \sup_{\varphi,k,l}(f_{ik}(\varphi) - f_{il}(\varphi) - (k-l)T_i) \\ &= \sup_{\varphi,k,l}((a_{ik}(\varphi_i) + R_{ik}(\varphi)) - (a_{il}(\varphi_i) + R_{il}(\varphi)) - (k-l)T_i) \\ &= \sup_{\varphi,k,l}(a_{ik}(\varphi_i) - a_{il}(\varphi_i) - (k-l)T_i + R_{ik}(\varphi) - R_{il}(\varphi)) \\ &\leq \sup_{\varphi_i,k,l}(a_{ik}(\varphi_i) - a_{il}(\varphi_i) - (k-l)T_i) + \sup_{\varphi,k} R_{ik}(\varphi) - \inf_{\varphi,l} R_{il}(\varphi). \end{aligned}$$

Given the worst-case and best-case response-times of task τ_i including the effect of activation jitter of higher priority tasks, and given the release jitter of τ_i itself, its worst-case (absolute) finalization jitter is therefore bounded by

$$FJ_i \leq AJ_i + WR_i - BR_i. \tag{18}$$

We note that for periodically released tasks with activation jitter, finalization jitter differs from response-time jitter. We merely mention that [12, 11] do not consider activation jitter.

As an example of finalization jitter with activation jitter, consider a task set $\mathcal{T}_1'$ with the same characteristics as $\mathcal{T}_1$, except that task τ_2 now has an activation jitter $AJ_2 = 2$ and a deadline $D_2 = T_2 - AJ_2 = 5 - 2 = 3$. Since the worst-case response-time of a task is independent of its own activation jitter, the worst-case response-times of τ_1 and τ_2 are independent of the activation jitter AJ_2 of τ_2 and $WR_2 \leq D_2$, tasks τ_1 and τ_2 remain schedulable. Figure 2 illustrates timelines of $\mathcal{T}_1'$ with a critical instant and an optimal instant for τ_3. From this figure, we conclude that the finalization jitter FJ_3 of task τ_3 is bounded by $FJ_3 \leq AJ_3 + WR_3 - BR_3 = 0 + 17 - 4 = 13$.

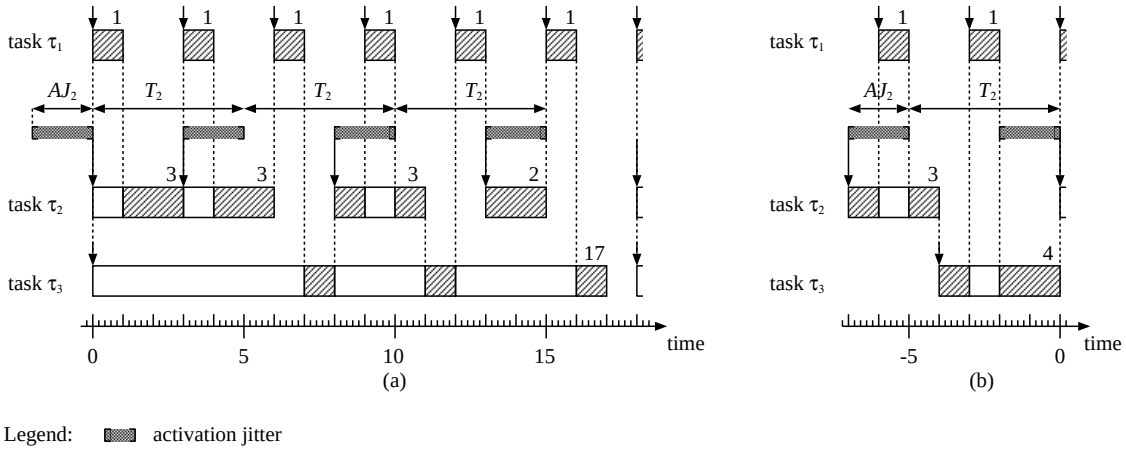

Figure 2: Timelines of $\mathcal{T}_1'$ with (a) a critical instant and (b) an optimal instant for task τ_3.

4 Recapitulation of analysis for two-level H-FPPS

In two-level HSFs, scheduling is performed at the system and application levels by the *global* and *local* schedulers respectively. Whereas the global scheduler determines which of the servers (hence, the associated application) should be provided the shared resource (i.e. processor), the local scheduler determines which task of a given application should execute once the application has secured access to the processor via the global scheduler. Although it is possible to have different scheduling policies at both levels, we assume system-wide FPPS for tasks and budgets. Nonetheless, at the application level, we adhere to the principle of *locality of scheduling analysis* which means that we make no assumptions about the scheduling policy of budgets or about the characteristics of budgets other than the budget allocated to the particular application under consideration. This principle facilitates independent design, analysis and validation of systems but comes at the cost of pessimism of the analysis. It is adhered to by [26, 38]; thus, their results also apply with FPPS for budgets.

Conversely, the analyses in [1, 10, 14, 36] build upon the assumption of FPPS for budgets. Moreover, [1, 10, 14] assume arbitrary phasing of budgets, and assume that the other characteristics of budgets are known, as well as their priorities. Although these assumptions allow for a reduction of the pessimism in the analysis, this reduction comes at the cost of an increase in the complexity of the analysis, whilst the analysis still remains pessimistic [10, 3].

In this section, we briefly discuss how the worst-case response-time of a task in an application can be determined using the notion of worst-case (i.e. minimum) available budget capacity introduced

in Section 1.4. Next, we review the definition of this notion in different resource models presented in the literature. We conclude the section by considering best-case response-time analysis of tasks using the EDP resource model. We remark that this discussion focuses on scheduling of tasks (i.e. local level) and does not address scheduling of budgets (i.e. global level). However, if FPPS is employed at the global level, the analysis presented in Section 3 can be directly reused by viewing the budgets themselves as 'tasks' on a dedicated resource. Likewise, the same 'trick' can be applied if a different scheduling algorithm is used at the global level i.e. the response-time analysis for that algorithm assuming a dedicated resource can be directly applied to determine schedulability of budgets provided locality of scheduling analysis is obeyed at the local level.

4.1 Worst-case response-time analysis of tasks

Equation (8) for the worst-case response-time of a task τ_i of a set $\mathcal{T}$ assumes that the entire processor is available to $\mathcal{T}$. However, when only the capacity of a budget β_α is available to the tasks of an application $\mathcal{A}_\alpha$, we can simply replace the x at the left hand side of Equation (8) by the *worst-case (i.e. minimum) available capacity* $WC_\alpha^\beta(x)$, as explained in [1]. This notion is further explained in Section 4.2. The worst-case response-time $WR_{i,\alpha}$ of a task $\tau_{i,\alpha}$ of application $\mathcal{A}_\alpha$ with associated budget β_α is thus given by the smallest $x \in \mathbb{R}^+$ satisfying

$$WC_\alpha^\beta(x) = WC_{i,\alpha} + \sum_{1 \le j < i} \left\lceil \frac{x}{T_{j,\alpha}} \right\rceil WC_{j,\alpha}. \tag{19}$$

For the approaches considered in subsequent sections (4.3 to 4.5), such a smallest value exists for $\tau_{i,\alpha}$ when $WU_{i-1,\alpha}^{\mathcal{A}} < U_\alpha^\beta$. Because we assume $WU_\alpha^{\mathcal{A}} \le U_\alpha^\beta$, $WU_{i-1,\alpha}^{\mathcal{A}} < U_\alpha^\beta$ holds for all tasks of $\mathcal{A}_\alpha$.

A *critical instant* of a task $\tau_{i,\alpha}$ is assumed when (1) $\tau_{i,\alpha}$ experiences a maximum interference of higher priority tasks, i.e. (1a) $\tau_{i,\alpha}$ is simultaneously released with all tasks in $\mathcal{A}_\alpha$ having a higher priority than $\tau_{i,\alpha}$ and (1b) all those higher priority tasks experience a maximum release delay at the simultaneous release and a minimum release delay at subsequent releases; and (2) the simultaneous release coincides with the start of an interval with a minimum supply of resource β_α.

To determine $WR_{i,\alpha}$, an auxiliary function $\overline{WC_\alpha^\beta}(y)$ is introduced, which is defined as

$$\overline{WC_\alpha^\beta}(y) = \min\{x \mid WC_\alpha^\beta(x) = y\}. \tag{20}$$

This auxiliary function, is described as the 'inverse of the availability function' $A_s^{inv}(t)$ in [1] and referred to as *service time bound function* $\mathbf{tbf}_\Gamma(t)$ in [38]. The worst-case response-time $WR_{i,\alpha}$ is now given by the smallest $x \in \mathbb{R}^+$ satisfying

$$x = \overline{WC_\alpha^\beta}\left(WC_{i,\alpha} + \sum_{1 \le j < i} \left\lceil \frac{x}{T_j} \right\rceil WC_{j,\alpha} \right). \tag{21}$$

To calculate $WR_{i,\alpha}^\tau$, we can again use an iterative procedure based on recurrence relationships.

$$\begin{aligned} wr_{i,\alpha}^{(0)} &= WC_{i,\alpha} \\ wr_{i,\alpha}^{(l+1)} &= \overline{WC_\alpha^\beta}\left(WC_{i,\alpha} + \sum_{1 \le j < i} \left\lceil \frac{wr_i^{(l)}}{T_{j,\alpha}} \right\rceil WC_{j,\alpha} \right), \quad l = 0, 1, \ldots \end{aligned}$$

The procedure is stopped when the same value is found for two successive iterations or when the deadline is exceeded. In the latter case, task $\tau_{i,\alpha}$ is not schedulable.

4.2 Worst-case available capacity analysis

As seen in Equation (19), in order to determine the worst-case response-time of tasks, we need the *worst-case available capacity* $WC^{\beta}(t)$ that becomes available in an interval of length t from a budget β. The term worst-case available capacity originates from [5] and is also used in [10]. Other terms used in the literature for this notion are *least supply function* $S^*(t)$ [31], *characteristic function* $Z_S(t)$ of a periodic server S [26], (*resource*) *supply bound function* $sbf_\Gamma(t)$ for a periodic resource Γ [38], *(resource) supply bound function* $sbf_\Omega(t)$ for an EDP resource Ω [16], and *availability function* $A\_S(t)$ of a server S [1]. We identify four classes of worst-case assumptions that are made in the literature for this situation and illustrate them by means of an example set $\mathcal{B}_1$ of three budgets. For ease of presentation, these budget characteristics (i.e. period, capacity and for the EDP resource model, deadline) are chosen to be identical to the characteristics of our example set $\mathcal{T}_1$ of tasks as shown in Table 1.

For $t > 0$, we may assume without loss of generality that the interval has an overlap with at least two periods of the budget β [10]. A minimum amount of capacity becomes available in an interval of length $t > 0$ when the capacity becomes available *as early as possible* in a first period of β overlapping with the interval and *as late as possible* in the last overlapping period. The definition of this *worst-case scenario* distinguishes the models discussed in the subsequent subsections.

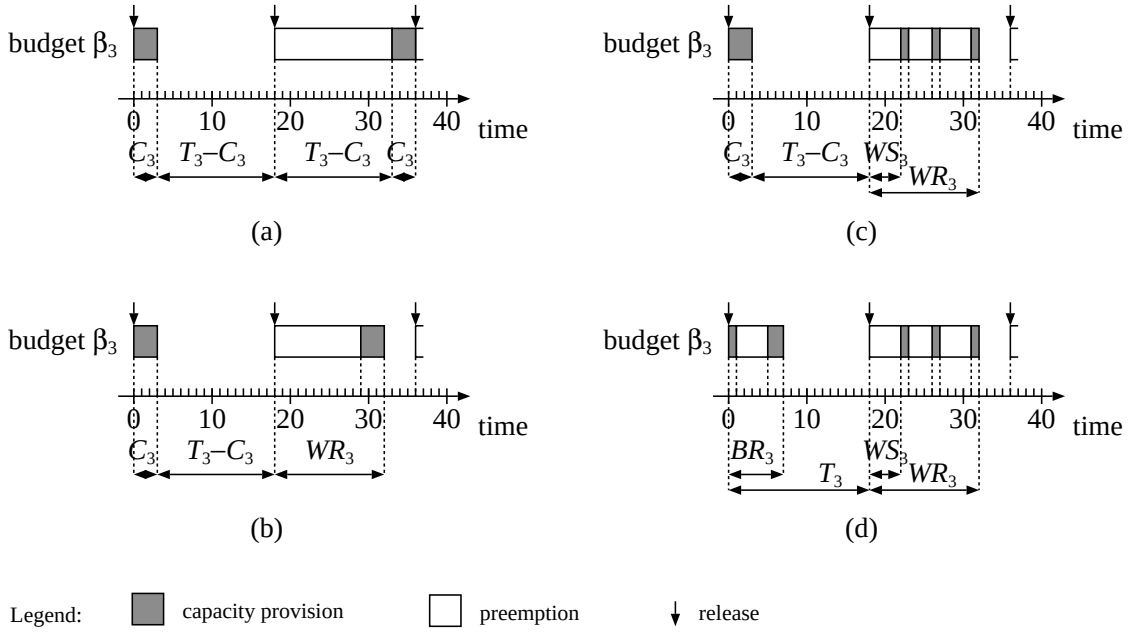

Figure 3: Worst-case assumptions for available capacity analysis for budget β_3 of our example $\mathcal{B}_1$ for two consecutive periods of β_3 according to (a) [26, 36, 38], (b) [1, 16], (c) [14], and (d) [10].

The term WS_3 in Figure 3(d) denotes the *worst-case start time* of budget β_3 [5]. It refers to the longest interval from the activation time to the actual start of the provision of the budget for which the resource provision requirement (defined for the model under consideration) can be met. If the budget is provided later than WS_3, the guarantee on resource provision will not be met because there will be insufficient time left until the completion of the period. When applied to a task, the notion implies that the task misses its deadline and is, therefore, not schedulable.

4.3 Periodic resource model

The periodic resource model [38], $\Gamma = (\Pi, \Theta)$ is characterized by a capacity $\Theta \in \mathbb{R}^+$ and a *period* $\Pi \in \mathbb{Z}^+$ such that it *guarantees* the allocation of Θ units of resource every Π time units with no assumptions on the exact provisioning of Θ within Π. We note that the domain of Π can be extended to $\mathbb{R}^+$ without any loss in generality.

Without any knowledge about scheduling at the budget level, the worst-case scenario is then assumed to occur when the capacity of β becomes *entirely* available at the beginning in the first period and at the end in the last period, as illustrated in Figure 3(a) for two consecutive periods of β_3 of our example $\mathcal{B}_1$. In this model, a *resource supply bound function* $sbf_\Gamma(t)$ is defined which coincides with our worst-case available capacity $WC^\Gamma(t)$ i.e. the minimum resource supply of Γ during t time units. As originally proposed by Shin and Lee [38], the required $sbf_\Gamma(t)$ for a periodic resource model is given by

$$sbf_\Gamma(t) = \begin{cases} y\Theta + max\{0, t - x - y\Pi\} & t \geq x \\ 0 & otherwise \end{cases}, \tag{22}$$

where $x = 2(\Pi - \Theta)$ and $y = \left\lfloor \frac{t-(\Pi-\Theta)}{\Pi} \right\rfloor$. They also define a *linear supply bound function* $lsbf_\Gamma(t)$ which gives a linear lower bound of the supply bound function $sbf_\Gamma(t)$. This lower bound is given by

$$lsbf_\Gamma(t) = \begin{cases} \frac{\Theta}{\Pi}(t - x) & t \geq x \\ 0 & otherwise \end{cases}. \tag{23}$$

Furthermore, they prove that $lsbf_\Gamma(t)$ is actually a lower bound of $sbf_\Gamma(t)$. We do not repeat the proof here but rather direct the interested reader to [38], Lemma 1 where the proof was presented. Figure 4(a) shows the $sbf_\Gamma(t)$ and corresponding $lsbf_\Gamma(t)$ for budget β_3 of our example $\mathcal{B}_1$. These lines follow from the worst-case available capacity depicted in Figure 3(a) and match the results obtainable using Equations (22) and (23) respectively.

