Existing worst-case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption is too optimistic

Citation for published version (APA):

Document status and date:
Published: 01/01/2006

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 04. Sep. 2020
Existing worst-case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption is too optimistic

Reinder J. Bril
Technische Universiteit Eindhoven (TU/e),
Department of Mathematics and Computer Science,
Group System Architecture and Networking (SAN),
Den Dolech 2, 5600 AZ Eindhoven, The Netherlands
r.j.bril@tue.nl

Abstract

This paper revisits response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption (FPDS), arbitrary phasing, and deadlines within periods. We show that existing worst-case response time analysis, as presented in [5, 6, 7], is too optimistic. In particular, the worst-case response time of a task is not necessarily assumed for the first job of that task when released at an ε-critical instant. We also show that existing best-case response time analysis [4] indeed yields a lower bound.

1 Introduction

Based on the seminal paper of Liu and Layland [14], many results have been achieved in the area of analysis for fixed-priority preemptive scheduling (FPPS). Arbitrary preemption of real-time tasks has a number of drawbacks, though. In particular in systems using cache memory, e.g. to bridge the speed gap between processors and main memory, arbitrary preemptions induce additional cache flushes and reloads. As a consequence, system performance and predictability are degraded, which complicates system design, analysis and testing [7, 9, 12, 16]. Although fixed-priority non-preemptive scheduling (FPNS) may resolve these problems, it generally leads to reduced schedulability compared to FPPS. Therefore, alternative scheduling schemes have been proposed between the extremes of arbitrary preemption and no preemption. These schemes are also known as deferred preemption or co-operative scheduling [6], and are denoted by fixed-priority scheduling with deferred preemption (FPDS) in the remainder of this paper.

Worst-case response time analysis of periodic real-time tasks under FPDS, arbitrary phasing, and deadlines with periods has been addressed in a number of papers [5, 6, 7, 12]. In this paper, we will show that the existing analysis is not exact. Whereas it has been shown in [5] that the analysis presented in [6, 7, 12] is pessimistic, we will show by means of an example consisting of just two tasks that the analysis presented in [5, 6, 7] is optimistic. We explore the example by considering best-case and worst-case response times under FPDS as a function of the relative phasing between the tasks. The exploration reveals that, although the example refutes the existing analysis, it does not refute the conjecture in [5] about an ε-critical instant. Concerning best-case response time analysis, we found that a job that experiences a Δ-optimal instant [4] may not be able to immediately start executing upon its activation. As a consequence, the best-case response time analysis under FPDS and arbitrary phasing as presented in [4] indeed yields a lower bound. This is a similar result as presented in [15] for FPPS with arbitrary phasing and deadlines greater than periods.

This paper is organized as follows. Section 2 briefly describes a basic real-time scheduling model for FPPS and refined model for FPDS. Response time analysis for FPPS and FPDS is recapitulated in Section 3, and response times for FPDS are expressed in terms of response times for FPPS. In Section 4, we present an example that refutes existing worst-case response time analysis under FPDS. We subsequently explore the example by considering response times under both FPPS and FPDS. Section 5 discusses the results of the exploration and presents topics of current investigation. The paper is concluded in Section 6.

2 Scheduling models

This section briefly describes a basic real-time scheduling model for FPPS and a refined model for FPDS. Most of the definitions and assumptions of these models are taken from [2], and originate from [14].
2.1 Basic model for FPPS

We assume a single processor and a set \( T \) of \( n \) periodically released, independent tasks \( \tau_1, \tau_2, \ldots, \tau_n \). At any moment in time, the processor is used to execute the highest priority task that has work pending. So, when task \( \tau_i \) is being executed, and a release occurs for a higher priority task \( \tau_j \), then the execution of \( \tau_i \) is preempted, and will resume when the execution of \( \tau_j \) has ended, as well as all other releases of tasks with a higher priority than \( \tau_i \) that have taken place in the meantime.

