Tighter schedulability analysis of synchronization protocols based on overrun without payback for hierarchical scheduling frameworks
Behnam, M.; Nolte, T.; Bril, R.J.

Published in:

DOI:
10.1109/ICECCS.2011.11

Published: 01/01/2011

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 author's 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

Citation for published version (APA):

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

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
Tighter schedulability analysis of synchronization protocols based on overrun without payback for hierarchical scheduling frameworks

Moris Behnam, Thomas Nolte
Mälardalen Real-Time Research Centre
P.O. Box 883, SE-721 23 Västerås, Sweden
moris.behnam@mdh.se

Reinder J. Bril
Technische Universiteit Eindhoven (TU/e)
Den Dolech 2, 5612 AZ Eindhoven
The Netherlands

Abstract—In this paper, we show that both global as well as local schedulability analysis of synchronization protocols based on the stack resource policy (SRP) and overrun without payback for hierarchical scheduling frameworks based on fixed-priority preemptive scheduling (FPPS) are pessimistic. We present tighter global and local schedulability analysis, illustrate the improvements of the new analysis by means of examples, and show that the improved global analysis is both uniform and sustainable. We evaluate the new global and local schedulability analysis based on an extensive simulation study and compare the results with the existing analysis.

I. INTRODUCTION

Background: Over the years, there has been a growing attention for hierarchical scheduling of real-time systems due to its ability to provide temporal isolation between multiple real-time subsystems executing upon a common processing platform. The Hierarchical Scheduling Framework (HSF) provides means for decomposing a complex system into well-defined parts called subsystems, and a subsystem provides an introspective interface that specifies the timing properties of the subsystem precisely. This implies that subsystems can be independently developed, analyzed and tested, and later assembled without introducing unwanted temporal interference.

Supporting global resource sharing between subsystems is a major challenge, since it increases the complexity of the analysis of a system considerably. Due to this complexity, most of the proposed techniques are based on some simplifying assumptions which make the analysis easier, e.g., [1], [2]. The consequence of these assumptions is that they add pessimism in the analysis which increases the required CPU resources of systems. For some systems, the pessimism in the analysis is not significant and can be ignored, but for others it may be significant.

As large extents of embedded systems are resource constrained, a tight analysis is instrumental in a successful deployment of HSF techniques in real applications. We therefore aim at reducing potential pessimism in existing schedulability analysis for HSFs that support sharing of global shared resources. Looking further at existing industrial real-time systems, fixed priority preemptive scheduling (FPPS) is the de facto standard of task scheduling, hence we focus on an HSF with support for FPPS for tasks within a subsystem. Having such support will simplify migration to and integration of existing legacy applications into the HSF, avoiding a too big technology revolution for engineers.

Our current research efforts are directed towards the conception and realization of a two-level HSF that is based on (i) FPPS for both global scheduling of budgets (allocated to subsystems) and local scheduling of tasks (within a subsystem), (ii) the periodic resource model [3] for budgets, and (iii) the Stack Resource Policy (SRP) [4] for both inter- and intra-subsystem resource sharing. For such an HSF, two mechanisms have been studied that prevent depletion of a budget during global resource access, i.e. skipping [1] and overrun [2]. Note that, budget depletion during global resource access may cause tasks from other subsystems missing their deadline. The overrun mechanism comes in two flavors, i.e. with payback and without payback.

In this paper, we aim at tighter analysis for the overrun mechanism without payback, assuming the same introspective interface for subsystems as the existing analysis.

Contributions: We show that existing global and local schedulability analysis of synchronization protocols based on SRP and overrun without payback for two-level hierarchical scheduling based on FPPS is pessimistic. We present tighter global and local analysis assuming that the deadline of a subsystem holds for the sum of its normal budget and its overrun budget, and illustrate the improvements by means of examples. We identify the system parameters that have a great effect on the improvement of the proposed global and local analysis. In addition we evaluate the improvements that both global and local new analysis can achieve compared with the traditional analysis, in terms of CPU resources, by exploring the system load [2] in a simulation study.