Finally, we make the following observations: Although not explicitly stated in [38], we define $sbf_\Gamma(t)$ and $lsbf_\Gamma(t)$ as piecewise functions because a negative capacity which could otherwise arise for $0 < t < 2(\Pi - \Theta)$ does not make any sense. For $sbf_\Gamma(t)$, the condition $t \geq \Pi - \Theta$ may also be used as done in [16]. This condition denotes the *worst-case starting time* of β. If the resource is made available after this time within a given period, T, the guaranteed capacity according to the periodic resource model cannot be provided. In any case, it can be clearly seen from Figure 3(a) that $sbf_\Gamma(t)$ evaluates to zero in the interval $\Pi - \Theta < t < 2(\Pi - \Theta)$ thus justifying our choice of the same condition i.e. $t \geq x(= 2(\Pi - \Theta))$ in both Equation (22) and Equation (23). Secondly, according to [38], under assumptions of no resource sharing, the condition $2\Pi < T_m$ should be adhered to when using the period resource model for reasons of efficient implementation where T_m corresponds to task τ_m having the shortest period.

4.4 Explicit-deadline periodic (EDP) resource model

The EDP resource model [16] is a generalization of the periodic model with characteristics $\Omega = (\Pi, \Theta, \triangle)$ where $\Theta, \Pi \in \mathbb{R}^+$ are defined as before i.e. capacity and period respectively, and $\triangle \in \mathbb{R}^+$ is the *(relative) deadline* of the EDP resource with $\Theta \leq \Delta \leq \Pi$. Hence, Θ units of resource are *guaranteed* every period Π, before the deadline $\triangle$. This leads to the worst-case scenario depicted in Figure 3(b) where the capacity of β becomes *entirely* available at the beginning in the first

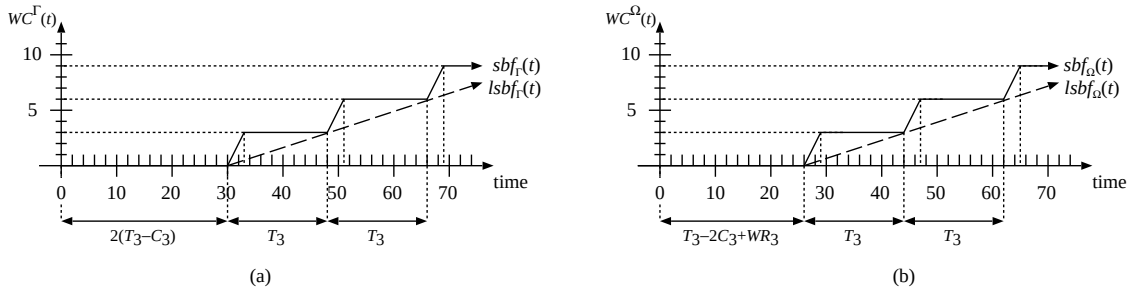

Figure 4: Worst-case available capacity $WC^{\beta}(t)$ in an interval of length t from a budget β_3 of our example $\mathcal{B}_1$ according to (a) [26, 36, 38], and (b) [1, 16].

period and by the *deadline* in the last period. Clearly, by setting $\Delta = \Pi$, the EDP model reverts to the periodic model i.e. $\Gamma(\Pi, \Theta) = \Omega(\Pi, \Theta, \Pi)$. Furthermore, we note that the analysis in [1] corresponds to these assumptions by choosing a so-called 'initial latency' equal to $WR_3^{\beta} - C_3^{\beta}$. Similar to the period model, a *resource supply bound function* $sbf_{\Omega}(t)$ defined as the minimum resource supply of Ω during t time units and corresponding to our notion of worst-case available capacity $WC^{\Omega}(t)$ is given by

$$sbf_{\Omega}(t) \quad = \quad \begin{cases} y\Theta + max\{0, t - x - y\Pi\} & t \geq x) \\ 0 & otherwise \end{cases}, \tag{24}$$

where $x = \Pi + \Delta - 2\Theta$ and $y = \left\lfloor \frac{t-(\Delta-\Theta)}{\Pi} \right\rfloor$. As for the periodic resource model, a *linear supply bound function* $lsbf_{\Omega}(t)$ which gives a linear lower bound of the supply bound function $sbf_{\Omega}(t)$ is defined as follows:

$$lsbf_{\Omega}(t) \quad = \quad \begin{cases} \frac{\Theta}{\Pi}(t - x) & t \geq x \\ 0 & otherwise \end{cases}. \tag{25}$$

Figure 4(b) shows the $sbf_{\Omega}(t)$ and corresponding $lsbf_{\Omega}(t)$ for budget β_3 of our example $\mathcal{B}_1$. We observe that this figure is identical to Figure 4(a) for the periodic resource model except for an offset of $\Pi - \Delta = T_3 - WR_3$ (in Figure 4(a) w.r.t Figure 4(b)). Herein lies the basic difference between the two resource models. Using the EDP model, the resource β may be made available to the application earlier, possibly resulting in improved schedulability of the application i.e. by shortening the interval between the best- and worst-case response-times of tasks in the application.

4.5 Other models

Apart from the periodic and EDP resource models, other models are conceivable. Referring again to Figure 3, the availability in the last period can be improved by using worst-case analysis techniques for that period [14] leading to the situation in Figure 3(c). Finally, the availability in the first period can be improved by using best-case analysis techniques for that period [10], producing yet another model depicted in Figure 3(d). Nevertheless, we once again emphasize that unlike the models in Sections 4.3 and 4.4, classes (c) and (d) deviate from the principle of locality of schedulability analysis as they make assumptions on the characteristics of other budgets in the system in order to derive the best- and/or worst-case response-times. Lastly, we observe that $WC^{\beta}(t)$ gradually improves from class (a) till class (d). Unfortunately, the complexity of

the worst-case response-time analysis of tasks also increases in the same order. Moreover, the assumptions for all four classes are pessimistic and therefore give rise to a lower bound on the worst-case available capacity. As an example, (d) is pessimistic because the best- and worst-case response-times are not necessarily assumed for the same phasing nor for two subsequent releases and the definition of a *critical instant* for tasks presented in Section 4.1 does not apply for this class as shown in [10]. Finally, because the worst-case response-time will typically not be assumed for subsequent releases either, both (c) and (d) become pessimistic as soon as we consider more than two periods, as explained in [3].

In what follows, we focus on the EDP model and when necessary show how the results can easily be applied to the periodic model. However, before discussing the unavailability model, we briefly address best-case response-time analysis.

4.6 Best-case response-time analysis of tasks

The same approach employed in Section 4.1 for worst-case response-time analysis of tasks can also be used for their best-case analysis. Equation (9) for the best-case response-time of a task τ_i of a set $\mathcal{T}$ assumes that the entire processor is available to $\mathcal{T}$. However, when only the capacity of a budget β_α is available to the tasks of an application $\mathcal{A}_\alpha$, we can simply replace the x at the left hand side of Equation (9) by the *best-case (i.e. maximum) available capacity* $BC_\alpha^\beta(x)$. Hence, the best-case response-time $BR_{i,\alpha}$ of a task $\tau_{i,\alpha}$ of application $\mathcal{A}_\alpha$ with associated budget β_α is given by the largest $x \in \mathbb{R}^+$ satisfying

$$BC_\alpha^\beta(x) = BC_{i,\alpha} + \sum_{1 \le j < i} \left(\left\lceil \frac{x}{T_{j,\alpha}} \right\rceil - 1 \right) BC_{j,\alpha}. \tag{26}$$

For the approaches considered in preceding sections (4.3 to 4.5), such a smallest value exists for $\tau_{i,\alpha}$ if and only if $BU_{i-1,\alpha}^{\mathcal{A}} < U_\alpha^\beta$. Because $BU_\alpha^{\mathcal{A}} < WU_\alpha^{\mathcal{A}}$ by definition, and we assume $WU_\alpha^{\mathcal{A}} \le U_\alpha^\beta$ and $BC_{i,\alpha} > 0$ for all tasks in any application, the relation $BU_{i-1,\alpha}^{\mathcal{A}} < U_\alpha^\beta$ trivially holds for all tasks of $\mathcal{A}_\alpha$.

An *optimal instant* of a task $\tau_{i,\alpha}$ is assumed when (1) the completion of $\tau_{i,\alpha}$ coincides with the simultaneous release of all tasks in $\mathcal{A}_\alpha$ with a higher priority than $\tau_{i,\alpha}$; (2) all the higher priority tasks experience maximum delay at the simultaneous release and minimum delay in all previous releases; and (3) the simultaneous release coincides with the end of an interval with a maximum supply of resource β_α during which its capacity C_α^β is made available as early as possible in the last period and as late as possible (i.e. at the deadline Δ of the EDP resource) in all previous releases within the overlapping interval [6]. Unlike the critical instant which is the same for all tasks, the optimal instant for each task may vary following directly from the definition of an optimal instant above.

To determine $BR_{i,\alpha}$, an auxiliary function $\overline{BC_\alpha^\beta}(y)$ may also be defined as done for the worst-case analysis. However, we skip the details and now proceed to the best-case available capacity analysis for the EDP resource model.

4.7 Best-case available capacity analysis

As seen in Equation (26), in order to determine the best-case response-time of tasks, we need the *best-case available capacity* $BC^\beta(t)$ that becomes available in an interval of length t from a budget β. This is the maximum capacity of the resource that can be made available to the application within the given time interval.

For $t > 0$, we may assume without loss of generality that the interval has an overlap with at least two periods of the budget β [10]. A maximum amount of capacity becomes available in an

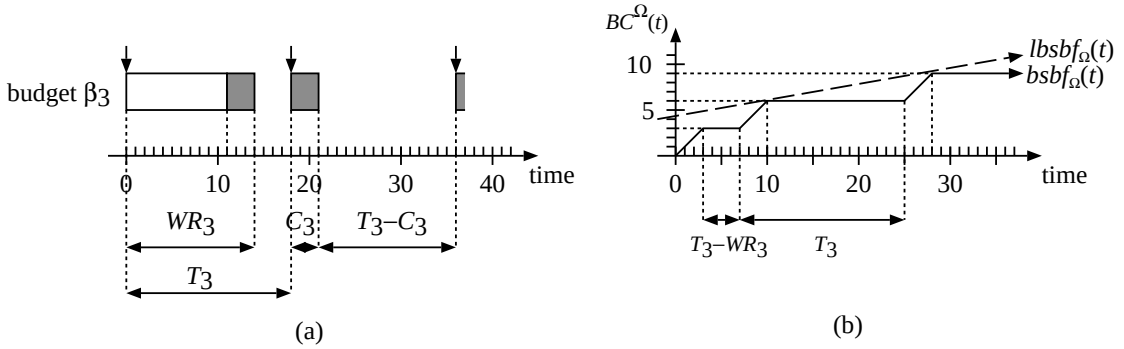

Figure 5: (a) Best-case assumptions leading to the (b) best-case available capacity $BC^{\beta}(t)$ in an interval of length t from a budget β_3 of our example $\mathcal{B}_1$ assuming the EDP resource model.

interval of length $t > 0$ when the capacity becomes available *as late as possible* in a first period of β overlapping with the interval and *as early as possible* in the last overlapping period. For the EDP resource model, we can then define a *best-case resource supply bound function* $bsbf_{\Omega}(t)$ which coincides with our best-case available capacity $BC^{\Omega}(t)$ i.e. the maximum resource supply of Ω during t time units as follows:

$$bsbf_{\Omega}(t) = \begin{cases} t & 0 \leq t \leq \Theta \\ \Theta & \Theta \leq t \leq x \\ (z+1)\Theta + max\{0, t-x-z\Pi\} & t \geq x \end{cases}, \tag{27}$$

where $x = \Theta + (\Pi - \Delta)$, $y = 2\Theta - \Delta$ and $z = \left\lfloor \frac{t-y}{\Pi} \right\rfloor$. Similar to the worst-case available capacity analysis (Section 4.4), we define a *linear best-case supply bound function* $lbsbf_{\Omega}(t)$ which gives a linear upper bound of the best-case supply bound function $bsbf_{\Omega}(t)$ is defined as follows:

$$lbsbf_{\Omega}(t) = \frac{\Theta}{\Pi}(t - (2\Theta + \Pi - \Delta)) + 2\Theta. \tag{28}$$

Figure 5(a) depicts the best-case assumptions for resource availability leading to the best-case supply in Figure 5(b) which shows the $bsbf_{\Omega}(t)$ and corresponding $lbsbf_{\Omega}(t)$ for budget β_3 of our example $\mathcal{B}_1$ as defined by Equations (27) and (28) respectively. In [28], a *maximum supply function,* $Z_{\Pi}^{max}(t)$, which corresponds to the best-case resource supply bound function for a periodic resource is defined in the context of hierarchical scheduling for dependent applications. However, the function $Z_{\Pi}^{max}(t)$ was only defined in words with no corresponding mathematical equation. Likewise, at the time of writing, we are unaware of any best-case analysis for the EDP resource model. Therefore, the analysis presented in this and the previous section is novel.

To conclude this section, we remark that for the periodic resource model, the best-case assumption for available capacity can be obtained by letting WR_3 go to T_3 in Figure 5(a).

5 Response-time analysis by modeling resource unavailability

In this section, we show how a two-level H-FPPS on a shared EDP resource can be converted to single-level FPPS on a shared resource by viewing the resource *unavailability* as interference by two highest priority fictive periodic tasks, τ_{-1} and τ_0 respectively. Next, we derive equations for the worst- and best-case response-time analysis according to this *unavailability model*.

5.1 Modeling unavailability of a budget

We will now show by means of construction that the worst-case and best-case *unavailability* of Ω can be modeled as worst-case and best-case *interference* of two fictive tasks, a periodic task τ_0 with activation jitter and a strictly periodic task $\tau_{\text{-}1}$. Task τ_0 is characterized by $T_0 = \Pi$, $AJ_0 = \Theta$, a *fixed* computation time $C_0 = BC_0 = WC_0 = \Delta - \Theta$, and $BD_0 = WD_0 = C_0$. Task $\tau_{\text{-}1}$ is characterized by $T_{\text{-}1} = \Pi$, a *phasing* $\varphi_{\text{-}1} = \Delta - \Theta$ relative to the end of the activation interval of task τ_0 (which has an arbitrary phasing), a *fixed* computation time $C_{\text{-}1} = BC_{\text{-}1} = WC_{\text{-}1} = \Pi - \Delta$, and $BD_{\text{-}1} = WD_{\text{-}1} = C_{\text{-}1}$. These task characteristics are summarized in Table 2 for easy reference. The *utilization* U_0^τ of τ_0 and $U_{\text{-}1}^\tau$ of $\tau_{\text{-}1}$ are given by $U_0^\tau = C_0/T_0 = (\Delta - \Theta)/\Pi$ and $U_{\text{-}1}^\tau = C_{\text{-}1}/T_{\text{-}1} = (\Pi - \Delta)/\Pi$, respectively. Hence, $U_0^\tau + U_{\text{-}1}^\tau = (\Pi - \Theta)/\Pi = 1 - U_\alpha^\beta$, where U_α^β is the utilization of $\mathcal{A}_\alpha$. This clearly shows that by including these two fictive tasks, the processor is fully utilized (i.e. $U_{\text{-}1}^\tau + U_0^\tau + U_\alpha^\beta = 1$).

(fictive) task	T	$C = D$	AJ	φ
$\tau_{\text{-}1}$	Π	$\Pi - \Delta$	0	$\Delta - \Theta$
τ_0	Π	$\Delta - \Theta$	Θ	-

Table 2: Characteristics of fictive tasks $\tau_{\text{-}1}$ and τ_0.

A situation with a worst-case (i.e. minimum) resource supply of Ω in an interval starting at time t_{S} (and extending till the end of the subsequent period without loss of generality) [16] is shown in Figure 6(a). The same figure also illustrates that the *unavailability* of Ω in that interval can be modeled by the *worst-case interference* of $\tau_{\text{-}1}$ and τ_0.