Each task \( \tau_i \) is characterized by a (release) period \( T_i \in \mathbb{R}^+ \), a computation time \( C_i \in \mathbb{R}^+ \), a (relative) deadline \( D_i \in \mathbb{R}^+ \), where \( C_i = \min(D_i, T_i) \), and a phasing \( \phi_i \in \mathbb{R} \). An activation (or release) time is a time at which a task \( \tau_i \) becomes ready for execution. A release of a task is also termed a job. The job of task \( \tau_i \) with release time \( \phi_i \) serves as a reference activation, and is referred to as job zero. The release of job \( k \) of \( \tau_i \) therefore takes place at time \( a_{ik} = \phi_i + kT_i, \ k \in \mathbb{Z} \). The deadline of job \( k \) of \( \tau_i \) takes place at \( d_{ik} = a_{ik} + D_i \). The set of phasings \( \phi_i \) is termed the phasing \( \phi \) of the task set \( T \).

The active (or response) interval of job \( k \) of \( \tau_i \) is defined as the time span between the activation time of that job and its completion time \( c_{ik} \), i.e. \( [a_{ik}, c_{ik}] \). The response time \( r_{ik} \) of job \( k \) of \( \tau_i \) is defined as the length of its active interval, i.e. \( r_{ik} = c_{ik} - a_{ik} \). Figure 1 illustrates the above basic notions for an example job of task \( \tau_i \).

The worst-case response time \( WR_i \) of a task \( \tau_i \) is the largest response time of any of its jobs, i.e.

\[
WR_i = \sup_{q,k} r_{ik}.
\]

A critical instant of a task is defined to be an (hypothetical) instant that leads to the worst-case response time for that task. Typically, such an instant is described as a point in time with particular properties. As an example, a critical instant for tasks under FPPS is given by a point in time for which all tasks have a simultaneous release. The best-case response time \( BR_i \) of task \( \tau_i \) is its shortest response time, i.e.

\[
BR_i = \inf_{q,k} r_{ik}.
\]

An optimal instant of a task is defined to be an (hypothetical) instant that leads to the best-case response time.

We assume that we do not have control over the phasing \( \phi \), 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 [10, 11, 14]. We also assume other standard basic assumptions [14], 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 \( \tau_i \) 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, i.e. each job of a task must be completed before its deadline. Hence, a set \( T \) on \( n \) periodic tasks can be scheduled if and only if

\[
WR_i \leq D_i
\]

for all \( i = 1, \ldots, n \).

For notational convenience, we assume that the tasks are given in order of decreasing priority, i.e. task \( \tau_1 \) has highest priority and task \( \tau_n \) has lowest priority.

2.2 Refined model for FPDS

For FPDS, we need to refine our basic model of Section 2.1. Each job of task \( \tau_i \) is now assumed to consist of \( m_i \) subjobs. The \( j^{th} \) subjob of \( \tau_i \) is characterized by a computation time \( C_{i,j} \in \mathbb{R}^+ \), where \( C_i = \sum_{j=1}^{m_i} C_{i,j} \). We assume that subjobs are non-preemptable. Hence, tasks can only be preempted at subjob boundaries, i.e. at so-called preemption points. For convenience, we will use the term \( F_i \) to denote the computation time \( C_{i,m_i} \) of the final subjob of \( \tau_i \). Note that when \( m_i = 1 \) for all \( i \), we have FPNS as special case.

3 Recapitulation of response time analysis

In this section, we recapitulate worst-case response time analysis and best-case response time analysis for both FPPS and FPDS. Because we will express response times under FPDS in terms of response times under FPPS, we will use subscripts D and P to denote FPDS and FPPS, respectively. Moreover, we will use a functional notation for response times when needed, e.g. \( WR_i(C_i) \).
3.1 Response time analysis for FPPS

3.1.1 Worst-case response time analysis

To determine worst-case response times under arbitrary phasing, it suffices to consider only critical instants. For FPPS, critical instants are given by time points at which all tasks have a simultaneous release [14].