Overview: This paper has the following structure. In Section II we present related work. A real-time scheduling model is the topic of Section III. The existing global and local schedulability analysis is recapitulated in Section IV, and tighter global and local analysis is presented in Sections V and VI, respectively. Section VII presents a simulation study evaluating the improvement that both global and local new analysis can achieve. The paper is concluded in Section VIII.
II. RELATED WORK

During the past decade, there has been considerable interest on hierarchical scheduling of real-time systems [5], [6], [7], [3]. Deng and Liu [5] proposed a two-level HSF for open systems, where subsystems may be developed and validated independently. Kuo and Li [6] and Lipari and Baruah [7] presented schedulability analysis techniques for such two-level frameworks with FPPS and Earliest Deadline First (EDF) global schedulers, respectively. Shin and Lee [3] proposed the periodic resource model $\Gamma(\Pi, \Theta)$ to specify guaranteed periodic CPU allocations, where $\Pi \in \mathbb{R}^+$ is a period and $\Theta \in \mathbb{R}^+$ is a periodic allocation time ($0 < \Theta \leq \Pi$). Easwaran, and Lee [8] proposed the explicit deadline periodic (EDP) resource model $\Omega(\Pi, \Theta, \Delta)$ that extends the periodic resource model by explicitly distinguishing a relative deadline $\Delta \in \mathbb{R}^+$ for the allocation time $\Theta$ ($0 < \Theta \leq \Delta \leq \Pi$).

For synchronization protocols in HSFs, two mechanisms have been studied to prevent depletion of a budget during global resource access, i.e. *skipping* and *overrun* (with payback and without payback). The idea of skipping in the context of HSFs, was used by the SIRAP protocol [1], and its associated analysis supports composability. It works as follows: when a job tries to access a global shared resource, it will be granted the access to the resource if the remaining subsystem budget is enough to lock and release the global resource before budget depletion. Otherwise, the access to the shared resource will be delayed until the next activation period. Overrun with payback was first introduced in the context of aperiodic servers in [9]. The mechanism was later (re-) used for a synchronization protocol in the context of two-level hierarchical scheduling in [10] and extended with overrun without payback. Overrun mechanism works as follows: when the budget of a subsystem depletes and an internal job has not released the lock of a global shared resource, the subsystem overruns its budget and the job continues its execution until it releases the locked resource. This mechanism is called overrun with payback if the subsystem budget is decreased by the amount of the overrun in the next activation period followed by an overrun, otherwise it is called overrun without payback. The analysis presented in [10] does not support independent subsystems development, i.e., the parameters of the other subsystems should be available in order to perform the analysis of each subsystem. However an analysis supporting composability was described in [2], [11].

In this paper we present tighter global and local analysis for overrun without payback and we evaluate, by means of simulation study, the improvements that the new tighter analysis can achieve compared with the traditional analysis, in terms of CPU resources.

III. REAL-TIME SCHEDULING MODEL

We consider a two-level hierarchical FPPS model using the periodic resource model to specify guaranteed CPU allocations to tasks of subsystems and using a synchronization protocol for mutual exclusive resource access to global resources based on SRP \(^1\) and overrun without payback.

System model: A system $\mathcal{SYS}$ contains a set $\mathcal{R}$ of $M$ global logical resources $R_1$, $R_2$, $\ldots$, $R_M$, a set $\mathcal{S}$ of $N$ subsystems $S_1$, $S_2$, $\ldots$, $S_N$, a set $\mathcal{B}$ of $N$ budgets for which we assume a periodic resource model [3], and a single processor. Each subsystem $S_i$ has a dedicated budget associated to it. In the remainder of this paper, we leave budgets implicit, i.e. the timing characteristics of budgets are taken care of in the description of subsystems. Subsystems are scheduled by means of FPPS and have fixed, unique priorities. For notational convenience, we assume that subsystems are given in order of decreasing priorities, i.e. $S_1$ has the highest priority and $S_N$ has the lowest priority.

Subsystem model: Each subsystem $S_i$ contains a set $T_s$ of $n_s$ periodic tasks $\tau_{s1}$, $\tau_{s2}$, $\ldots$, $\tau_{sn_s}$, with fixed, unique priorities, which are scheduled by means of FPPS. For notational convenience, we assume that tasks are given in order of decreasing priorities, i.e. $\tau_1$ has highest priority and $\tau_{n_s}$ has lowest priority. The set $\mathcal{R}_s$ denotes the subset of $M_s$ global resources accessed by subsystem $S_i$. The maximum time that a subsystem $S_i$ executes while accessing resource $R_l \in \mathcal{R}$ is denoted by $X_{sil}$, where $X_{sil} \in \mathbb{R}^+ \cup \{0\}$ and $X_{sil} > 0 \iff R_l \in \mathcal{R}_s$. The timing characteristics of $S_i$ are specified by means of a triple $(P_s, Q_s, X_s)$, where $P_s \in \mathbb{R}^+$ denotes its (budget) period, $Q_s \in \mathbb{R}^+$ its (normal) budget, and $X_s$ the set of maximum execution access times of $S_i$ to global resources. The maximum value in $X_s$ (or zero when $X_s$ is empty) is denoted by $X_s$.