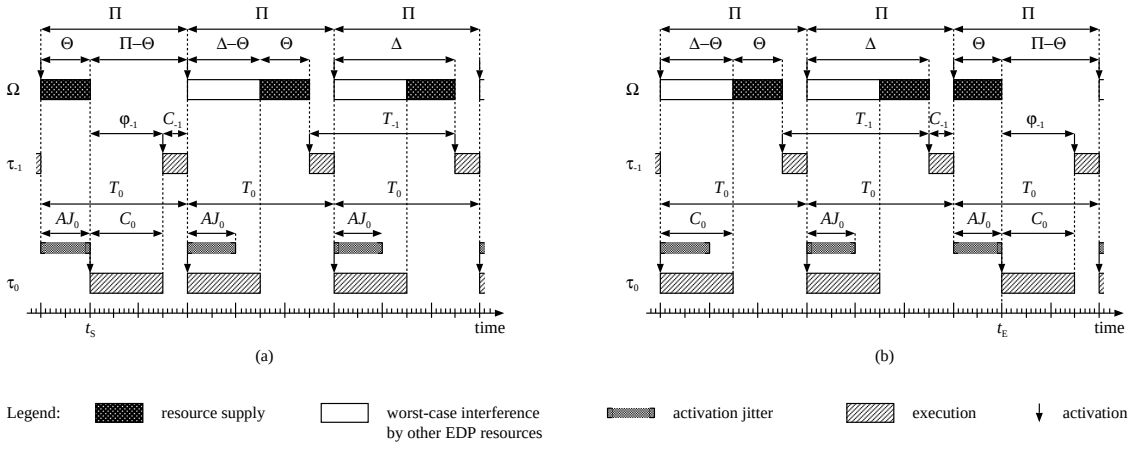

Figure 6: A situation with (a) a worst-case (i.e. minimum) resource supply of an EDP resource Ω in an interval starting at time t_{S}, (b) a best-case (i.e. maximum) resource supply of an EDP resource Ω in an interval ending at time t_{E}, and two periodic tasks τ_0 and $\tau_{\text{-}1}$ modeling the *unavailability* of Ω.

Similar to the worst-case, a situation with a best-case (i.e. maximum) resource supply of Ω in an interval ending at time t_{E} (and starting at the beginning of the previous period without loss of generality) is shown in Figure 6(b). The same figure also illustrates that the *unavailability* of Ω in that interval can be modeled by the *best-case interference* of $\tau_{\text{-}1}$ and τ_0. We note that tasks $\tau_{\text{-}1}$ and τ_0 always execute in disjoint intervals of time. Furthermore, when the deadline Δ of the EDP resource Ω is equal to its period Π, the computation time of $\tau_{\text{-}1}$ becomes zero, i.e. $C_{\text{-}1} = \Pi - \Delta = 0$, and the resulting situation models a periodic resource. Finally, we note that when Ω is equal to the entire resource, it is always available and the computation times C_0 and $C_{\text{-}1}$ of τ_0 and $\tau_{\text{-}1}$ respectively both become zero, i.e. $C_0 = \Pi - \Theta = C_{\text{-}1} = \Pi - \Pi = 0$.

For the analysis of the tasks of (generic) application $\mathcal{A}_\alpha$, we consider an extension $\widehat{\mathcal{A}_\alpha}$ of $\mathcal{A}_\alpha$ with the fictive tasks τ_0 and $\tau_{\text{-}1}$ at the two highest priorities i.e. $\widehat{\mathcal{T}_\alpha^\beta} = \mathcal{T}_\alpha^\beta \cup \{\tau_0, \tau_{\text{-}1}\}$, Under two dedicated assumptions for resource provisioning to $\mathcal{A}_\alpha$, the analytical results for the tasks of $\mathcal{A}_\alpha$ when executed on a shared EDP resource are identical to the results for those tasks of $\widehat{\mathcal{A}_\alpha}$ when executed with τ_0 and $\tau_{\text{-}1}$ on a shared resource.

We start with the two dedicated assumptions for resource provisioning to application $\mathcal{A}_\alpha$ based on the resource supply of the EDP resource β_α. Worst-case response-time analysis and best-case response-time analysis are subsequently addressed.

5.2 Assumptions for resource provisioning to $\mathcal{A}_\alpha$

For *worst-case analysis* of the tasks of an application $\mathcal{A}_\alpha$, it is required that the *minimum* amount of resources that is guaranteed to $\mathcal{A}_\alpha$ is known. We therefore assume that the capacity Θ of a budget β_α is *entirely* available to its associated application $\mathcal{A}_\alpha$, similar to the [16].

Similarly, for *best-case analysis* of the tasks of $\mathcal{A}_\alpha$, it is required that the *maximum* amount of resources that is provided to $\mathcal{A}_\alpha$ is known. We therefore assume that *only* the capacity Θ of a budget β_α is available to its associated application $\mathcal{A}_\alpha$, similar to hard resource reservations [33].

We note that by assuming (idling) periodic servers and a 1-to-1 relationship between applications and budgets, the capacity of a budget becomes *entirely* and *exclusively* available to an application.

5.3 Worst-case response-time analysis

Worst-case response-time analysis of tasks with activation jitter on a single processor under FPPS has been addressed in [40], amongst others. For worst-case deadlines at most equal to periods minus activation jitter, i.e. $WD_i \leq T_i - AJ_i$, the worst-case response-time WR_i of task τ_i of $\widehat{\mathcal{A}_\alpha}$ is given by the *smallest* value $x \in \mathbb{R}^+$ that satisfies

$$x = WC_i + \left\lceil \frac{x - \varphi_{\text{-}1}}{T_{\text{-}1}} \right\rceil WC_{\text{-}1} + \left\lceil \frac{x + AJ_0}{T_0} \right\rceil WC_0 + \sum_{1 \leq j < i} \left\lceil \frac{x + AJ_j}{T_j} \right\rceil WC_j. \tag{29}$$

This equation is similar to Equation (16), the only new parameter being $\varphi_{\text{-}1}$ which models the (fixed) phase of task $\tau_{\text{-}1}$. We also remark that this equation only holds for this specific case where we use these two fictive tasks to represent the unavailability of the EDP resource model. It can be applied (i.e. holds for this specific case) because we know the critical instant from the unavailability model conversion.

To calculate WR_i, we can use an iterative procedure based on recurrence relationships as outlined in Section 3.1, starting with a lower bound, e.g. WC_i. The worst-case response-time $WR_{i,\alpha}$ of $\tau_{i,\alpha}$ of $\mathcal{A}_\alpha$ is equal to WR_i of τ_i of $\widehat{\mathcal{A}_\alpha}$.

We remark that Equation (29) completely eliminates the need for an auxiliary (inverse) function needed in Equation (21). Therefore, this analysis using the unavailability model approach is simplified compared to the supply bound function-based approach described in Section 4.1. The same observation holds for the best-case response-time analysis presented next.

5.4 Best-case response-time analysis

Best-case response-time analysis of tasks on a single processor under FPPS has been addressed in [9, 34, 7], amongst others. For $WD_i \leq T_i - AJ_i$, the best-case response-time BR_i of task τ_i of $\widehat{\mathcal{A}_\alpha}$ is given by the *largest* $x \in \mathbb{R}^+$ that satisfies

$$x = BC_i + \left(\left\lceil \frac{x + \varphi_{\text{-}1}}{T_{\text{-}1}} \right\rceil - 1 \right) BC_{\text{-}1} + \left(\left\lceil \frac{x - AJ_0}{T_0} \right\rceil - 1 \right)^+ BC_0 + \sum_{1 \le j < i} \left(\left\lceil \frac{x - AJ_j}{T_j} \right\rceil - 1 \right)^+ BC_j. \tag{30}$$

Here, the notation w^+ stands for $\max(w, 0)$, which is used to indicate that the number of preemptions of tasks with a higher priority than τ_i cannot become negative. Again, we observe that Equation (30) closely matches Equation (17) except that here, the (fixed) phase $\varphi_{\text{-}1}$ of task $\tau_{\text{-}1}$ has been taken into account.

To calculate BR_i, we can again use an iterative procedure based on recurrence relationships, starting with an upper bound, e.g. the worst-case response-time WR_i of task τ_i. The best-case response-time $BR_{i,\alpha}$ of $\tau_{i,\alpha}$ of $\mathcal{A}_\alpha$ is equal to BR_i of τ_i of $\widehat{\mathcal{A}_\alpha}$.

5.5 Applying the response-time analysis to an example

To conclude this section, we apply the unavailability model presented above to an example. We derive the worst- and best-case response-times by construction and show that they are identical to the results obtained using Equations (29) and (30) respectively. We consider an application $\mathcal{A}_2$ consisting of two tasks (task set $\mathcal{T}_2^\beta$) whose characteristics are presented in Table 3. We assume that this application is associated with budget β_2 of our previous example set $\mathcal{B}_1$ whose characteristics correspond to those of task set $\mathcal{T}_1$ in Table 1. For the sake of convenience, we repeat the characteristics of β_2 in Table 4 using the notation introduced in Section 4.4 for the EDP model.

task	$T = D$	C	WR	BR
$\tau_{1,2}$	7	1	5	1
$\tau_{2,2}$	20	4	20	10

Table 3: Characteristics of $\mathcal{T}_2^\beta$ (of application $\mathcal{A}_2$) with worst-case and best-case response-times of tasks.

server	Π	Θ	Δ*
β_2	5	2	3

Table 4: Characteristics of β_2
*Δ = worst-case response-time of β_2 in $\mathcal{B}_1$.

Now, we convert $\mathcal{A}_2$ on a shared EDP resource to $\widehat{\mathcal{A}_2}$ on an FPPS shared resource by defining $\tau_{\text{-}1}$ and τ_0 whose characteristics (derived from Table 2) are: $T_0 = T_{\text{-}1} = 5$, $C_{\text{-}1} = D_{\text{-}1} = 2$, $\varphi_{\text{-}1} = 1$, $C_0 = D_0 = 1$, and $AJ_0 = 2$.

Figure 7(a) shows a timeline of $\mathcal{T}_2^\beta$ with a critical instant for task $\tau_{2,2}$ and a(n) (EDP) worst-case scenario for β_2 as defined in Sections 4.1 and 4.4 respectively. Similarly, Figure 7(b) depicts a timeline of $\widehat{\mathcal{T}_2^\beta}$ where the unavailability model is used to calculate the worst-case response-times of $\tau_{1,2}$ and $\tau_{2,2}$. As seen, both timelines yield the same results namely, $WR_{1,2} = 5$ and $WR_{2,2} = 20$. Unsurprisingly, the results obtained using Equation (29) are also identical to those presented in Figure 7. The actual derivation of the analytic results is presented in Appendix B.1.

We repeat the same procedure to obtain the best-case response-times of all tasks in $\mathcal{T}_2^\beta$ summarized in Table 3. The timelines in Figure 8 depict an optimal instant for $\tau_{2,2}$ (refer to Section 3.3) which we use to determine the best-case response-times of $\tau_{1,2}$ and $\tau_{2,2}$ by construction. In these timelines, we also assume an EDP *best-case scenario* for β_2 in which the simultaneous release of $\tau_{2,2}$ and all higher priority tasks coincides with the completion of budget β_2 whose capacity is available as early as possible in that period and as late as possible in all previous releases of β_2. The derivation analytically using Equation (30) is presented in Appendix B.1. We simply state here that the graphical and analytic approaches yield exactly the same results.

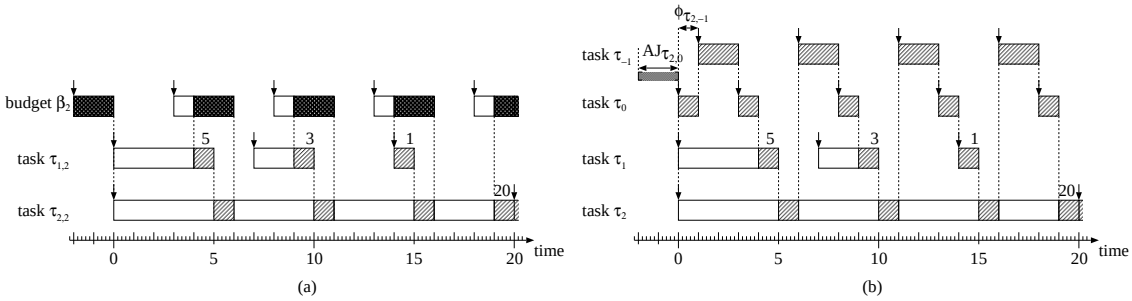

Figure 7: Timelines for a critical instant of task $\tau_{2,2}$ for (a) budget β_2 and application $\mathcal{A}_2$ according to [1, 16] and for (b) an extension $\widehat{\mathcal{A}_2}$ of $\mathcal{A}_2$ with fictive tasks $\tau_{\text{-}1}$ and τ_0.

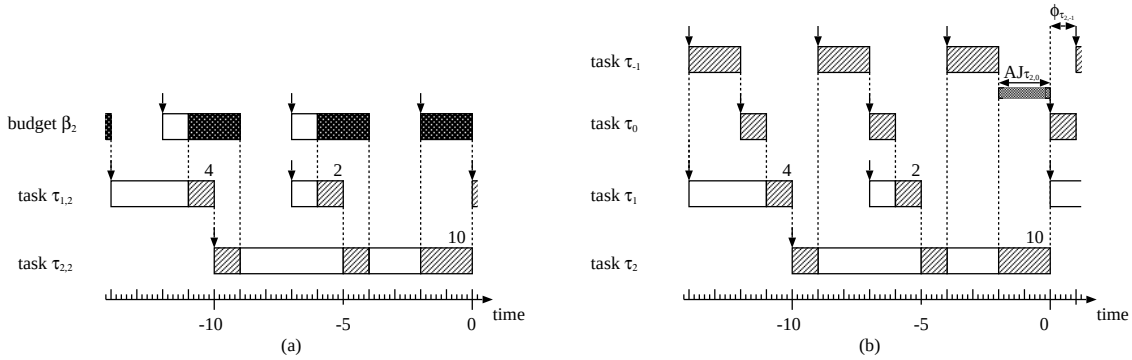

Figure 8: Timelines for an optimal instant of task $\tau_{2,2}$ for (a) budget β_2 and application $\mathcal{A}_2$ and for (b) an extension $\widehat{\mathcal{A}_2}$ of $\mathcal{A}_2$ with fictive tasks $\tau_{\text{-}1}$ and τ_0.

Having described the unavailability model, we proceed in the next section to apply it in deriving linear response-time upper bounds for two-level H-FPPS.

6 Response-time upper bounds

Response-Time Analysis (RTA) is a method widely used in fixed priority systems to determine schedulability of tasks by comparing their worst-case response-times to their deadlines. It has been addressed in preceding sections of this document for the real-time scheduling model considered (see Sections 2, 3 and 4). Unfortunately, exact response-time analysis is known to be pseudo-polynomial in complexity [2, 20]. Thus, it is sometimes desirable to use linear response-time upper bounds to check on a task-by-task basis whether an exact response-time calculation is required. This is especially the case in open systems where it might be necessary to have run-time admission tests or in development tools [15]. Linear response-time upper bounds constitute a *sufficient condition* which, although pessimistic, has been shown in [13] to significantly improve the efficiency of exact schedulability tests.

In this section, we derive closed form upper bounds for sets of independent tasks with deadlines less than or equal to their periods, activation jitter and arbitrary phasing scheduled using FPPS. We build upon the work presented in [4] and [15]. We begin by recapitulating the response-time upper bound presented in [15] for a shared resource. Subsequently, we describe how this method can be applied to a shared EDP resource using the *(resource) supply bound function* sbf_Ω [16] from which we derive a *linear supply lower bound* $lsbf_\Omega$ in the manner employed in [38] for the periodic resource model. To enable this conversion to the EDP resource model, we also extend the

analysis by considering the specific phasing of fictive task τ_{-1} relative to fictive task τ_0. Finally, we show that by converting the shared EDP resource to a shared resource using the unavailability model described in Section 5, the analysis in [15] can be directly reused. Furthermore, we prove that by first summing the interference due to the fictive tasks before taking their *interference upper bound*, the two methods are equivalent.

6.1 Existing analysis for FPPS

Similar to the approach outlined in Section 4.1 of [15], we now derive response-time upper bounds for tasks scheduled under FPPS. However, we do not consider tasks with arbitrary deadlines as is the case in that paper. Rather, we assume that for each task, τ_i, the relation $D_i \leq T_i - AJ_i$ holds (since otherwise there may be too little time between two successive releases to complete the task). The other assumptions in our real-time model for FPPS hold (see Section 2).