From this notion of critical instants, Joseph and Pandya [10] have derived that for deadlines within periods (i.e. $D_i \leq T_i$) the worst-case response time $WR_i^p$ of a task $\tau_i$ is given by the smallest $x \in \mathbb{R}^+$ that satisfies

$$x = C_i + \sum_{j<i} \left\lfloor \frac{x}{T_j} \right\rfloor C_j. \tag{4}$$

Assuming a critical instant at time zero, the factor $\left\lfloor \frac{x}{T_j} \right\rfloor$ in (4) gives the maximal number of preemptions that an execution of task $\tau_i$ suffers from task $\tau_j$ in an interval $[0,x)$. To calculate worst-case response times, we can use an iterative procedure based on recurrence relationships [1]. The procedure starts with a lower bound.

$$wr_i^{(0)} = C_i$$
$$wr_i^{(k+1)} = C_i + \sum_{j<i} \left\lfloor \frac{wr_i^{(k)}}{T_j} \right\rfloor C_j$$

The procedure is stopped when the same value is found for two successive iterations of $k$ or when the deadline $D_i$ is exceeded. In the former case, it yields the smallest solution of the recursive equation, i.e. the worst-case response time of $\tau_i$. In the latter case the task is not schedulable. Termination of the procedure is ensured by the fact that the sequence $wr_i^{(k)}$ is bounded (from below by $C_i$, and from above by $D_i$) and non-decreasing, and that different values for successive iterations differ at least $\min_{j<i} C_j$.

The interested reader is referred to [11, 17] for techniques to derive worst-case response times for arbitrary deadlines. The main difference with deadlines within periods is that for arbitrary deadlines the worst-case response time of a task is not necessarily assumed for the first job that is released at the critical instant.

3.1.2 Best-case response time analysis

To determine best-case response times under arbitrary phasing, it suffices to consider only so-called optimal (or favourable) instants [3, 15]. For FPPS, an optimal instant for task $\tau_i$ is given by a point in time for which the completion of $\tau_i$ coincides with the simultaneous release of all higher priority tasks.

From this notion of optimal instants, it has been derived that for deadlines within periods the best-case response time $BR_i^p$ of a task $\tau_i$ is given by the largest $x \in \mathbb{R}^+$ that satisfies

$$x = C_i + \sum_{j<i} \left( \left\lfloor \frac{x}{T_j} \right\rfloor - 1 \right) C_j. \tag{5}$$

Assuming an optimal instant at time zero, the factor $\left\lfloor \frac{x}{T_j} \right\rfloor - 1$ in (5) gives the minimal number of preemptions that an execution of task $\tau_i$ suffers from task $\tau_j$ in an interval $(-x,0)$. To calculate best-case response times, we can use the following iterative procedure based on recurrence relationships. The procedure starts with an upper bound. When the worst-case response time $WR_i^p$ of $\tau_i$ is known, we can use it as initial value.

$$br_i^{(0)} = WR_i^p$$
$$br_i^{(k+1)} = C_i + \sum_{j<i} \left( \left\lfloor \frac{br_i^{(k)}}{T_j} \right\rfloor - 1 \right) C_j$$

The procedure is stopped when the same value is found for two successive iterations of $k$, yielding the largest solution of the recursive equation, i.e. the best-case response time of $\tau_i$. Termination of the procedure is ensured by the fact that the sequence $br_i^{(k)}$ is bounded (from below by $C_i$, and from above by $WR_i^p$) and non-increasing, and that different values for successive iterations differ at least $\min_{j<i} C_j$.

We are not aware of a technique to derive best-case response times for arbitrary deadlines. However, Redell and Sanfridson [15] illustrate that the technique given above yields a lower bound for best-case response times of tasks with arbitrary deadlines.

3.2 Response time analysis for FPDS

In this section, we recapitulate response time analysis for FPDS and arbitrary phasing for deadlines within periods. Note that worst-case response time analysis for FPNS and arbitrary deadlines is presented in [8], assuming that all task parameters are taken from $\mathbb{Z}$.