Task model: The timing characteristics of a task $\tau_{s1} \in T_s$ are specified by means of a quartet $(T_{sil}, C_{si}, D_{si}, C_{si})$, where $T_{sil} \in \mathbb{R}^+$ denotes its minimum inter-arrival time, $C_{si} \in \mathbb{R}^+$ its worst-case computation time, $D_{si} \in \mathbb{R}^+$ its (relative) deadline, $C_{si}$ a set of maximum execution times of $\tau_{s1}$ to global resources, where $C_{si} \leq D_{si} \leq T_{sil}$ and $P_s \leq T_{sil}$ [2]. The set $\mathcal{R}_s$ denotes the subset of $\mathcal{R}$ accessed by task $\tau_{s1}$. The maximum time that a task $\tau_{s1}$ executes while accessing resource $R_l \in \mathcal{R}$ is denoted by $c_{sil}$, where $c_{sil} \in \mathbb{R}^+ \cup \{0\}$, $C_{si} \geq c_{sil}$, and $c_{sil} > 0 \iff R_l \in \mathcal{R}_s$.

Resource model: The CPU supply refers to the amount of CPU allocation that a virtual processor can provide. The supply bound function $sbf_{\Omega}(t)$ of the EDP resource model $\Omega(\Pi, \Theta, \Delta)$ that computes the minimum possible CPU supply for every interval length $t$ is given by [3]

$$sbf_{\Omega}(t) = \begin{cases} t - (k + 1)(\Pi - \Theta) + (\Pi - \Delta) & \text{if } t \in V^{(k)} \\ (k - 1)\Theta & \text{otherwise,} \end{cases}$$

\(^1\)The focus of this paper is on synchronization protocols for global logical resources. We do therefore not consider local logical resources.
where \( k = \max \left( \left( t - (\Delta - \Theta) \right) / \Pi \right), 1 \) and \( V^{(k)} \) denotes an interval \([k\Pi + \Delta - 2\Theta, k\Pi + \Delta - \Theta]\).

The supply bound function \( \text{sb}_{\Gamma}(t) \) of the periodic resource model \( \Gamma(\Pi, \Theta) \) is a special case of (1), i.e. with \( \Delta = \Pi \).

**Synchronization protocol:** Overrun without payback prevents depletion of a budget of a subsystem \( S_i \) during access to a global resource \( R_l \) by temporarily increasing the budget of \( S_i \) with \( X_{sl} \), the maximum time that \( S_i \) executes while accessing \( R_l \). To be able to use SRP in an HSF for synchronizing global resources, its associated ceiling terms needs to be extended.

**Resource ceiling:** With every global resource \( R_l \), two types of resource ceilings are associated; an external resource ceiling \( RC_l \) for global scheduling and an internal resource ceiling \( rc_{sl} \) for local scheduling. According to SRP, these ceilings are defined as

\[
    RC_l = \min(N, \min\{s \mid X_{sl} > 0\}), \quad (2)
\]

\[
    rc_{sl} = \min(n_s, \min\{i \mid c_{si} > 0\}). \quad (3)
\]

Note that we use the outermost \( \min \) in (2) and (3) to define \( RC_l \) and \( rc_{sl} \) also in those situations where no subsystem uses \( R_l \) and no task of \( T_x \) uses \( R_l \), respectively.

**System/subsystem ceiling:** The system/subsystem ceilings are dynamic parameters that change during the execution. The system/subsystem ceiling is equal to the lowest external/internal resource ceiling of a currently locked resource in the system/subsystem.

Under SRP, a task \( \tau_{si} \) can only preempt the currently executing task \( \tau_{sj} \) (even when accessing a global resource) if the priority of \( \tau_{si} \) is greater (i.e. the index \( i \) is lower) than \( S_i \)’s subsystem ceiling. A similar condition for preemption holds for subsystems.

**Concluding remarks:** The maximum time \( X_{sl} \) that \( S_i \) executes while accessing \( R_l \) can be reduced by assigning a value to \( rc_{sl} \) that is smaller than the value according to SRP. For HSRP [10], the internal resource ceiling is therefore set to the highest priority, i.e. \( rc_{sl}^{\text{HSRP}} = 1 \). Decreasing \( rc_{sl} \) may cause a subsystem to become unfeasible for a given budget [12], however, because the tasks with a priority higher than the old ceiling and at most equal to the new ceiling may no longer be feasible.