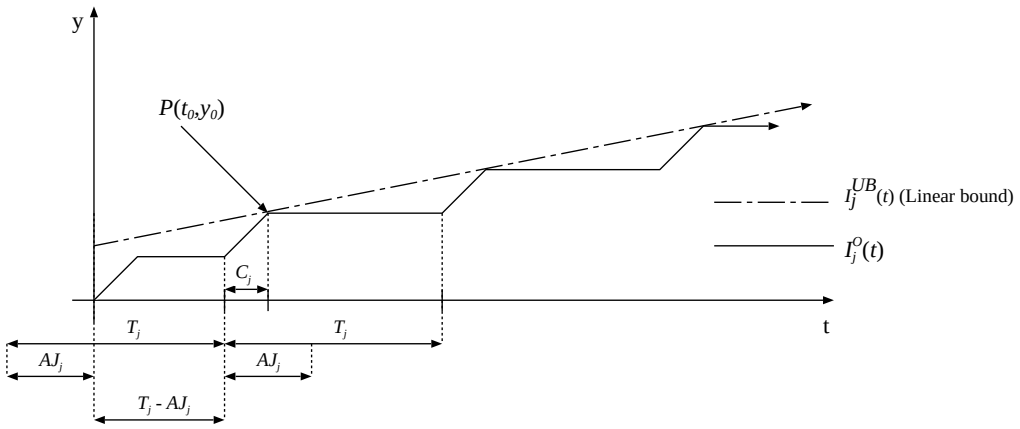

Figure 9: Worst-case interference $I_j^O(t)$ of task τ_j and the corresponding upper bound $I_j^{UB}(t)$ assuming it is the only task in the system.

Let $I_j(t)$ be the *worst-case interference* due to task τ_j, i.e. the total time the processor spends executing τ_j during the interval $[0, t)$ in the worst-case scenario. Task τ_j experiences a maximum release delay upon activation and a minimum delay at subsequent releases. Let $I_j^O(t)$ denote the worst-case interference due to task τ_j when it is the *only* task in the system, from which the following relation directly holds: $I_j^O(t) \geq I_j(t), \forall\, t$.

Given that τ_j is the only task in the system, the processor will, in general, execute the first job of τ_j followed by an interval of idleness ($= T_j - C_j - AJ_j$) after which it subsequently executes C_j every T_j. This situation is depicted in Figure 9.

Our goal is to derive the linear upper bound, $I_j^{UB}(t)$, on $I_j^O(t)$ shown as a dashed line in Figure 9. The slope of this line equals the utilization U_j^τ of task τ_j given by Equation (2). Hence, we can determine the equation of the line once we have the coordinates of any point lying on it. We select point $P(t_0, y_0)$ in Figure 9 which represents the smallest value of t for which $I_j^O(t) = I_j^{UB}(t)$ (assuming that $AJ_j > 0$).

Since there can be only two invocations of the task[1] before point $P(t_0, y_0)$, its y-coordinate is given by

$$y_0 = 2C_j. \tag{31}$$

[1]Note that this follows from our assumption of deadlines less than or equal to periods which, in turn, simplifies the analysis. In particular, the following derivation only holds if $C_i < D_i \leq T_i - AJ_i$. For a more general derivation covering the case where deadlines may exceed periods, we refer the interested reader to [15].

From Figure 9, the t-coordinate of point $P(t_0, y_0)$ is simply

$$t_0 = (T_j - AJ_j) + C_j. \tag{32}$$

Combining Equations (32) and (31), the desired equation for the linear bound is

$$I_j^{UB}(t) = U_j^\tau t + U_j^\tau AJ_j + C_j(1 - U_j^\tau), \tag{33}$$

where U_j^τ is the utilization of task τ_j.

To obtain an upper bound on the total interference of task τ_i in the time interval $[0, t)$ due to higher priority tasks, we sum the interference from all the higher priority tasks i.e.

$$\sum_{1 \le j < i} I_j^{UB}(t) = t \cdot \sum_{1 \le j < i} U_j^\tau + \sum_{1 \le j < i} \left(U_j^\tau AJ_j + C_j(1 - U_j^\tau) \right). \tag{34}$$

The upper bound on the demand of the processor due to task τ_i and higher priority tasks is then obtained by adding the computation time C_i of task τ_i to Equation (34) i.e.

$$y = C_i + t \cdot \sum_{1 \le j < i} U_j^\tau + \sum_{1 \le j < i} \left(U_j^\tau AJ_j + C_j(1 - U_j^\tau) \right). \tag{35}$$

We note that Equation (35), which gives an upper bound on the total demand at level-i i.e. the worst-case interference due to higher priority tasks combined with the computation time of task τ_i, may be considered as the *demand upper bound function* denoted by $ldbf(t)$ (cf. [38]).

Let $O_i^{UB}(C_i)$ be an upper bound on the longest time that the processor takes to execute task τ_i and all higher priority tasks. It is given by the intersection of the line $y = t$ and the line in Equation (35) as follows:

$$O_i^{UB}(C_i) = \frac{C_i + \sum_{1 \le j < i} \left(U_j^\tau AJ_j + C_j(1 - U_j^\tau) \right)}{1 - \sum_{1 \le j < i} U_j^\tau}. \tag{36}$$

Equation (36) gives an upper bound on the worst-case response-time of any task scheduled on a single shared resource using FPPS. If this resulting value of $O_i^{UB}(C_i)$ is less than D_i, we conclude that task τ_i is schedulable. Otherwise, another method must be adopted to determine schedulability e.g. an exact schedulability analysis.

The interested reader is directed to [15] where a proof that Equation (36) is actually an upper bound on the *worst-case occupied time* $O_i(C_i)$ [5] due to the interference of task τ_i and all higher priority tasks is provided. For notational convenience, we will use R_i^{UB} instead of $O_i^{UB}(C_i)$ in the rest of this document as the former suggests (mnemonically that it concerns the) response-time upper bound.

6.2 Applying the existing analysis to H-FPPS using the linear supply bound function $lsbf_\Omega(t)$

The response-time upper bound presented in Section 6.1 assumes the entire processor is available to our task set $\mathcal{T}$. We now discuss how this result can be applied in the context of a shared EDP resource. The approach we adopt is the same as was used in Section 4.1 when we converted the worst-case response-time analysis of a single level FPPS to a two-level H-FPPS. What we require is a *linear supply bound function* $lsbf_\Omega(t)$ which gives the *minimum* resource supply of Ω in the interval $[0, t)$. Such a (tight) lower bound on an EDP resource supply was defined in Section 4.4 by considering the worst-scenario for resource provisioning (see Equation (25)). A task arriving just after the depletion of the budget's capacity $C_\alpha^\beta (= \Theta)$ experiences maximum starvation of

$(\Pi - \Theta) + (\Delta - \Theta)$ after which Θ units of resource is guaranteed to β_α every Π units of time. The slope of this lower bound is the utilization U_α^β of β_α i.e. $\frac{\Theta}{\Pi}$, with an intercept on the time axis equal to the maximum starvation. The intersection between the lines in Equations (25) and (35) gives the desired equation for calculating the worst-case response-time upper bound for tasks on a shared EDP resource and is presented in Equation (37) i.e.

$$R_{i,\alpha}^{UB}(\Omega) = \frac{C_i + \sum_{1 \leq j < i} \left(U_j^\tau A J_j + C_j(1 - U_j^\tau)\right) + U_\alpha^\beta(\Pi + \Delta - 2\Theta)}{U_\alpha^\beta - \sum_{1 \leq j < i} U_j^\tau}. \tag{37}$$

We denote this bound as $R_{i,\alpha}^{UB}(\Omega)$ to highlight the dependence on parameters from the EDP resource, Ω and also append a subscript α to show that task $\tau_{i,\alpha}$ is associated with application $\mathcal{A}_\alpha$.

At this juncture, we revisit our previous example of the application $\mathcal{A}_2$ and its associated budget β_2 whose characteristics were presented in Tables 3 and 4 respectively. Applying Equation (37) to the example, we obtain worst-case response-time upper bounds of 6.5 and 25.11 for $\tau_{1,2}$ and $\tau_{2,2}$ respectively (see Appendix B.2 for the calculations) from which we conclude that whereas task $\tau_{1,2}$ is schedulable, an exact schedulability test is required for task $\tau_{2,2}$. These results are summarized in Table 5. The results in the last column of that table come from the alternative approach described next.

task	$T = D$	C	WR	$R_{i,2}^{UB}(\Omega)$	$\widehat{R_i^{UB}}$
$\tau_{1(,2)}$	7	1	5	6.5	7.5
$\tau_{2(,2)}$	20	4	20	25.11	26.67

Table 5: (Upper bounds of) worst-case response-times for tasks $\tau_{1(,2)}$ and $\tau_{2(,2)}$ in application $\mathcal{A}_2$ ($\widehat{\mathcal{A}_2}$).

Finally, we remark that by checking tasks in priority order, highest priority first, the summation terms in Equations (36) and (37) can be computed incrementally. Hence, this sufficient schedulability test can be computed in linear time with respect to the number of tasks.

6.3 An alternative approach based on unavailability model

In Section 5 we showed that the unavailability model provides a means to convert a shared EDP resource to a shared resource. Our motivation for this approach was to enable reuse of existing results. However, in order to do this, we first need to extend Equation (36) by taking the fixed phase $\varphi_{\text{-}1}$ of task $\tau_{\text{-}1}$ into account. In deriving the linear upper bound of the interference due to task $\tau_{\text{-}1}$, we follow the same method outlined in Section 6.1, the only difference being that we add $\varphi_{\text{-}1}$ to the right hand side of Equation (32). The resulting linear bound is therefore given by:

$$I_{\text{-}1}^{UB}(t) = U_{\text{-}1}^\tau t - \varphi_{\text{-}1} U_{\text{-}1}^\tau + C_{\text{-}1}(1 - U_{\text{-}1}^\tau). \tag{38}$$

Equation (33) can be directly used for task τ_0. Hence, the worst-case response-time upper bound for tasks using the unavailability approach is as follows:

$$\widehat{R_i^{UB}} = \frac{C_i + \sum_{-1 \leq j < i} \left(U_j^\tau A J_j + C_j(1 - U_j^\tau)\right) - \varphi_{\text{-}1} U_{\text{-}1}^\tau}{1 - \sum_{-1 \leq j < i} U_j^\tau}. \tag{39}$$

We do not include the phasing term in the summation in the numerator of Equation (39) because including specific phasing of the non-fictive tasks is more involved as our assumption of a critical instant no longer holds in general and the analysis is far from trivial (see, for example, [29]).

Nevertheless, the approach works for task $\tau_{\text{-}1}$ for the following two reasons: (1) $\tau_{\text{-}1}$ and τ_0 execute in distinct periods i.e. they do not interfere with each other; and (2) their characteristics are completely independent of the actual task set but *solely* depend on the characteristics of budget β_α which we assume to be known a priori.

Referring to our leading example, the modified task set $\widehat{T_2^\beta}$ with additional fictive tasks $\tau_{\text{-}1}$ and τ_0 whose characteristics were outlined in Section 5.5 was used to obtain worst-case response-time upper bounds of 7.5 and 26.67 for tasks τ_1 and τ_2 respectively. From these results, no conclusion can be drawn about the schedulability of either task (since they both fail to meet the sufficient condition). Thus, an exact schedulability analysis is required. This is in contrast to the results obtained using $R_{i,2}^{UB}(\Omega)$ from which we could directly conclude schedulability of task $\tau_{1,2}$.

From a computational standpoint, using the unavailability method does not result in any significant improvement given the fact that Equations (37) and (39) are very similar. Moreover, the results in Table 5 clearly show that the unavailability method as presented so far is more pessimistic. This follows from the fact that the pessimism in calculating response-time upper bounds by the method outlined in Section 6.1 can only increase with the number of tasks in the application and the unavailability model invariably increases this number by two.

However, we can improve on the response bound obtained using the unavailability model by first combining the two fictive tasks $\tau_{\text{-}1}$ and τ_0 before calculating the (worst-case) interference due this new, combined task, say $\tau_\dagger$. In other words, for these two fictive tasks, we lift the assumption of determining the worst-case interference of a task assuming it is the *only* task in the system. Instead, we find the combined interference due to these two tasks simultaneously present in the system as described next.

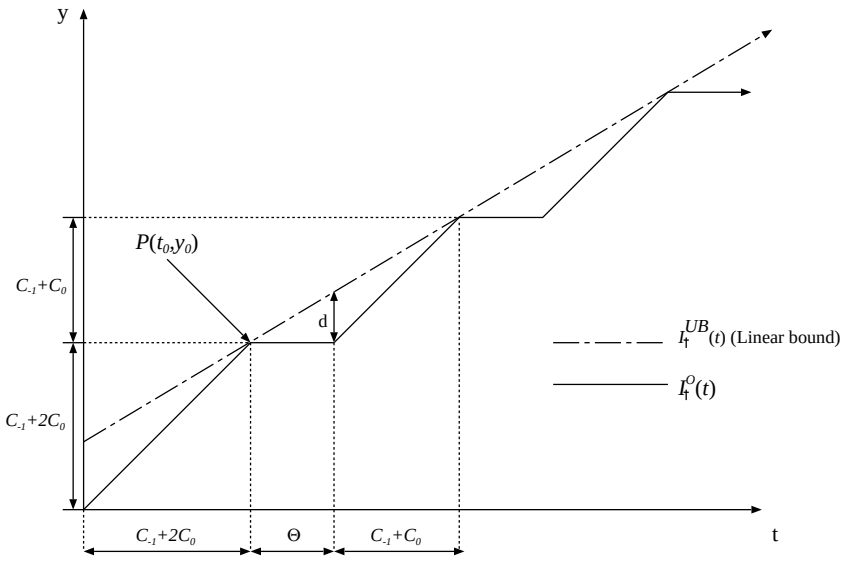

Figure 10: Worst-case interference $I_\dagger^O(t)$ of task $\tau_\dagger$ and the corresponding upper bound $I_\dagger^{UB}(t)$ assuming it is the only task in the system where $\tau_\dagger$ is the sum of $\tau_{\text{-}1}$ and τ_0.

Determining $I_\dagger^{UB}(t)$ is done in exactly the same way as we outlined in Section 6.1 i.e. we determine the coordinates of the first point $P(t_0, y_0)$ satisfying the condition $I_\dagger^O(t) = I_\dagger^{UB}(t)$. The worst-case interference due to combined fictive task $\tau_\dagger$ in an interval beginning at time $t = t_S$ is illustrated in Figure 6(a) by viewing $\tau_{\text{-}1}$ and τ_0 as a single task $\tau_\dagger$. We observe an initial processor occupancy of duration $2C_0 + C_{\text{-}1}$ followed by an idle period of duration Θ before subsequently executing $C_\dagger = C_{\text{-}1} + C_0 = \Pi - \Theta$ every Π units of time; see Figure 6(a). From Figure 10, we observe that the desired point has equal t- and y-coordinates i.e. $P(t_0, y_0) = P(t_0, t_0)$ where t_0 is

$$t_0 = 2C_0 + C_{-1} = \Delta + \Pi - 2\Theta = y_0. \tag{40}$$

The slope of the linear bound equals the utilization of $\tau_\dagger$ i.e. $U_\dagger^\tau = U_{-1}^\tau + U_0^\tau = \dfrac{\Pi - \Theta}{\Pi}$. Hence, the linear upper bound of $\tau_\dagger$ is given by

$$I_\dagger^{UB}(t) = U_\dagger^\tau (t - x) + x, \tag{41}$$

where $x = \Delta + \Pi - 2\Theta$.

With reference to Figure 10, we denote the largest difference between $I_\dagger^O(t)$ and $I_\dagger^{UB}(t)$ as d which can be determined at $t_1 = t_0 + \Theta$ as follows:

$$\begin{aligned} d &= I_\dagger^{UB}(t_1) - I_\dagger^O(t_1) \\ &= \left(\frac{C_\dagger}{T_\dagger}(t_1 - y_0) + y_0\right) - y_0 \\ &= \left(\frac{\Pi - \Theta}{\Pi}\right)\Theta \\ &= \Theta - \frac{\Theta^2}{\Pi}. \end{aligned} \tag{42}$$

From Equation (42), we observe that d can be increased by choosing a large Π. Also, by equating the first derivative of d (w.r.t. Θ) to zero we deduce that d attains a maximum value when we choose $\Theta = \frac{\Pi}{2}$. These observations can be useful at the design stage in the event we have control over the values of Π and/or Θ.

The upper bound on the total interference of task τ_i in the time interval $[0, t)$ due to higher priority tasks is now given by the summation of Equation (34) (interference due to all real higher priority tasks) and Equation (41) (interference due to the combined fictive task $\tau_\dagger$). $R_i^{UB\dagger}$ naturally follows as the intersection of Equation (41) and the line $y = t$ and is given by the following equation:

$$R_i^{UB\dagger} = \frac{C_i + \sum_{1 \le j < i} \left(U_j^\tau A J_j + C_j(1 - U_j^\tau)\right) + x(1 - U_\dagger^\tau)}{1 - U_\dagger^\tau - \sum_{1 \le j < i} U_j^\tau}. \tag{43}$$

Using Equation (43) the new worst-case response-time upper bounds for tasks τ_1 and τ_2 are found to be 6.5 and 25.11 respectively. An interesting observation is that these values are the same as those obtained using $lsbf_\Omega(\mathrm{t})$ reported in Table 5! This result is not a coincidence as we show in the following theorem.