3.2.1 Worst-case response time analysis

In this section, we briefly recapitulate the results from [5, 6, 7]. The non-preemptive nature of subjobs may cause blocking of a task by at most one lower priority task under FPDS. The maximum blocking $B_i$ of task $\tau_i$ by a lower priority task is equal to the longest computation time of any subjob of a task with a priority lower than task $\tau_i$, i.e.

$$B_i = \max_{j<i} \max_{1 \leq k \leq m(j)} C_{j,k}. \tag{6}$$

The worst-case response time $\bar{WR}_i^p$ under FPDS and arbitrary phasing presented in [6] and [7] is given by

$$\bar{WR}_i^p(\Delta) = WR_i^p(B_i + C_i - (F_i - \Delta)) + (F_i - \Delta). \tag{7}$$
According to [7], $\Delta$ is an arbitrary small positive value needed to ensure that the final subjob has actually started, i.e. $0 < \Delta \leq F_i$. Hence, when task $\tau_i$ has consumed $C_i - (F_i - \Delta)$, the final subjob has (just) started.

As described in [5], the analysis in [6, 7] does not take into account that $\tau_i$ can only be blocked by a subjob of a lower priority task if that subjob starts an amount of time $\Delta$ before the simultaneous release of $\tau_i$ and all tasks with a higher priority than $\tau_i$. That paper therefore revisits critical instants, and postulates the following conjecture.

**Conjecture 1** An $\varepsilon$-critical instant of a task $\tau_i$ under FPDS and arbitrary phasing occurs when that task is released simultaneously with all tasks with a higher priority than $\tau_i$, and the subjob with the longest computation time of all lower priority tasks starts an infinitesimal time $\varepsilon > 0$ before that simultaneous release.

From this conjecture, it is concluded that a critical instant for FPDS is a supremum for all but the lowest priority task, i.e. that instant can not be assumed. The results in [6, 7] are identical to the results in [5] for the lowest priority task, and the results become similar for the other tasks by replacing $B_i$ in (7) by $(B_i - \Delta)^+$, i.e.

$$WR_i^D(\Delta) = WR_i^P((B_i - \Delta)^+ + C_i - (F_i - \Delta)) + (F_i - \Delta). \quad (8)$$

Here, the notation $w^+$ stands for $\max(w, 0)$, which is used to indicate that the blocking time can not become negative for the lowest priority task. According to [5], the worst-case response time is actually a supremum for all but the lowest priority task, i.e.

$$WR_i^D = \lim_{\Delta \to 0} WR_i^D(\Delta). \quad (9)$$

### 3.2.2 Best-case response time analysis

Best-case response time analysis has been addressed in [4]. According to that paper, the best-case response time of the highest priority task $\tau_1$ is equal to its computation time, i.e.

$$BR_i^P = C_i. \quad (10)$$

To determine best-case response times under FPDS and arbitrary phasing for a lower priority task $\tau_i$, the paper revisits optimal instants, and postulates the following conjecture.

**Conjecture 2** A $\Delta$-optimal instant of a lower priority task $\tau_i$ (with $1 < i \leq n$) under FPDS and arbitrary phasing occurs when the final sub-job of $\tau_i$ starts a (sufficiently small) finite time $\Delta > 0$ before the simultaneous release of all tasks with a higher priority than $\tau_i$.

Note that a $\Delta$-optimal instant can be assumed, unlike an $\varepsilon$-critical instant, which is a supremum for all but the lowest priority task. Based on this conjecture, the following lower bound is determined for best-case response times of lower priority tasks

$$BR_i^P(\Delta) = BR_i^P(C_i - (F_i - \Delta)) + (F_i - \Delta). \quad (11)$$

### 4 A counterexample

The task characteristics of an example refuting existing worst-case response time analysis of real-time tasks under FPDS and arbitrary phasing is given in Table 1. The table includes the results of the exploration of best-case response times and worst-case response times of the example under FPPS and FPDS. Note that the (processor) utilization factor $U$ of the task set $T_1$ is given by $U = \frac{2}{3} + \frac{1}{\Delta} = 1$.