The results in this paper apply for any internal resource ceiling \( rc_{sl}^{P} \) where \( rc_{sl} \geq rc_{sl}^{P} \geq rc_{sl}^{\text{HSRP}} = 1.2 \)

**IV. Recap of Existing Schedulability Analysis**

In this section, we briefly recapitulate the global schedulability analysis presented in [10] and the local schedulability analysis described in [2], [11]. Although the global schedulability analysis presented in [2], [11] looks different, it is based on the analysis described in [10] and therefore yields the same result.

For illustration purposes, we will use an example system \( Sys_1 \) containing two subsystems \( S_1 \) and \( S_2 \) sharing a global resource \( R_1 \). The characteristics of the subsystems are given in Table I.

<table>
<thead>
<tr>
<th>Subsystem</th>
<th>( P_s )</th>
<th>( Q_s + X_s )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_1 )</td>
<td>5</td>
<td>( 2 )</td>
</tr>
<tr>
<td>( S_2 )</td>
<td>7</td>
<td>( 2 + X_2 )</td>
</tr>
</tbody>
</table>

**Table I:** Subsystem characteristics of \( Sys_1 \).

**A. Global analysis**

The worst-case response time \( WR_s \) of subsystem \( S_i \) is given by the smallest \( x \in \mathbb{R}^+ \) satisfying

\[
    x = B_s + (Q_s + X_s) + \sum_{t<s} \left[ \frac{x}{\bar{F}_t} \right] (Q_t + X_t), \quad (4)
\]

where \( B_s \) is the maximum blocking time of \( S_i \) by lower priority subsystems, i.e.

\[
    B_s = \max(0, \max\{X_{tl} \mid t > s \wedge X_{tl} > 0 \wedge RC_l \leq s\}). \quad (5)
\]

Note that we use the outermost \( \max \) in (5) to define \( B_s \) in those situations where the set of values of the innermost \( \max \) is empty. To calculate \( WR_s \), we can use an iterative procedure based on recurrence relationships, starting with a lower bound, e.g. \( B_s + \sum_{t<s}(Q_s + X_t) \). The condition for global schedulability is given by

\[
    \forall 1 \leq s \leq N \quad WR_s \leq P_s. \quad (6)
\]

We observe that the global analysis is similar to basic analysis for FPPS with resource sharing, where the period \( P_s \) of a subsystem \( S_i \) serves as deadline for the sum of the normal budget \( Q_s \) and the overrun budget \( X_s \). Hence the interference of higher priority subsystems \( S_i \) is based on the sum \( Q_t + X_t \). We will therefore use a superscript \( P \) to refer to this basic analysis for subsystems, e.g. \( WR_s^P \).

In the sequel, we are not only interested in the worst-case response time of a subsystem \( S_i \) for particular values of \( B_s, Q_s, \) and \( X_s \), but in the value as a function of the sum of these three values. We will therefore use a functional notation when needed, e.g. \( WR_s(B_s + Q_s + X_s) \).

The global feasibility area of the existing analysis is illustrated for our example system \( Sys_1 \) in Figure 1(a). Note that the y-axis is excluded, because we assume the capacity of subsystems to be positive, i.e. \( Q_s > 0 \).

\( ^3 \)Strictly spoken, [10] uses (4) excluding \( X_s \) for \( WR_s \). The smallest positive solution of (4) is required to be at most equal to \( P_s \) to prevent additional interference of the next activation of (the budget of) \( S_i \).
B. Local analysis

The existing condition for local schedulability of a subsystem $S_i$ [2] is given by

$$\forall 1 \leq i \leq n_s, \exists x \in D_{si}, b_{si} + C_{si} + \sum_{j < i} \left[ \frac{x}{T_{sj}} \right] \cdot C_{sj} \leq \text{bfbf}_{\Gamma_s}(x),$$

where $b_{si}$ is the maximum blocking time of $\tau_{si}$ by lower priority tasks, i.e.

$$b_{si} = \max(0, \max\{c_{sji} \mid j > i \land c_{sji} > 0 \land r_{ci} \leq i\}),$$

and $\text{bfbf}_{\Gamma_s}(x)$ is the supply bound function of the periodic resource model $\Gamma_s(P_s, Q_s)$ for the subsystem $S_i$ under consideration. Note that we use the outermost $\max$ in (8) to define $b_{si}$ also in those situations where the set of values of the innermost $\max$ is empty.