Theorem 1 *$R_i^{UB}(\Omega)$ and $R_i^{UB\dagger}$ are equivalent methods for calculating response-time upper bounds for tasks sharing an EDP resource*[2]*.*

[2]Essentially, the proof simply involves substituting $x = \Delta + \Pi - 2\Theta$ and $U^\Omega = 1 - U_\dagger^\tau = \frac{\Theta}{\Pi}$ into Equation (43) to obtain Equation (37).

Proof

$$
\begin{aligned}
R_i^{UB\dagger} &= \frac{C_i + \sum_{1\leq j<i}\left(U_j^\tau AJ_j + C_j(1-U_j^\tau)\right) + x(1-U_\dagger^\tau)}{1-U_\dagger^\tau - \sum_{1\leq j<i} U_j^\tau} \\
&= \frac{C_i + \sum_{1\leq j<i}\left(U_j^\tau AJ_j + C_j(1-U_j^\tau)\right) + (\Delta+\Pi-2\Theta)(1-\frac{\Pi-\Theta}{\Pi})}{(1-\frac{\Pi-\Theta}{\Pi}) - \sum_{1\leq j<i} U_j^\tau} \\
&= \frac{C_i + \sum_{1\leq j<i}\left(U_j^\tau AJ_j + C_j(1-U_j^\tau)\right) + (\Delta+\Pi-2\Theta)(\frac{\Theta}{\Pi})}{(\frac{\Theta}{\Pi}) - \sum_{1\leq j<i} U_j^\tau} \\
&= \frac{C_i + \sum_{1\leq j<i}\left(U_j^\tau AJ_j + C_j(1-U_j^\tau)\right) + U_\alpha^\beta(\Pi+\Delta-2\Theta)}{U_\alpha^\beta - \sum_{1\leq j<i} U_j^\tau} \\
&= R_{i,\alpha}^{UB}(\Omega) \quad \blacksquare
\end{aligned}
$$

For the example considered, we conclude that the approach of first combining the fictive tasks $\tau_{\text{-}1}$ and τ_0 before taking the linear upper bound of the interference due to the combined task yields better results than summing the interference upper bounds of the individual tasks $\tau_{\text{-}1}$ and τ_0. We refer to these approaches as *tangent of combination* and *sum of tangents* respectively. In the next section, we extend this idea to real tasks and investigate whether the tangent of combination approach dominates the sum of tangents approach in general for tasks having the harmonic periods.

7 Improving response-time upper bounds

In Section 6.3, we saw that we could improve on the response bound obtained using the unavailability model by first combining the fictive tasks $\tau_{\text{-}1}$ and τ_0 before calculating the interference due to the combined task $\tau_\dagger$ without jeopardizing the linear computation time bound. Intuitively, it should also be possible to apply this approach to the real tasks in task set $\mathcal{T}_\alpha^\beta$ having *harmonic*[3] *periods*, thereby improving on the response-time upper bounds of *lower priority tasks*. This may result in significant improvements in the effectiveness of the sufficient schedulability condition for applications having several tasks with the same (harmonic) period(s). We begin with an example application on an EDP resource for which we illustrate the idea and the potential improvements it offers. Subsequently, we revert back to tasks scheduled on a single FPPS shared resource for which we consider both sum of tangents and tangent of combination approaches and derive equations for them starting with two tasks and generalizing to n tasks having the same period. We then extend this idea to harmonic tasks.

task	$T=D$	C	WR	$\widehat{R_i^{UB\dagger}}$	$\widehat{R_i^{UB\ddagger}}$
$\tau_{1(,3)}$	14	1	5	6.5	6.5
$\tau_{2(,3)}$	14	2	9	13.78	13.78
$\tau_{3(,3)}$	33	2	21	33.62	32.08

Table 6: (Upper bounds of) worst-case response-times for tasks $\tau_{1(,3)}$, $\tau_{2(,3)}$ and $\tau_{3(,3)}$ in application $\mathcal{A}_3$ ($\widehat{\mathcal{A}_3}$).

Consider application $\mathcal{A}_3$ consisting of three tasks with the two highest priority tasks having the same period. The characteristics of the task set are summarized in Table 6. The associated budget remain β_2 from our previous example. We note that the utilization of the task set is

[3] A precise (mathematical) definition of harmonic tasks is given in Section 7.2

$\mathcal{U}_3^{\mathcal{A}} = {}^{127}/_{462} < \mathcal{U}_3^{\beta} = {}^{2}/_{5}$. Thus, the application $\mathcal{A}_3$ satisfies the necessary conditions stipulated in Equations (6) and (7).

Figure 11 shows the derivation of the response-times of tasks in task set $\mathcal{T}_3^{\beta}$ by construction, assuming a critical instant for all tasks scheduled on a shared EDP resource using FPPS. From this figure, we obtain $WR_{1,3} = 5$, $WR_{2,3} = 9$ and $WR_{3,3} = 21$ from which we conclude that the application $\mathcal{A}_3$ is schedulable since the worst-case response-times of all its tasks do not exceed their deadlines.

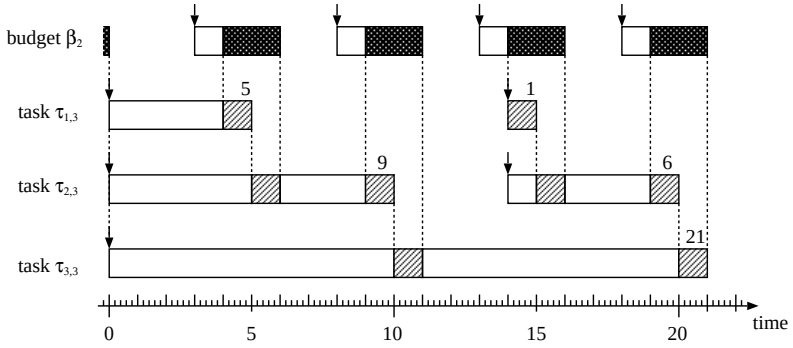

Figure 11: Timeline for a critical instant of all tasks for budget β_3 and application $\mathcal{A}_3$ according to [1, 16].

We convert $\mathcal{A}_3$ to $\widehat{\mathcal{A}_3}$ using the unavailability model transformation and apply Equation (43) to obtain the worst-case response-time upper bounds of 6.5, 13.78 and 33.62 for tasks τ_1, τ_2 and τ_3 respectively from which we conclude that whereas tasks τ_1 and τ_2 are schedulable since they satisfy the sufficient condition, an exact schedulability test is required for τ_3 (see Appendix B.3 for derivations).

Now, we attempt to improve the response-time upper bound of task τ_3 by first combining tasks τ_1 and τ_2 to obtain task $\tau_{\ddagger}$ and then calculating the interference due to the combined task since they have a common period. Assuming $\tau_{\ddagger}$ is the only task in the system, we obtain a situation in which $C_{\ddagger} = C_1 + C_2 = 3$ executes every $T_{\ddagger} = T_1 = T_2 = 14$ time units. The slope of the line equals the utilization of $\tau_{\ddagger}$ i.e. $U_{\ddagger}^{\tau} = U_1^{\tau} + U_2^{\tau} = {}^{3}/_{14}$. Also, point $P(t_0, t_0)$ lies on the line where $t_0 = C_{\ddagger}$. Thus, the linear upper bound of $\tau_{\ddagger}$ is given by

$$I_{\ddagger}^{UB}(t) = U_{\ddagger}^{\tau}(t - t_0) + t_0 = \frac{3}{14}(t - 3) + 3. \tag{44}$$

The worst-case response-time of task τ_3 is then calculated as follows:

$$\begin{aligned} \widehat{R_3^{UB\ddagger}} &= \frac{C_3 + t_0(1 - U_{\ddagger}^{\tau}) + x(1 - U_{\dagger}^{\tau})}{1 - U_{\ddagger}^{\tau} - U_{\dagger}^{\tau}} \\ &= \frac{2 + 3(1 - \frac{3}{14}) + 4(1 - \frac{3}{5})}{1 - \frac{3}{14} - \frac{3}{5}} \\ &= \frac{417}{13} = 32.08, \end{aligned}$$

where $x = \Delta + \Pi - 2\Theta = 4$. We observe that using this improved response-time upper bound, we can directly conclude that τ_3 is schedulable and it is no longer necessary to perform an exact schedulability analysis which is pseudo-polynomial in time complexity. Of course, the response-time upper bounds for tasks τ_1 and τ_2 remain unchanged. However, the result illustrates the potential of this approach to improve the effectiveness of response-time upper bound calculations. Next, we apply the idea to a task set scheduled on a shared resource.

7.1 Improved closed-form (worst-case) response-time upper bounds for tasks having the same period and scheduled on a shared resource

Consider application $\mathcal{A}_4$ consisting of three tasks scheduled on a single shared processor using FPPS. The characteristics of the task set are summarized in Table 7. The two highest priority tasks τ_1 and τ_2 both have the same period of 10. By combining the interference due to these tasks, we show that it is possible to obtain tighter bounds for task τ_3.

task	$T = D$	C	WR	R_i^{UB}, $\sum(\perp)$	R_i^{UB}, $\perp(\sum_{FPPS})$
τ_1	10	4	4	4	4
τ_2	10	3	7	9	9
τ_3	21	2	9	21.67	13.67

Table 7: (Upper bounds of) worst-case response-times for tasks τ_1, τ_2 and τ_3 in application $\mathcal{A}_4$.

Figure 12 shows the derivation of the response-times of tasks in task set $\mathcal{T}_4$ by construction, assuming a critical instant for all tasks. From this figure, we obtain $WR_1 = 4$, $WR_2 = 7$ and $WR_3 = 9$ from which we conclude that the application $\mathcal{A}_4$ is schedulable since the worst-case response-times of all its tasks do not exceed their deadlines.

Using Equation (36), we directly determine the worst-case response-time upper bounds of 4, 9 and 21.67 for tasks τ_1, τ_2 and τ_3 respectively according to the approach proposed in [15] and discussed in Section 6.1. We refer to this method as the *sum of tangents,* $\sum(\perp)$, approach since the total interference of higher priority tasks is derived by summing the linear upper bounds of the interference of each task (i.e. tangents of the demand bound functions), assuming it is the only task in the system. We now consider another approach[4].

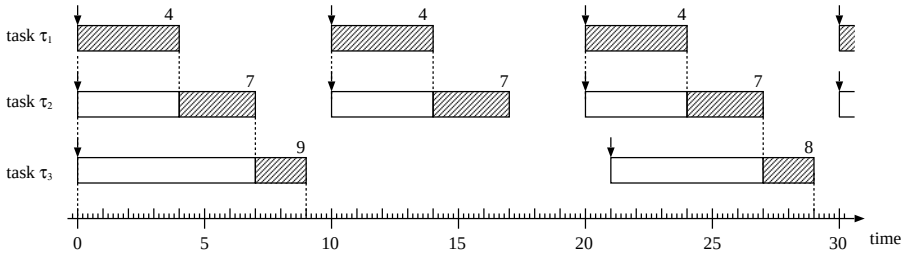

Figure 12: Timeline for a critical instant of all tasks of application $\mathcal{A}_4$.

7.1.1 Tangent of combination approach $\perp(\sum_{FPPS})$

We can go a step further to derive tighter upper bounds by applying FPPS to determine the worst-case interference of the same-period higher priority tasks relative to one another before taking the upper bound. We call this approach *tangent of combination* and add a subscript to denote that our 'summation' is based on FPPS analysis. The interesting observation here is that since these tasks have the same period, we do not need to use the recursive Equation (8) presented in Section 3.1 but a (simplified) linear equivalent of it. Graphically, the approach is illustrated in Figure 13(a) for task τ_3 in our leading example in Table 7. We now derive an equation for the upper bound by the tangent of combination approach for two tasks and then generalize to n tasks having the same period.

[4]Derivation of the response-time upper bounds for both approaches are presented in Appendix B.4

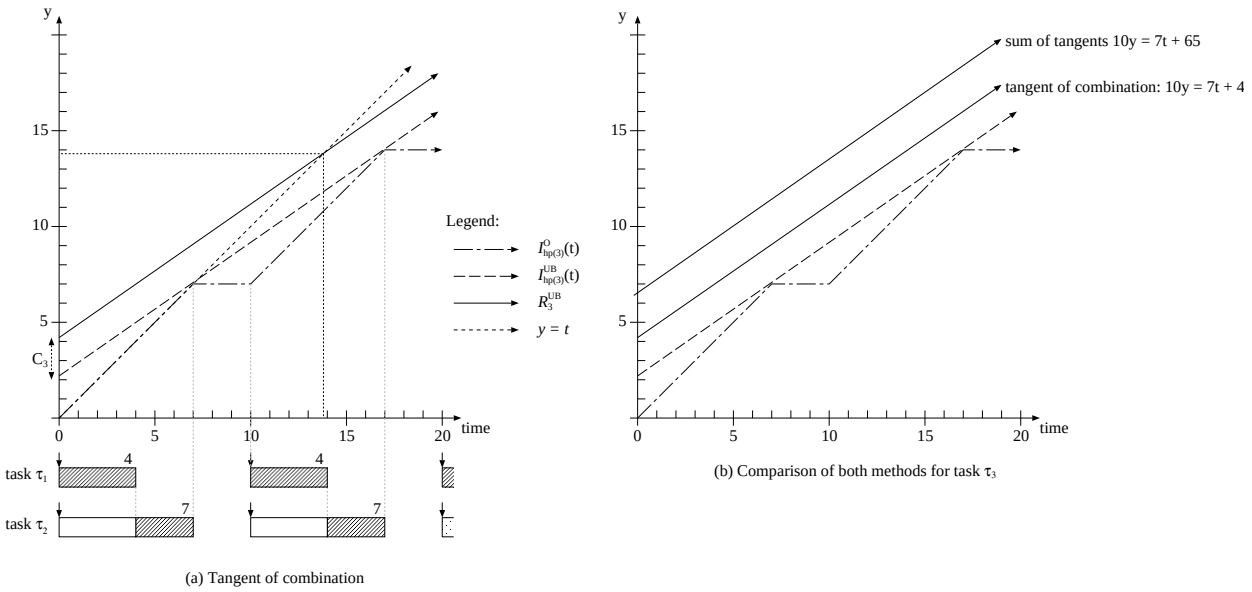

Figure 13: Linear worst-case response-time upper bound of task τ_3 of application $\mathcal{A}_4$ using (a) tangent of combination including (b) a comparison of the two approaches discussed.

Assume tasks τ_x and τ_y are the two (higher priority) tasks having the same period. Then point (C, C) lies on the desired line where $C = C_x + C_y$. The slope of the line is given by the utilization of both tasks combined. Therefore, Equation (45) results:

$$\sum_{j \in hp(i)} I_j^{UB}(t) \quad = \quad \frac{C}{T}(t - C) + C + t \cdot \sum_{k \in Nhp(i)} U_k + \sum_{k \in Nhp(i)} C_k(1 - U_k), \tag{45}$$

where $T = T_x = T_y$, $hp(i)$ denotes all tasks with higher priority than task τ_i and $Nhp(i)$ is the set of higher priority tasks other than τ_x and τ_y (if any) i.e. $hp(i) = Nhp(i) \cup \{\tau_x, \tau_y\}$.

The total occupancy due to task τ_i and all its higher priority tasks, $O_i^{UB}(C_i)$, is then given by the intersection of the lines $y = C_i + \sum_{j \in hp(i)} I_j^{UB}(t)$ and $y = t$.

Supposing there are m higher priority tasks $\tau'_1, \ldots, \tau'_m \in Nhp(i)^{\mathrm{c}}$ with the same period. Then point (C', C') lies on the desired line whose slope is the combined utilization of all the m tasks and $C' = C'_1 + \ldots + C'_m$. Thus, we obtain Equation (46):

$$\sum_{j \in hp(i)} I_j^{UB}(t) \quad = \quad \frac{C'}{T}(t - C') + C' + t \cdot \sum_{k \in Nhp(i)} U_k + \sum_{k \in Nhp(i)} C_k(1 - U_k), \tag{46}$$

where $T' = T'_1 = \cdots = T'_m$, $hp(i)$ denotes all tasks of higher priority than task τ_i of which $Nhp(i)$ is a subset of those tasks having a period different from T with its complement denoted by $Nhp(i)^{\mathrm{c}}$ i.e. $hp(i) = Nhp(i) \cup Nhp(i)^{\mathrm{c}}$.