<table>
<thead>
<tr>
<th>task</th>
<th>$T$</th>
<th>$C$</th>
<th>$WR^P$</th>
<th>$BR^P$</th>
<th>$WR^D$</th>
<th>$BR^D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau_1$</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>$\tau_2$</td>
<td>7</td>
<td>1.2 + 3</td>
<td>8.6</td>
<td>6.6</td>
<td>7</td>
<td>5</td>
</tr>
</tbody>
</table>

**Table 1.** Task characteristics of $T_1$ and worst-case and best-case response times under FPPS and FPDS.

#### 4.1 Existing analysis is too optimistic

In this section, we assume that the deadlines are within periods, i.e. $D_i \leq T_i$. We will now show that the worst-case response time of task $\tau_2$ as determined by (8) is too optimistic.

Based on (8) and using $\Delta = 0.1$, we derive

$$WR_2^P(\Delta) = WR_2^P((B_2 - \Delta)^+ + C_2 - (F_2 - \Delta)) + (F_2 - \Delta)$$

$$= WR_2^P(0 + 4.2 - (3.0 - 0.1)) + (3.0 - 0.1)$$

$$= WR_2^P(1.3) + 2.9 = 6.2.$$

Figure 2 shows a timeline with the executions of the two tasks of $T_1$ under FPDS in an interval of length 35, i.e. equal to the hyperperiod $H$ of the tasks, which is equal to the least common multiple (lcm) of the periods. The schedule in $[0, 35]$ is repeated in the intervals $[hH, (h + 1)H]$ with $h \in \mathbb{Z}$, i.e. the schedule is periodic with period $H$. As illustrated in Figure 2, the derived value for $WR_2^P(\Delta)$ corresponds with the response time of the $1^{st}$ job of task $\tau_2$ upon a simultaneous release with task $\tau_1$, i.e. when task $\tau_2$ is released at an $\varepsilon$-critical instant. However, the response time of the $5^{th}$ job of task $\tau_2$ is equal to 7 in that figure, illustrating that the existing analysis is too optimistic. Nevertheless, the task set is schedulable under FPDS for deadlines equal to periods for this phasing.
4.2 Exploration

We will now consider the example presented in the previous section in more detail, by determining the best-case response times and worst-case response times for both tasks under FPPS and FPDS. To this end, we vary the relative phasing \( \varphi_R \) of task \( \tau_2 \) with respect to \( \tau_1 \), i.e. \( \varphi_R = \varphi_2 - \varphi_1 \). Because the greatest common divisor of \( T_1 \) and \( T_2 \) is equal to 1, we can restrict \( \varphi_R \) to values in the interval \([0,1)\). In this section, we will vary the phasing \( \varphi_2 \) of \( \tau_2 \) and keep the phasing \( \varphi_1 \) of task \( \tau_1 \) equal to zero, i.e. \( \varphi_R = \varphi_2 \).

### 4.2.1 Response times for FPPS

Figure 3 shows a timeline with the executions of the two tasks of \( T_1 \) under FPPS in an interval of a length 35, i.e. equal to the hyperperiod of the tasks. Because both tasks have a simultaneous release at time zero, that time point is a critical instant. Based on [11, 17], we therefore conclude that the job of task \( \tau_2 \) with the largest response time in \([0,35)\) experiences a worst-case response time \( WR_2^p \), i.e. \( WR_2^p = 8.6 \).

Figure 4 shows a timeline for \( T_1 \) with an initial release of \( \tau_1 \) at time zero and an initial release of \( \tau_2 \) at time 0.4. Hence, the relative phasing \( \varphi_R \) of task \( \tau_2 \) with respect to \( \tau_1 \) is a equal to 0.4. For this phasing, task \( \tau_2 \) experiences an optimal instant at time 35, corresponding with the completion of the \( 5^{th} \) job of task \( \tau_2 \). That job is released at time 28.4, and the best-case response time \( BR_2^p \) is therefore equal to \( c_{2.5} - d_{2.5} = 35 - 28.4 = 6.6 \). Similar to the example with arbitrary deadlines shown in [15], the job experiencing the best-case response time cannot immediately start its execution upon its release, but it is delayed by a previous job. In this case, the job is deferred for an amount of time 0.4. As a result, the best-case response time determined by the technique described in Section 3.1.2 yields a lower bound, being 6.2.

The worst-case and best-case response times of task \( \tau_2 \) under FPPS are shown as functions of the relative phasing \( \varphi_R \) in Figure 5.

A remarkable aspect of this example is that for every relative phasing \( \varphi_R \), the end-jitter \( EJ_2^p \) of task \( \tau_2 \) is constant, i.e.

\[
EJ_2^p = \sup_{\varphi_R} (WR_2^p(\varphi_R) - BR_2^p(\varphi_R)) = 1.6.
\]

### 4.2.2 Response times for FPDS

Figure 2 shows a timeline with the executions of the two tasks of \( T_1 \) under FPDS with a simultaneous release at time zero in an interval with a length equal to the hyperperiod of the tasks. Given Figure 2, we observe that for this specific phasing the \( 2^{nd} \) job of task \( \tau_2 \) has the shortest response time, which is equal to 5.4, and the \( 5^{th} \) job of task \( \tau_2 \) has the longest response time, which is equal to 7. Moreover, we observe that for this specific phasing the \( 1^{st} \) and \( 7^{th} \) job of task \( \tau_1 \) both experience a shortest response time of 2, and the \( 3^{rd} \) job of task \( \tau_1 \) has the longest response time, which is equal to 4.4.

Now reconsider Figure 2. We observe that the allocation of the processor to the tasks does not change when the relative phasing \( \varphi_R \) is increased with at most 0.4. All response times of the jobs of task \( \tau_2 \) that are activated in the interval \([0,35)\) decrease linearly with the increase of \( \varphi_R \) from 0 till 0.4. The worst-case response time \( WR_2^D(\varphi_R) \) and the best-case response time \( BR_2^D(\varphi_R) \) of task \( \tau_2 \) therefore also...
decrease. On the other hand, the executions of the jobs of task \( \tau_1 \) are not affected. As a result, \( WR_1^D(\varphi_R) \) and \( BR_1^D(\varphi_R) \) of task \( \tau_1 \) remain the same, i.e., 4.4 and 2, respectively.

For a relative phasing \( \varphi_R = 0.4 \), the 3\(^{rd} \) job of task \( \tau_2 \) can immediately start upon its activation; see Figure 6. When the phasing is increased even further, the 1\(^{st} \) subjob of the 3\(^{rd} \) job of task \( \tau_2 \) will defer the execution of the 4\(^{th} \) job of \( \tau_1 \). Because the utilization of the task set is equal to 1, this implies that all executions of the tasks will be deferred. As a consequence, the response times of all the jobs of \( \tau_2 \) remain the same for \( \varphi_R \in [0.4, 1] \), and the worst-case response time \( WR_2^D(\varphi_R) \) and the best-case response time \( BR_2^D(\varphi_R) \) of task \( \tau_2 \) remain the same. However, the executions of the jobs of task \( \tau_1 \) are affected. As a result, \( WR_1^D(\varphi_R) \) and \( BR_1^D(\varphi_R) \) of task \( \tau_1 \) increase linearly with the increase of \( \varphi_R \).