7.1.2 Comparison of results

Comparing the results in the last two columns of Table 7, we observe improvements in the worst-case response-time upper bounds for task τ_3 by combining tasks τ_1 and τ_2 using tangent of combi-

nation as opposed to using sum of tangents. Furthermore, comparing the corresponding equations for these approaches namely Equation (34) for sum of tangents and Equation (45), we notice that in both cases, the slope of the resulting upper bound is the same. The improvement, therefore, corresponds to a lowering of the intercept with the vertical axis, leading to an earlier intersection with the line $y = t$. Therefore, we come to the conclusion that in general for tasks having the same period, tangent of combination dominates the original sum of tangents approach. By dominating, we imply that the results obtained using the dominant class are *at least as good as* (but typically better than) those of the dominated class. This clearly holds for task τ_3 in our example application $\mathcal{A}_4$ as shown in Figure 13(b).

Finally, we remark that one important feature of the tangent of combination approach described in Section 7.1.1 is that it results in better schedulability analysis *without hampering the linear time bounds*. This claim is obvious from the equations which we derived for this approach.

7.2 Improved closed-form (worst-case) response-time upper bounds for harmonic tasks scheduled on a shared FPPS resource

We now extend the idea presented in Section 7.1 to harmonic tasks and show that it is possible to improve the response-time of lower priority tasks by combining the interference due to higher priority harmonic tasks. A (sub-)set of tasks is said to be harmonic if and only if for any pair of tasks τ_i and τ_j from the set, one is the multiple of the other. Mathematically, this can be expressed in terms of their least common multiple as follows:

$$\underset{i,j \in \mathcal{S}}{\forall} \text{lcm}(T_i,T_j) \in \{T_i, T_j\},$$

where $\mathcal{S}$ is the set of harmonic tasks[5], T_i and T_j are the periods of tasks τ_i and τ_j from $\mathcal{S}$, and $\text{lcm}(T_i,T_j)$ is the *least common multiple* of their periods i.e. the smallest positive integer that is a multiple of both T_i and T_j. An algorithm to extract such subsets from a task set is presented in [22].

task	$T = D$	C	WR	R_i^{UB}, $\sum(\perp)$	R_i^{UB}, $\perp$ $(\sum_{FPPS})$
τ_1	5	2	2	2	2
τ_2	10	3	5	7	7
τ_3	17	2	9	17.67	13.67

Table 8: (Upper bounds of) worst-case response-times for tasks τ_1, τ_2 and τ_3 in application $\mathcal{A}_5$.

We consider yet another example application $\mathcal{A}_5$ consisting of three tasks whose characteristics are presented in Table 8 along with the corresponding response-time upper bounds[6] based on the two approaches discussed in Section 7.1. Tasks τ_1 and τ_2 are harmonic according to the definition of harmonic tasks above. The tangent of combination approach is illustrated graphically in Figure 14(a) for task τ_3 of application $\mathcal{A}_5$. Next, we derive closed-form equations for this approach.

[5]We consider only one set of harmonic tasks. The case involving multiple sets of harmonic tasks is a subject of future work.

[6]Derivation of the response-time upper bounds for both approaches are presented in Appendix B.5

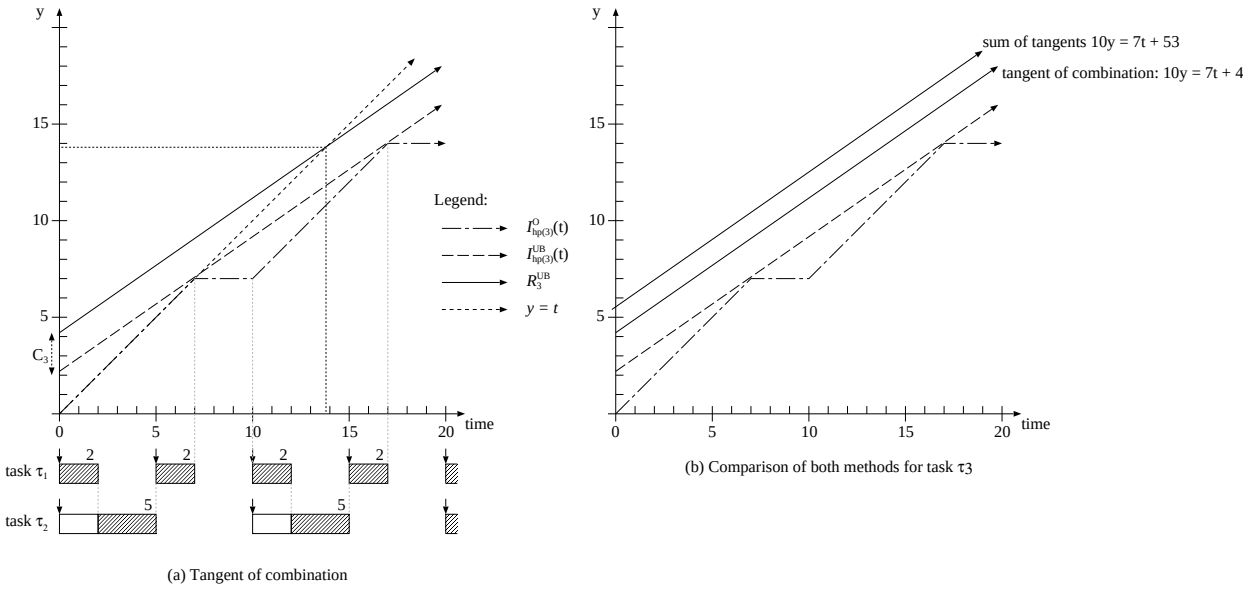

Figure 14: Linear worst-case response-time upper bound of task τ_3 of application $\mathcal{A}_5$ using (a) tangent of combination including (b) a comparison of the two approaches discussed.

7.2.1 Tangent of combination approach $\perp$ ($\sum_{FPPS}$)

Assume tasks τ_x and τ_y are the two (higher priority) harmonic tasks e.g. tasks τ_1 and τ_2 in Figure 14(a). Let $\text{lcm}(T_x, T_y)$ be as previously defined. Then point (C, C) lies on the desired line, the slope of which is given by the utilization of both tasks combined within $\text{lcm}(T_x, T_y)$ where C is the combined demand of tasks τ_x and τ_y within $\text{lcm}(T_x, T_y)$ given by:

$$C = \frac{\text{lcm}(T_x, T_y)}{T_x} C_x + \frac{\text{lcm}(T_x, T_y)}{T_y} C_y. \tag{47}$$

Therefore, Equation (48) results:

$$\sum_{j \in hp(i)} I_j^{UB}(t) \quad = \quad U(t - C) + C + t \cdot \sum_{k \in Nhp(i)} U_k + \sum_{k \in Nhp(i)} C_k(1 - U_k), \tag{48}$$

where $U = \frac{C_x}{T_x} + \frac{C_y}{T_y} = \frac{C_x T_y + C_y T_x}{\text{lcm}(T_x, T_y)}$, $hp(i)$ denotes all tasks with higher priority than task τ_i and $Nhp(i)$ is the set of higher priority tasks other than τ_x and τ_y (if any) i.e. $hp(i) = Nhp(i) \cup \{\tau_x, \tau_y\}$. The total occupancy due to task τ_i and all its higher priority tasks, $O_i^{UB}(C)$, is then given by the intersection of the lines $y = C_i + \sum_{j \in hp(i)} I_j^{UB}(t)$ and $y = t$.

Now suppose we have m higher priority harmonic tasks $\tau'_1, \ldots, \tau'_m \in \mathcal{S}$. The point (C', C') lies on the desired line whose slope is the combined utilization of all the m tasks. Thus, we obtain Equation (49):

$$\sum_{j \in hp(i)} I_j^{UB}(t) \quad = \quad U'_m(t - C') + C' + t \cdot \sum_{k \in Nhp(i)} U_k + \sum_{k \in Nhp(i)} C_k(1 - U_k), \tag{49}$$

where $C' = \frac{\text{lcm}(T'_1,\ldots,T'_m)}{T'_1}C'_1 + \ldots + \frac{\text{lcm}(T'_1,\ldots,T'_m)}{T'_m}C'_m = \text{lcm}(T'_1,\ldots,T'_m)\cdot U'_m$, $U'_m = \frac{C'_1}{T'_1} + \ldots + \frac{C'_m}{T'_m}$, $hp(i)$ denotes all tasks of higher priority than task τ_i and $Nhp(i)$ denote a subset of those tasks not belonging to the set $\mathcal{S}$ of harmonic tasks.

Clearly, (49) is linear in the number of tasks.

7.2.2 Comparison of results

As seen in last two columns of Table 8, the results show an improvement from sum of tangents to tangent of combination as was the case for tasks having the same period. We observe that the definition of harmonic tasks as presented in Section 7.2 ensures that the greatest continuous demand for the resource corresponds to the beginning of every hyperperiod. Therefore, the points used to derive Equation (49) are guaranteed to always lie on the linear upper bound.

Combining two harmonic tasks leads to a pessimistic response-time bound, if computational load is shifted upfront, i.e. by summing up computation times. The existing approach is dominated by our novel approach when the off-set of our interference function is lower, i.e. given $T'_2 = k\cdot T'_1$ and $k \in \mathbb{N}^+$:

$$\begin{aligned} C'_1(1-U'_1) + C'_2(1-U'_2) &> (kC'_1 + C'_2)(1-U'_1-U'_2) \\ kC'_1U'_2 + C'_2U'_1 &> (k-1)(1-U'_1)C'_1 \\ 2C'_1C'_2/T'_1 &> (k-1)(1-U'_1)C'_1 \\ C'_2 &> \frac{1}{2}(k-1)(T'_1 - C'_1) \end{aligned} \tag{50}$$

For $k = 1$ we get $C_2 > 0$, so that the tangent of the combination dominates the sum of tangents [15]. We can guarantee that the combined task of two harmonic tasks leads to the *tightest possible* interference bound, if the following condition holds: $(k-1)C'_1 + C'_2 \geq (k-1)T'_1$. This condition holds by construction for our combined, fictive task $\tau_\dagger$. As a result, when the period of a task is harmonic with its budget, our unavailability approach together with the tangent-of-combined-tasks approach may lead to improved response-time upper bounds. We leave investigations to exploit this novelty to generate timing interfaces [38, 16] as a future work.

Furthermore, we remark that although Equation (49) is linear, we cannot directly conclude that linear time complexity is preserved as the time required to determine which tasks are harmonic also needs to be taken into account (future work).

8 Conclusions

In this document, we considered two-level hierarchical scheduling of independent applications using FPPS for tasks and budgets. Our goal was to unify the analysis with that for single-level FPPS which has been extensively studied. By reusing existing results, we avoid reinventing the wheel as well as the inherent risk of errors as pointed out in some recent works.

We presented an overview of real-time analysis for scheduling of hard real-time tasks using FPPS both in single- and two-level hierarchical frameworks. For the latter, we briefly reviewed existing approaches for which we compared the *worst-case* (i.e. *minimum) available capacity* $WC^\beta(t)$ of a budget β in an interval of length t. Using the EDP resource model as a reference, we presented the unavailability model which converts an application on a shared periodic resource to one on a shared resource by viewing the unavailability of the period resource as the interference due to two fictive highest priority tasks. We derived worst- and best-case response-times for this model and showed both by means of construction and conventional analytical approaches that the model therein presented is not only accurate but equally allows for direct reuse of existing results.

Furthermore, we illustrated the applicability of this novel model by considering the linear response-time upper bound analysis as described in [15]. First, we extended the approach in [15] to also cater for the EDP resource model. Next, we showed that it is equally possible to derive response-time upper bounds by applying the unavailability model. Finally, we proved that the results obtained by both approaches are identical.

In addition, our analysis provided us with useful insights on means to improve response-time upper bounds for tasks with the same period. For a two-level HSF, we illustrated this idea using the unavailability model transformation (though it can easily be applied directly to the EDP model). Moreover, we applied the same approach to a shared resource for which we discussed a new approach for improving worst-case response-time upper bounds of lower priority tasks by combining the interference due to higher priority tasks having the same or harmonic periods. We derived a condition for which our novel tangent of combination approach dominates the sum of tangents approach proposed in [15], and illustrated this effect by means of examples. Moreover, for tasks with the same periods our approach always improves on the existing approach.

9 Future work

In Section 7, we investigated means to improve response-time upper bounds for both tasks scheduled on a shared EDP resource and those scheduled on a shared resource. An interesting extension will be to incorporate activation jitter into the analysis and subsequently, investigate whether the (dominance) relationship established between the two methods is preserved. If that no longer holds, means to select the optimal method for a given scenario can be studied.

Lifting the assumption of a single set of harmonic tasks in Section 7.2 to allow for multiple sets of harmonic tasks and determining the time complexity and criteria to choose the best combination of harmonic tasks in the case where a task belongs to more than one set are also conceivable extensions.

Next to these, another possible direction of future work is to investigate the generation of timing interfaces for the periodic and/or EDP resource model(s) using the unavailability model and tangent of combination approach based on the idea sketched in Section 7.2.2.

Finally, further investigation and consolidation of the observations made in Appendix A are also subjects for future work.

Acknowledgments

We thank Mike Holenderski and Martijn M.H.P. van den Heuvel for discussions and reviews of the manuscript of this document. We also acknowledge the management of the Faculty of Mathematics and Computer Science, Technische Universiteit Eindhoven, for initiating the "Honor in the Master" program in which context this research work was conducted.

References

The page numbers where the references occur (i.e. back references) have been included at the end of each entry.

[1] L. Almeida and P. Peidreiras. Scheduling with temporal partitions: response-time analysis and server design. In *Proc. of the* 4^{th} *ACM International Conference on Embedded Software (EMSOFT)*, pages 95 – 103, September 2004. 6, 14, 15, 16, 18, 24, 31

[2] N.C. Audsley, A. Burns, M.F. Richardson, K. Tindell, and A.J. Wellings. Applying new scheduling theory to static priority pre-emptive scheduling. *Software Engineering Journal*, 8(5):284–292, 1993. 6, 10, 11, 12, 13, 24

[3] P. Balbastre, I. Ripoll, and A. Crespo. Exact response time analysis of hierarchical fixed-priority scheduling. *Proc.* 15^{th} *Int'l Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 315–320, August 2009. 14, 19

[4] E. Bini and S.K. Baruah. Efficient computation of response time bounds under fixed-priority scheduling. In *Proc.* 15^{th} *Int'l Conference on Real-Time and Network Systems (RTNS)*, pages 95–104, March 2007. 7, 24

[5] R.J. Bril. *Real-time scheduling for media processing using conditionally guaranteed budgets*. PhD thesis, Technische Universiteit Eindhoven (TU/e), The Netherlands, July 2004. http://alexandria.tue.nl/extra2/200412419.pdf. 10, 11, 16, 26

[6] R.J. Bril. Towards pragmatic solutions for two-level hierarchical scheduling - part I: A basic approach for independent applications. Technical Report CS Report 07-19, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven (TU/e), The Netherlands, August 2007. http://www.win.tue.nl/ rbril/publications/CSR-07-19.pdf. 1, 7, 19

[7] R.J. Bril, L. Cucu-Grosjean, and J. Goossens. Exact best-case response time analysis of real-time tasks under fixed-priority pre-emptive scheduling for arbitrary deadlines. In *Proc. Work-in-Progress (WIP) session of the* 21^{st} *Euromicro Conference on Real-Time Systems (ECRTS)*, pages 1–4, July 2009. 6, 22

[8] R.J. Bril and P.J.L Cuijpers. Analysis of hierarchical fixed-priority pre-emptive scheduling revisited. Technical Report CS Report 06-36, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven (TU/e), The Netherlands, December 2006. 10

[9] R.J. Bril, E.F.M. Steffens, and W.F.J. Verhaegh. Best-case response times and jitter analysis of real-time tasks. *Journal of Scheduling*, 7(2):133–147, March 2004. 6, 9, 10, 11, 13, 22

[10] R.J. Bril, W.F.J. Verhaegh, and C.C. Wüst. A cognac-glass algorithm for conditionally guaranteed budgets. In *Proc.* 27^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 388–397, December 2006. 6, 14, 16, 18, 19