The worst-case and best-case response times of both task \( \tau_1 \) and task \( \tau_2 \) under FPDS are shown as a function of the phasing in Figure 7. The worst-case response time \( WR_2^D \) of task \( \tau_2 \) is equal to 7.0, and assumed for a relative phasing \( \varphi_R = 0 \), i.e., when task \( \tau_2 \) is released at an \( \varepsilon \)-critical instant. Note that the worst-case response time \( WR_1^D \) of task \( \tau_1 \), given by

\[
WR_1^D = \sup_{\varphi_R} WR_1^D(\varphi_R) = \lim_{\varphi_R \uparrow 1} WR_1^D(\varphi_R) = 5.0.
\]

is a supremum and not a maximum, i.e., that value can not be assumed. We therefore conclude that although the example refutes the worst-case response time analysis, it does not refute Conjecture 1 concerning an \( \varepsilon \)-critical instant.

The best-case response time for \( \tau_2 \) is equal to 5.0. This value is assumed for multiple values of the relative phasing \( \varphi_R \) in the interval \([0, 1]\). As an example, the 2\(^{nd} \) job of task \( \tau_2 \), which is released at time 7.4, experiences a best-case response time. Time point 12.4 is therefore a \( \Delta \)-optimal instant for task \( \tau_2 \). Similar to the best-case situation for FPDS shown in Figure 4, the job experiencing the best-case response time in Figure 6 cannot immediately start executing upon its activation. As a result, the best-case response time analysis presented in [4] indeed yields a lower bound, i.e. based on (11) and using \( \Delta = 0.1 \), we derive

\[
BR_2^D(\Delta) = BR_2^D(C_2 - (F_2 - \Delta)) + (F_2 - \Delta) = BR_2^D(4.2 - (3 - 0.1)) + (3 - 0.1) = BR_2^D(1.3) + 2.9 = 1.3 + 2.9 = 4.2,
\]

which is smaller than 5.0.

A remarkable aspect of this example is that for every relative phasing \( \varphi_R \), the end-jitter of both task \( \tau_1 \) and task \( \tau_2 \) is constant, i.e.

\[
\begin{align*}
EJ_1^D &= \sup_{\varphi_R} (WR_1^D(\varphi_R) - BR_1^D(\varphi_R)) = 2.4 \\
EJ_2^D &= \sup_{\varphi_R} (WR_2^D(\varphi_R) - BR_2^D(\varphi_R)) = 1.6.
\end{align*}
\]

5 Discussion

We have shown that even when deadlines are within periods, we cannot restrict ourselves to the response time of a single job of a task when determining the worst-case response time of that task under FPDS. The reason for this is that the final subjob of a task \( \tau_i \) can defer the execution of higher priority tasks, which can potentially give rise to higher interference for subsequent jobs of task \( \tau_i \).

Considering Figure 2, we see that every job of task \( \tau_2 \) in the interval \([0, 26.8]\) defers the execution of a job of task \( \tau_1 \). Moreover, that deferred job of task \( \tau_1 \) subsequently gives rise to additional interference for the next job of task \( \tau_2 \).
This situation ends when the job of \( \tau_2 \) is started at time \( t = 28 \), i.e. the 5th job of \( \tau_2 \) does not defer the execution of a job of \( \tau_1 \). Viewed in a different way, we may state that the active intervals of the jobs of \( \tau_1 \) and \( \tau_2 \) overlap in the interval \([0, 35]\). Note that this overlapping starts at time \( t = 0 \) and ends at time \( t = 35 \), and we therefore term this interval \([0, 35]\) a level-2 active period. Considering Figure 6, we observe that the overlapping active intervals of the jobs of \( \tau_1 \) and \( \tau_2 \) divide the interval \([0, 35]\) in two level-2 active intervals, i.e. \([0, 14.4]\) and \([14.4, 35]\). Informally, a level-i active period is a smallest interval that only contains entire active intervals of jobs of task \( \tau_i \) and jobs of tasks with a higher priority than task \( \tau_i \). Hence, the active interval of every job of a task \( \tau_i \) is contained in a level-i active period.

To determine the worst-case response time of a task \( \tau_i \), we therefore only have to consider level-i active periods.

However, as illustrated by the example shown in Section 4 and mentioned above, we cannot restrict ourselves to the response time of the first job of a task \( \tau_i \) in a level-i active period when determining the worst-case response time of that task under FPDS. Instead, we have to consider the response times of all jobs in a level-i active period.

We are currently investigating the possibility to determine the worst-case response time of a task \( \tau_i \) under FPDS and arbitrary phasing based on the response times of jobs of \( \tau_i \) in a level-i active period that starts at an \( \varepsilon \)-critical instant.

Note that our notion of level-i active period differs from the notion of level-i busy period [11, 13], which has been introduced to determine worst-case response times of tasks for arbitrary deadlines under FPPS and arbitrary phasing. The level-i busy period is defined as follows.

**Definition 1** A level-i busy period is a time interval \([a, b]\) within which jobs of priority \( i \) or higher are processed throughout \([a, b]\) but no jobs of level \( i \) or higher are processed in \((a - \varepsilon, a)\) or \((b, b + \varepsilon)\) for sufficiently small \( \varepsilon > 0 \).

From this definition, we immediately see that the level-2 busy period never ends for our example because \( U = 1 \). Conversely, the level-2 active period that started at time \( t = 0 \) in Figure 2 ends at time \( t = 35 \). There is another striking difference between level-i active periods and level-i busy periods. A level-i active period may start when a task with a lower priority is still being processed, as illustrated by the level-1 active period that starts at time \( t = 5 \) in Figure 2. The corresponding level-1 busy period does not start at time \( t = 5 \), but at time \( t = 6.2 \) instead.

6 Conclusion

In this document, we considered response times of real-time tasks under FPDS and arbitrary phasing. We showed by means of an example consisting of just two tasks that existing worst-case response time analysis for deadlines within periods as presented in [5, 6, 7] is too optimistic.

We explored the example by considering both best-case and worst-case response times under both FPPS and FPDS as a function of the relative phasing between the tasks. From this exploration, we gained the following results. We found that, although the example refutes the existing worst-case
response time analysis, it does not refute Conjecture 1 concerning the notion of \( \varepsilon \)-critical instant. The example merely reveals that the worst-case response time of a task scheduled under FPDS is not necessarily assumed for the first job of that task when released at an \( \varepsilon \)-critical instant. This is a similar result as presented in [13] for critical instants of tasks under FPPS with arbitrary phasing and deadlines greater than periods. Our example also revealed that a job that experiences a \( \Delta \)-optimal instant may not be able to immediately start executing upon its activation. As a consequence, the best-case response time analysis under FPDS and arbitrary phasing as presented in [4] indeed yields a lower bound. This is a similar result as presented in [15] for FPPS with arbitrary phasing and deadlines greater than periods.

Worst-case response time analysis of a task \( \tau_i \) under FPDS and arbitrary phasing is a topic of future work. We are currently investigating the possibility to determine the worst-case response time of a task \( \tau_i \) under FPDS and arbitrary phasing based on the response times of jobs of \( \tau_i \) in a so-called level-i active period that starts at an \( \varepsilon \)-critical instant. Initial results suggest that the technique is similar to existing techniques for FPPS with arbitrary phasing and arbitrary deadlines [11, 17].

Acknowledgement

First of all, I thank my son Wander for his inspiration; the problem with the existing analysis occurred to me while watching him play in a sandpit. Next, I thank Wim F.J. Verhaegh from Philips Research and Johan J. Lukkien from the Technische Universiteit Eindhoven (TU/e) for discussions. Finally, I thank Pieter J.L. Cuijpers and Rudolf H. Mak from the TU/e for their comments on earlier versions of this document.

References


A Timelines for \( \mathcal{T}_1 \)

Timelines for \( \mathcal{T}_1 \) under FPPS and FPDS for a relative phasing \( \Phi_R \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\} \) are shown in Figures 8 and 9, respectively.
Figure 8. Timelines for $T_1$ under FPPS for a relative phasing $\phi_R \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. 
Figure 9. Timelines for $T_1$ under FPDS for a relative phasing $\phi_R \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. 