[11] G. Buttazzo and A. Cervin. Comparative assessment and evaluation of jitter control methods. In *Proc.* 15^{th} *International Conference on Real-Time and Network Systems (RTNS)*, pages 163–172, March 2007. 12, 14

[12] G.C. Buttazzo. *Hard real-time computing systems - predictable scheduling algorithms and applications (*2^{nd} *edition)*. Springer, 2005. 12, 14

[13] R. I. Davis, A. Zabos, and A. Burns. Efficient exact schedulability tests for fixed priority real-time systems. *IEEE Transactions on Computers*, 57(9):1261–1276, September 2008. 24

[14] R.I. Davis and A. Burns. Hierarchical fixed priority pre-emptive scheduling. In *Proc.* 26^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 389–398, December 2005. 6, 10, 14, 16, 18

[15] R.I. Davis and A. Burns. Response time upper bounds for fixed priority real-time systems. In *Proc.* 29^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 407–418, December 2008. 7, 8, 24, 25, 26, 32, 36, 37, 41, 42

[16] A. Easwaran, M. Anand, and I. Lee. Compositional analysis framework using EDP resource models. In *Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International*, pages 129–138, December 2007. 6, 7, 16, 17, 18, 21, 22, 24, 31, 36, 41

[17] M. González Harbour, M.H. Klein, and J.P. Lehoczky. Fixed-priority scheduling with varying execution priority. In *Proc.* 12^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 116–128, December 1991. 6

[18] J. Goossens and R. Devillers. The non-optimality of the monotonic priority assignments for hard real-time offset free systems. *Real-Time Systems*, 13(2):107–126, September 1997. 6, 10

[19] P.K. Harter. Response times in level-structured systems. *ACM Transactions on Computer Systems*, 5(3):232–248, August 1987. 6

[20] M. Joseph and P. Pandya. Finding response times in a real-time system. *The Computer Journal*, 29(5):390–395, 1986. 8, 10, 11, 24

[21] M.H. Klein, T. Ralya, B. Pollak, R. Obenza, and M. González Harbour. *A Practitioner's Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems.* Kluwer Academic Publishers, 1993. 8

[22] T.-W. Kuo and A.K. Mok. Load adjustment in adaptive real-time systems. In *Proc.* 12^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 160–170, December 1991. 34

[23] John P. Lehoczky, Lui Sha, and Jay K. Strosnider. Enhanced aperiodic responsiveness in hard real-time environments. In *Proc.* 8^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 261–270, December 1987. 6

[24] J.P. Lehoczky. Fixed priority scheduling of periodic task sets with arbitrary deadlines. In *Proc.* 11^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 201–209, December 1990. 6

[25] J.Y.T. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance Evaluation*, 2(4):237–250, December 1982. 6

[26] G. Lipari and E. Bini. Resource partitioning among real-time applications. In *Proc.* 15^{th} *Euromicro Conference on Real-Time Systems (ECRTS)*, pages 151–158, July 2003. 6, 14, 16, 18

[27] C.L. Liu and J.W. Layland. Scheduling algorithms for multiprogramming in a real-time environment. *Journal of the ACM*, 20(1):46–61, January 1973. 6, 8, 9, 10

[28] J.L. Lorente, G. Lipari, and E. Bini. A hierarchical scheduling model for component-based real-time systems. In *Proc.* 20^{th} *International Conference on Parallel and Distributed Processing Symposium (IPDPS)*, page 8 pp., April 2006. 6, 20

[29] J. Mäki-Turja and M. Nolin. Fast and tight response-times for tasks with offsets. In *Proc.* 17^{th} *Euromicro Conference on Real-Time Systems (ECRTS)*, pages 127–136, July 2005. 6, 27

[30] C. Mercer, R. Rajkumar, and J. Zelenka. Temporal protection in real-time operating systems. In *Proc.* 11^{th} *IEEE Workshop on Real-Time Operating Systems and Software (RTOSS)*, pages 79–83, May 1994. 6

[31] A.K. Mok, X.A. Feng, and D. Chen. Resource partition for real-time systems. In *Proc.* 7^{th} *Real-Time Technology and Applications Symposium (RTAS)*, pages 75–84, May 2001. 6, 16

[32] A.K.-L. Mok. *Fundamental design problems of distributed systems for the hard-real-time environment.* Phd thesis, Massachusetts Institute of Technology, May 1983. http://www.lcs.mit.edu/publications/pubs/pdf/MIT-LCS-TR-297.pdf. 6

[33] R. Rajkumar, K. Juvva, A. Molano, and S. Oikawa. Resource kernels: A resource-centric approach to real-time and multimedia systems. In *Proc. SPIE, Vol. 3310, Conference on Multimedia Computing and Networking (CMCN)*, pages 150–164, January 1998. 22

[34] O. Redell and M. Sanfridson. Exact best-case response time analysis of fixed priority scheduled tasks. In *Proc.* 14^{th} *Euromicro Conference on Real-Time Systems (ECRTS)*, pages 165–172, June 2002. 6, 9, 10, 11, 13, 22

[35] J. Regehr. Scheduling tasks with mixed preemption relations for robustness to timing faults. In *Proc.* 23^{rd} *IEEE Real-Time Systems Symposium (RTSS)*, pages 315–326, December 2002. 6

[36] S. Saewong, R. Rajkumar, J.P. Lehoczky, and M.H. Klein. Analysis of hierarchical fixed-priority scheduling. In *Proc.* 14^{th} *Euromicro Conference on Real-Time Systems (ECRTS)*, pages 152–160, June 2002. 6, 14, 16, 18

[37] L. Sha, R. Rajkumar, and J.P. Lehoczky. Priority inheritance protocols: an approach to real-time synchronisation. *IEEE Transactions on Computers*, 39(9):1175–1185, September 1990. 6

[38] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *Proc.* 24^{th} *IEEE Real-Time Systems Symposium (RTSS)*, pages 2–13, December 2003. 6, 14, 15, 16, 17, 18, 24, 26, 36

[39] J.K. Strosnider, J.P. Lehoczky, and L. Sha. The deferrable server algorithm for enhanced aperiodic responsiveness in hard real-time environments. *IEEE Transactions on Computers*, 44(1):73–91, January 1995. 6

[40] K.W. Tindell, A. Burns, and A.J. Wellings. An extendible approach for analyzing fixed priority hard real-time tasks. *Real-Time Systems*, 6(2):133–151, 1994. 6, 22

[41] A.M. van Renssen, S.F. Geuns, J.P.H.M. Hausmans, W. Poncin, and R.J. Bril. On utilization bounds for a periodic resource under rate monotonic scheduling. In *Proc. Work-in-Progress (WiP) session of the* 21^{st} *Euromicro Conference on Real-Time Systems (ECRTS)*, pages 25–28, July 2009. 6

[42] Y. Wand and M. Saksena. Scheduling fixed-priority tasks with preemption threshold. In *Proc.* 6^{th} *International Conference on Real-Time Computing Systems and Applications (RTCSA)*, pages 328–335, December 1999. 6

A Improved and/or faster schedulability analysis using the unavailability model

In Sections 6.1 and 6.3, we presented two methods for calculating response-time upper bounds of tasks which we subsequently showed to be equivalent. In applying the method based on the unavailability model, we can take advantage of the fact that the characteristics of the fictive tasks depend *solely* on the characteristics of the budget β_α to further reduce pessimism in calculating the *worst-case interference* due to $\tau_\dagger$ as follows: Rather than using the *(linear) worst-case interference upper bound* $I_\dagger^{UB}(t)$, we use the exact *worst-case interference* $I_\dagger^{O}(t)$ for the combined fictive task $\tau_\dagger$, both depicted in Figure 10, again noting that this worst-case scenario is based on the assumption that the task is the *only* one in the system just as in [15].

The procedure is outlined as follows:

Step 1: Using Equation (37) on A_a (or Equation (43) on $\widehat{\mathcal{A}_a}$ since they are equivalent), we determine the worst-case occupied time of task τ_i, say t_0.

Step 2: Using Equation (51), we calculate the *exact* worst-case interference due to task $\tau_\dagger$ at the time t_0 determined in step 1. This can be determined in a manner similar to the supply bound function in [16] and based on Figure 10 is given by:

$$I_\dagger^{O}(t) \quad = \quad \begin{cases} t & 0 < t \leq x \\ x + y(\Pi - \Theta) + max\{0, t - x - y\Pi - \Theta\} & otherwise \end{cases}, \tag{51}$$

where $x = C_{\text{-}1} + 2C_0 = (\Pi - \Delta) + 2(\Delta - \Theta) = \Pi + \Delta - 2\Theta$ and $y = \left\lfloor \frac{t-x}{\Pi} \right\rfloor$.

Step 3: Next, we recalculate the worst-case occupied time of task τ_i using the following equation in which we now use the exact worst-case interference of $\tau_\dagger$ with the unavailability model from Step 2:

$$\widehat{R_i^{UB\dagger}}(t) = \frac{C_i + \sum_{1 \leq j < i} \left(U_j^\tau A J_j + C_j(1 - U_j^\tau) \right) + I_\dagger^{O}(t)}{1 - \sum_{1 \leq j < i} U_j^\tau}, \tag{52}$$

Step 4: (Optional) Iteratively apply steps 2 and 3 until two consecutive values of $\widehat{R_i^{UB\dagger}}(t)$ are the same.

In essence, referring again to Figure 10, the gain resulting from this method corresponds to the region between the two curves i.e. $I_\dagger^{UB}(t) - I_\dagger^{O}(t)$ which is always greater than zero except at the points where the two curves intersect. Given a system of several applications, $\Pi \gg \Theta$ and the gain becomes even more significant (cf. Equation (42)).

We now apply this idea to our previous example application $\mathcal{A}_2$ whose characteristics were summarized in Table 5 (and reproduced in Table 9). The application is served by budget β_2 having parameters as follows: $\Pi = 5$, $\Theta = 2$ and $\Delta = 3$.

task	$T = D$	C	WR	$R_i^{UB}(\Omega)$	$\widehat{R_i^{UB\dagger}}(t)$
τ_1	7	1	5	6.5	5.5
τ_2	20	4	20	25.11	24.33

Table 9: (Upper bounds of) worst-case response-times for tasks τ_1 and τ_2 in application $\mathcal{A}_2$ ($\widehat{\mathcal{A}_2}$) based on proposed extension.

In step 1, we calculate $R_1^{UB}(\Omega) = 6.5$ and $R_2^{UB}(\Omega) = 25.11$ using Equation (37). Next, we determine $I_\dagger^{O}(6.5) = 4.5$ and $I_\dagger^{O}(25.11) = 4 + 12 + 0 = 16$ as stipulated in step 2. Finally,

using Equation (52) in step 3, we get response-time upper bounds of 5.5 and 24.33 for τ_1 and τ_2 respectively. The results show that the proposed method produces *tighter* bounds in general and in the worst case, coincides with $R_i^{UB}(\Omega)$ (i.e. at points where $I_\dagger^{UB}(t) = I_\dagger^{O}(t)$ such as point $P(t_0, y_0)$ in Figure 10).

However, the results also indicate that by terminating the procedure at step 3, we only reduce the pessimism due to the fictive tasks. Otherwise, being the highest priority task, the response-time upper bound of task τ_1, $\widehat{R_1^{UB\dagger}}(t)$, ought to coincide with the exact worst case response-time. But owing to the fact that we start the procedure using $lsbf_\Omega(t)$, the value of t obtained in step 1 may be larger than the actual demand due to task τ_1 (and the higher priority fictive tasks) and therefore results in some residual pessimism.

To further reduce the pessimism due to the fictive tasks, we apply step 4 as described previously. In the second iteration for task τ_1, we calculate $I_\dagger^{O(2)}(5.5) = 4$. Substitution this value into Equation (52), we obtain $\widehat{R_1^{UB\dagger}}^{(2)}(t) = 5$. Repeating steps 2 and 3 again, we calculate $I_\dagger^{O(3)}(5) = 4$; $\widehat{R_1^{UB\dagger}}^{(3)}(t) = 5$ at which point the procedure terminates, giving the expected result i.e. the exact worst-case response-time of task τ_1. The same approach can be performed to task τ_2 and all lower priority tasks in general though we note that it will not yield the exact worst-case response-time for those tasks due to the pessimism resulting from approximating the interference of higher priority tasks by the linear upper bound function given by Equation (38). Unfortunately, the complexity of the analysis increases and perhaps it may be better to revert to the classical response-time analysis at this stage especially because this refinement may not work in which case an exact analysis will still be required.

Given the duality of application $\mathcal{A}_\alpha$ on a shared EDP resource and the corresponding application $\widehat{\mathcal{A}_\alpha}$ using the unavailability model, we expect an equivalent or comparable method (when iterative step 4 is included) using the supply bound function $sbf_\Omega(t)$ and the original task set $\mathcal{T}_\alpha^\beta$ of application $\mathcal{A}_\alpha$. The proposed method involves finding the intersection between the line $y = sbf_\Omega(t)$ and Equation (35) which can be obtained by a recursive relation as sketched in Section 4.1. Nonetheless, whereas the method based on the unavailability model approaches the improved response-time upper bound from above (as shown in the preceding example), the $sbf_\Omega(t)$ method approaches it from below. Therefore, it may be the case that the unavailability approach will require fewer iterations since the process might terminate early when, for instance, the sufficient condition is met after the first iteration of step 3 (i.e. without having to perform step 4 at all). However, we note that the unavailability model is not guaranteed to yield the tightest (i.e. smallest) upper bound. For instance, in the example in Table 9, the $sbf_\Omega(t)$ approach yields a tighter bound of 21 (obtained by construction) compared to 24.33 using the unavailability approach. Therefore, there may be a trade off between quicker schedulability analysis for the latter and tighter bounds for the former.

In summary, this discussion reveals two important points:

1. By performing steps 1 to 3 only, we can obtain tighter response-time upper bounds (compared to the original approach in [15]) in linear time which *may not* be paralleled by using the conventional $(l)sbf_\Omega(t)$ method without hampering the linear time bound.

2. By including the optional step 4, an iterative procedure is required. However, the unavailability method may outperform the $(l)sbf_\Omega(t)$ method in terms of the number of iterations required to establish schedulability.

We present this discussion in an appendix as it is not yet sufficiently developed. Further investigations, particularly of the two claims above, are possible directions for future work. To conclude, we outline the steps performed in determining $\widehat{R_2^{UB\dagger}}(t)$.

Step 1: $R_2^{UB}(\Omega) = 25.11$

Step 2: $I_{\dagger}^{O(1)}(25.11) = 4 + 12 + 0 = 16$

Step 3: $\widehat{R_2^{UB\dagger}}^{(1)}(t) = \dfrac{4 + 1(1 - \frac{1}{7}) + 16}{1 - \frac{1}{7}} = \frac{146}{6} = 24.33$

Step 4(2b): $I_{\dagger}^{O(2)}(24.33) = 4 + 12 + 0 = 16$

Step 4(3b): $\widehat{R_2^{UB\dagger}}^{(2)}(t) = \dfrac{4 + 1(1 - \frac{1}{7}) + 16}{1 - \frac{1}{7}} = \frac{146}{6} = 24.33$ ∎

B Derivation of equations

This appendix contains the derivation of (some of) the results presented in this document.

B.1 Worst- and best-case response-time analysis using unavailability model

In Section 5, we presented equations to calculate the worst- and best-case response-times of tasks using the unavailability model. We now apply these equations to our example task set whose characteristics were presented in Table 3. For the sake of convenience, we reproduce that task set including the fictive tasks τ_{-1} and τ_0 in Table 10. The server characteristics are $\Pi = 5$, $\Theta = 2$ and $\Delta = 3$ as presented in Table 4.

task	T	D	C	AJ	φ	WR	BR
τ_{-1}	5	2	2	0	1	-	-
τ_0	5	1	1	2	-	-	-
τ_1	7	7	1	0	-	5	1
τ_2	20	20	4	0	-	20	10

Table 10: Characteristics of $\widehat{\tau_2^{\beta}}$ with worst-case and best-case response-times of tasks.

For worst-case response-time analysis, we use Equation (29), i.e.

$$wr_i^{(l+1)} = WC_i + \left\lceil \frac{x - \varphi_{-1}}{T_{-1}} \right\rceil WC_{-1} + \left\lceil \frac{x + AJ_0}{T_0} \right\rceil WC_0 + \sum_{1 \le j < i} \left\lceil \frac{x + AJ_j}{T_j} \right\rceil WC_j.$$

We use WC_i as a starting value for the iteration.

Task τ_1:

$$\begin{aligned}
wr_1^{(0)} &= 1 \\
wr_1^{(1)} &= 1 + \left\lceil \frac{1-1}{5} \right\rceil 2 + \left\lceil \frac{1+2}{5} \right\rceil 1 = 1 + 0 + 1 = 2 \\
wr_1^{(2)} &= 1 + \left\lceil \frac{2-1}{5} \right\rceil 2 + \left\lceil \frac{2+2}{5} \right\rceil 1 = 1 + 2 + 1 = 4 \\
wr_1^{(3)} &= 1 + \left\lceil \frac{4-1}{5} \right\rceil 2 + \left\lceil \frac{4+2}{5} \right\rceil 1 = 1 + 2 + 2 = 5 \\
wr_1^{(4)} &= 1 + \left\lceil \frac{5-1}{5} \right\rceil 2 + \left\lceil \frac{5+2}{5} \right\rceil 1 = 1 + 2 + 2 = 5
\end{aligned}$$

$WR_1 = 5 < 7 (= D_1)$. Hence, task τ_1 is schedulable.

Task τ_2:

$$\begin{aligned}
wr_2^{(0)} &= 4 \\
wr_2^{(1)} &= 4 + \left\lceil \frac{4-1}{5} \right\rceil 2 + \left\lceil \frac{4+2}{5} \right\rceil 1 + \left\lceil \frac{4}{7} \right\rceil 1 = 4+2+2+1 = 9 \\
wr_2^{(2)} &= 4 + \left\lceil \frac{9-1}{5} \right\rceil 2 + \left\lceil \frac{9+2}{5} \right\rceil 1 + \left\lceil \frac{9}{7} \right\rceil 1 = 4+4+3+2 = 13 \\
wr_2^{(3)} &= 4 + \left\lceil \frac{13-1}{5} \right\rceil 2 + \left\lceil \frac{13+2}{5} \right\rceil 1 + \left\lceil \frac{13}{7} \right\rceil 1 = 4+6+3+2 = 15 \\
wr_2^{(4)} &= 4 + \left\lceil \frac{15-1}{5} \right\rceil 2 + \left\lceil \frac{15+2}{5} \right\rceil 1 + \left\lceil \frac{15}{7} \right\rceil 1 = 4+6+4+3 = 17 \\
wr_2^{(5)} &= 4 + \left\lceil \frac{17-1}{5} \right\rceil 2 + \left\lceil \frac{17+2}{5} \right\rceil 1 + \left\lceil \frac{17}{7} \right\rceil 1 = 4+8+4+3 = 19 \\
wr_2^{(6)} &= 4 + \left\lceil \frac{19-1}{5} \right\rceil 2 + \left\lceil \frac{19+2}{5} \right\rceil 1 + \left\lceil \frac{19}{7} \right\rceil 1 = 4+8+5+3 = 20 \\
wr_2^{(7)} &= 4 + \left\lceil \frac{20-1}{5} \right\rceil 2 + \left\lceil \frac{20+2}{5} \right\rceil 1 + \left\lceil \frac{20}{7} \right\rceil 1 = 4+8+5+3 = 20
\end{aligned}$$

$WR_2 = 20 = D_2$. Hence, task τ_2 is (exactly) schedulable.

Likewise, for best-case response-time analysis, we use Equation (30), i.e.

$$br_i^{(l+1)} = BC_i + \left(\left\lceil \frac{x+\varphi_{\text{-}1}}{T_{\text{-}1}} \right\rceil - 1 \right) BC_{\text{-}1} + \left(\left\lceil \frac{x - AJ_0}{T_0} \right\rceil - 1 \right)^+ BC_0 + \sum_{1 \le j < i} \left(\left\lceil \frac{x - AJ_j}{T_j} \right\rceil - 1 \right)^+ BC_j$$

We use WR_i as a starting value for the iteration.

Task τ_1:

$$\begin{aligned}
br_1^{(0)} &= 5 \\
br_1^{(1)} &= 1 + \left(\left\lceil \frac{5+1}{5} \right\rceil - 1 \right) 2 + \left(\left\lceil \frac{5-2}{5} \right\rceil - 1 \right)^+ 1 = 1+2+0 = 3 \\
br_1^{(2)} &= 1 + \left(\left\lceil \frac{3+1}{5} \right\rceil - 1 \right) 2 + \left(\left\lceil \frac{3-2}{5} \right\rceil - 1 \right)^+ 1 = 1+0+0 = 1 \\
br_1^{(3)} &= 1 + \left(\left\lceil \frac{1+1}{5} \right\rceil - 1 \right) 2 + \left(\left\lceil \frac{1-2}{5} \right\rceil - 1 \right)^+ 1 = 1+0+0 = 1
\end{aligned}$$

Therefore, $BR_1 = 1$.

Task τ_2:

$$
\begin{aligned}
br_2^{(0)} &= 20 \\
br_2^{(1)} &= 4 + \left(\left\lceil \frac{20+1}{5} \right\rceil - 1\right) 2 + \left(\left\lceil \frac{20-2}{5} \right\rceil - 1\right)^+ 1 + \left(\left\lceil \frac{20}{7} \right\rceil - 1\right)^+ 1 = 4+8+3+2 = 17 \\
br_2^{(2)} &= 4 + \left(\left\lceil \frac{17+1}{5} \right\rceil - 1\right) 2 + \left(\left\lceil \frac{17-2}{5} \right\rceil - 1\right)^+ 1 + \left(\left\lceil \frac{17}{7} \right\rceil - 1\right)^+ 1 = 4+6+2+2 = 14 \\
br_2^{(3)} &= 4 + \left(\left\lceil \frac{14+1}{5} \right\rceil - 1\right) 2 + \left(\left\lceil \frac{14-2}{5} \right\rceil - 1\right)^+ 1 + \left(\left\lceil \frac{14}{7} \right\rceil - 1\right)^+ 1 = 4+4+2+1 = 11 \\
br_2^{(4)} &= 4 + \left(\left\lceil \frac{11+1}{5} \right\rceil - 1\right) 2 + \left(\left\lceil \frac{11-2}{5} \right\rceil - 1\right)^+ 1 + \left(\left\lceil \frac{11}{7} \right\rceil - 1\right)^+ 1 = 4+4+1+1 = 10 \\
br_2^{(5)} &= 4 + \left(\left\lceil \frac{10+1}{5} \right\rceil - 1\right) 2 + \left(\left\lceil \frac{10-2}{5} \right\rceil - 1\right)^+ 1 + \left(\left\lceil \frac{10}{7} \right\rceil - 1\right)^+ 1 = 4+4+1+1 = 10
\end{aligned}
$$

Therefore, $BR_2 = 10$.

B.2 Derivation of (linear) response-time upper bounds using $R_{i,\alpha}^{UB}(\Omega)$

We present the analytical derivation of the response-time upper bounds for the task set presented in Table 5 using the linear supply bound function approach defined by Equation (37) i.e.

$$R_{i,\alpha}^{UB}(\Omega) = \frac{C_i + \sum_{1 \le j < i} \left(U_j^\tau A J_j + C_j(1 - U_j^\tau)\right) + U_\alpha^\beta(\Pi + \Delta - 2\Theta)}{U_\alpha^\beta - \sum_{1 \le j < i} U_j^\tau},$$

for $\Pi = 5$, $\Theta = 2$ and $\Delta = 3$.

Task τ_1:

$$R_{1,2}^{UB}(\Omega) = \frac{1 + \frac{2}{5}(5+3-2(2))}{\frac{2}{5}} = \frac{13}{2} = 6.5$$

Task τ_2:

$$R_{2,2}^{UB}(\Omega) = \frac{4 + 1(1 - \frac{1}{7}) + \frac{2}{5}(5+3-2(2))}{\frac{2}{5} - \frac{1}{7}} = \frac{226}{9} = 25.11$$

For the unavailability approach, we use Equation (39) i.e.

$$R_i^{UB} = \frac{C_i + \sum_{-1 \le j < i} \left(U_j^\tau A J_j + C_j(1 - U_j^\tau)\right) - \varphi_{\text{-}1} U_{\text{-}1}^\tau}{1 - \sum_{-1 \le j < i} U_j^\tau}.$$

Task τ_1:

$$\widehat{R_1^{UB}} = \frac{1 + \left(2(1 - \frac{2}{5})\right) + \left(\frac{1}{5}(2) + 1(1 - \frac{1}{5})\right) - 1\left(\frac{2}{5}\right)}{1 - \frac{2}{5} - \frac{1}{5}} = \frac{15}{2} = 7.5$$

Task τ_2:

$$\widehat{R_2^{UB}} = \frac{4 + \left(2(1-\frac{2}{5})\right) + \left(\frac{1}{5}(2) + 1(1-\frac{1}{5})\right) + 1(1-\frac{1}{7}) - 1\left(\frac{2}{5}\right)}{1-\frac{2}{5}-\frac{1}{5}-\frac{1}{7}} = \frac{240}{9} = 26.67$$

B.3 Derivation of (linear) response-time upper bounds using $R_i^{UB\dagger}(C)$

We present the analytical derivation in of the response-time upper bounds for the task set presented in Table 6 using Equation (43) from the transformation to the unavailability model as described in Section 6.3 i.e.

$$R_i^{UB\dagger} = \frac{C_i + \sum_{1\leq j<i}\left(U_j^\tau AJ_j + C_j(1-U_j^\tau)\right) + x(1-U_\dagger^\tau)}{1-U_\dagger^\tau - \sum_{1\leq j<i} U_j^\tau},$$

where $x = \Delta + \Pi - 2\Theta$, $\Pi = 5$, $\Theta = 2$ and $\Delta = 3$.

Task τ_1:

$$\widehat{R_1^{UB\dagger}} = \frac{1 + 4(1-\frac{3}{5})}{1-\frac{3}{5}} = \frac{13}{2} = 6.5$$

Task τ_2:

$$\widehat{R_2^{UB\dagger}} = \frac{2 + 1(1-\frac{1}{14}) + 4(1-\frac{3}{5})}{1-\frac{3}{5}-\frac{1}{14}} = \frac{317}{23} = 13.78$$

Task τ_3:

$$\widehat{R_3^{UB\dagger}} = \frac{2 + 1(1-\frac{1}{14}) + 2(1-\frac{2}{14}) + 4(1-\frac{3}{5})}{1-\frac{3}{5}-\frac{1}{14}-\frac{2}{14}} = \frac{437}{13} = 33.62$$

B.4 Derivation of (linear) response-time upper bounds for tasks having the same period (Application $\mathcal{A}_4$)

This appendix contains the analytic derivation of the (worst-case) response-time upper bounds for tasks in application $\mathcal{A}_4$ whose characteristics were presented in Table 7 along with a summary of the results.

B.4.1 Sum of tangents $\sum(\perp)$ approach

Task τ_1: Since task τ_1 is the highest priority task, $R_1^{UB} = WR_1 = C_1 = 4$.

Task τ_2:

$$I_1^{UB}(t) = U_1 t + C_1(1-U_1) = \frac{4}{10}t + \frac{24}{10}$$

$$ldbf_2(t) = I_1^{UB}(t) + C_2 = \frac{4}{10}t + \frac{54}{10}$$

$$R_2^{UB} : t = \frac{4}{10}t + \frac{54}{10} \Rightarrow t = 9$$

Task τ_3:

$$I_1^{UB}(t) = U_1 t + C_1(1-U_1) = \frac{4}{10}t + \frac{24}{10}$$

$$I_2^{UB}(t) = U_2 t + C_2(1-U_2) = \frac{3}{10}t + \frac{21}{10}$$

$$ldbf_3(t) = I_1^{UB}(t) + I_2^{UB}(t) + C_3 = \frac{7}{10}t + \frac{45}{10} + 2$$

$$R_3^{UB} : t = \frac{7}{10}t + \frac{65}{10} \Rightarrow t = 21.67$$

B.4.2 Tangent of combination$\perp(\sum_{FPPS})$ approach

Tasks τ_1 and τ_2: Same as in sum of tangents.

Task τ_3: From Figure 13(a), points $p_1(7,7)$ and $p_2(17,14)$ lie on the required line $I^{UB}_{\{1 \bigcup 2\}}(t)$; gradient $= \frac{7}{10}$.

$$I^{UB}_{\{1 \bigcup 2\}}(t) = \frac{7}{10}(t-7) + 7 = \frac{7}{10}t + \frac{21}{10}$$

$$ldbf_3(t) = I^{UB}_{\{1 \bigcup 2\}}(t) + C_3 = \frac{7}{10}t + \frac{41}{10}$$

$$R_3^{UB} : t = \frac{7}{10}t + \frac{41}{10} \Rightarrow t = 13.67$$

B.5 Derivation of (linear) response-time upper bounds for harmonic tasks (Application $\mathcal{A}_5$)

This appendix contains the analytic derivation of the (worst-case) response-time upper bounds for tasks in application $\mathcal{A}_5$ whose characteristics were presented in Table 8 along with a summary of the results.

B.5.1 Sum of tangents $\sum(\perp)$ approach

Task τ_1: Since task τ_1 is the highest priority task, $O_i^{UB}(C) = WR_1 = C_1 = 2$.

Task τ_2:

$$I_1^{UB}(t) = U_1 t + C_1(1-U_1) = \frac{2}{5}t + \frac{6}{5}$$

$$ldbf_2(t) = I_1^{UB}(t) + C_2 = \frac{2}{5}t + \frac{21}{5}$$

$$R_2^{UB} : t = \frac{2}{5}t + \frac{21}{5} \Rightarrow t = 7$$

Task τ_3:

$$
\begin{aligned}
I_1^{UB}(t) &= U_1 t + C_1(1-U_1) = \tfrac{2}{5}t + \tfrac{6}{5} \\
I_2^{UB}(t) &= U_2 t + C_2(1-U_2) = \tfrac{3}{10}t + \tfrac{21}{10} \\
ldbf_3(t) &= I_1^{UB}(t) + I_2^{UB}(t) + C_3 = \tfrac{7}{10}t + \tfrac{33}{10} + 2 \\
R_3^{UB} &: \quad t = \tfrac{7}{10}t + \tfrac{53}{10} \Rightarrow t = 17.67
\end{aligned}
$$

B.5.2 Tangent of combination$\perp(\sum_{FPPS})$ approach

Tasks τ_1 and τ_2: Same as in sum of tangents.

Task τ_3: From Figure 14(a), points $p_1(7,7)$ and $p_2(17,14)$ lie on the required line $I^{UB}_{\{1 \bigcup 2\}}(t)$; gradient $= \frac{7}{10}$.

$$
\begin{aligned}
I^{UB}_{\{1 \bigcup 2\}}(t) &= \tfrac{7}{10}(t-7) + 7 = \tfrac{7}{10}t + \tfrac{21}{10} \\
ldbf_3(t) &= I^{UB}_{\{1 \bigcup 2\}}(t) + C_3 = \tfrac{7}{10}t + \tfrac{41}{10} \\
R_3^{UB} &: \quad t = \tfrac{7}{10}t + \tfrac{41}{10} \Rightarrow t = 13.67
\end{aligned}
$$

Index

EDP resource model, 17
- Linear supply bound function, 18
- Supply bound function, 18

FPPS, 8, 10
- Best-case response-time analysis, 11, 13
- Critical instant, 9, 10, 13
- Jitter analysis, 12
- Optimal instant, 9, 11, 13
- Schedulability conditions, 9
- Utilization, 9
- Worst-case response-time analysis, 10, 13

H-FPPS, 14
- Best-case available capacity of a(n) EDP budget, 19
- Best-case response-time analysis of tasks, 19
- Critical instant, 15
- Global scheduler, 14
- Local scheduler, 14
- Locality of scheduling analysis, 14
- Optimal instant, 19
- Worst-case available capacity of a budget, 16
- Worst-case response-time analysis of tasks, 15
- Worst-case start time of a budget, 16

Improved response-time upper bounds, 30
- Harmonic tasks, 34
- Least common multiple, 34
- Same-period tasks, 32
- Tangent of combination approach, 32

Periodic resource model, 17
- Linear supply bound function, 17
- Supply bound function, 17

Response-time upper bounds, 24
- FPPS, 25
- H-FPPS
  - Linear supply bound function, 26
  - Unavailability model approach, 27

Unavailability model, 20
- Assumptions for resource provisioning, 22
- Best-case response-time analysis, 22
- Worst-case response-time analysis, 22