The value for $X_{si}$ depends on the local scheduler and the synchronization protocol. The maximum time that subsystem $S_i$ executes while task $\tau_{si}$ accesses resource $R_i \in \mathcal{R}$ is denoted by $X_{si}$, where $X_{si} \in \mathbb{R}^+ \cup \{0\}$ and $X_{si} > 0 \iff c_{si} > 0$. For $c_{si} > 0$, $X_{si}$ is given by [2]

$$X_{si} = c_{si} + \sum_{j < r_{ci}} C_{sj}.$$  

The value for $X_{si}$ is given by

$$X_{si} = \max_{1 \leq i \leq n_s} X_{si}.$$  

V. Tighter GLOBAL ANALYSIS

As described in Section IV-A, the existing global schedulability analysis is based on FPPS, where the period $P_s$ serves as deadline for the sum of the normal budget $Q_s$ and overrun budget $X_s$.

A. Illustrating the improvement

The improvement of the global analysis is based on two observations:

1) Limited pre-emption of overrun budget $X_s$: while $S_i$ is accessing $R_i$ using $X_s$, it can only be pre-empted by subsystems with a priority higher than $RC_i$.

2) Blocking starts before the execution based on the overrun budget $X_s$ starts: to use its overrun budget $X_s$, $S_i$ needs to first lock a global resource.

From the first observation, we conclude that subsystem $S_1$ cannot pre-empt $S_2$ during those intervals of time when $S_2$ is accessing resource $R_1$ in general, and when $S_2$ is executing based on its overrun budget $X_2$ in particular. This limited preempt-ability of subsystem $S_2$ gives rise to improved schedulability of $S_2$.

From the second observation, we conclude that whenever $S_2$ uses its overrun budget $X_2$, it must have locked $R_1$ already during the consumption of its normal budget $Q_2$, i.e. before it starts consuming its overrun budget $X_2$. Hence, the system ceiling is already set to the priority of $S_1$ before $S_2$ starts consuming $X_2$, preventing $S_1$ to pre-empt.

The resulting improvements is illustrated in Figure 1(b), which we briefly explain by means of an example. Figure 2 shows a timeline with $Q_2 = 3.0$ and $X_2 = 1.0$, where the first job $\ell_{2,0}$ of $S_2$ locks $R_1$ just before the activation of $S_1$ at $t = 5$. Subsystem $S_2$ is therefore allowed to execute $X_2$ at $t = 5$, effectively deferring the execution of $S_1$. This has a number of consequences. Firstly, $S_2$ does not miss its deadline at time $t = 7$, as we would conclude from the existing analysis. Secondly, the worst-case response time of $S_2$ is no longer assumed for $\ell_{2,0}$, activated at $t = 0$, but instead for $\ell_{2,1}$ activated at $t = 7$, because the deferred execution of $S_1$ gives rise to additional interference for $\ell_{2,1}$. Rather than having to consider only a single job to determine schedulability, we therefore have to consider all jobs in a so-called level-$s$ active period, similar to the analysis for FPDS [14] and FPPS with preemption thresholds [15]. The level-2 active period starts at time $t = 0$, when both $S_1$ and $S_2$ become active, ends at time $t = 14$, when all pending work of $S_1$ and $S_2$ has been completed, and contains two jobs of $S_2$. Because both jobs meet their deadline, $S_2$ is schedulable.

Figure 2. Timeline for $Q_2 = 3.0$ and $X_2 = 1.0$ assuming blocking starts before overrun.

B. Improving the global analysis

In this section, we first recapitulate the notion of a level-$s$ active period. Next, we derive analysis for the worst-case finalization time $WF_{sk}^{Q_s}$ of the normal budget $Q_s$ of job $t_{sk}$ of subsystem $S_s$ relative to start of the constituting level-$s$ active period. Finally, we derive analysis for the worst-case response time $WR_{sk}$ of $S_s$.

The interested reader is referred to [13] for a detailed explanation.
1) Level-s active period: The worst-case length $WL_s$ of a level-s active period with $s \leq N$ is given by the smallest $x \in \mathbb{R}^+$ that satisfies

$$x = B_s + \sum_{t \leq s} \left\lceil \frac{x}{P_t} \right\rceil (Q_t + X_t).$$

To calculate $WL_s$, we can use an iterative procedure based on recurrence relationships, starting with a lower bound, e.g. $B_s + \sum_{t \leq s} (Q_t + X_t)$. The maximum number $w_l_s$ of jobs of subsystem $S_s$ in a level-s active period is given by

$$w_l_s = \left\lceil \frac{WL_s}{P_s} \right\rceil.$$  

2) Worst-case finalization time: For a job $\tau_{sk}$ of subsystem $S_s$ with $0 \leq k < w_l_s$, we split the interval from the start of the level-s active period to the finalization of job $\tau_{sk}$ in two sub-intervals: a first sub-interval including the execution of the normal budget $Q_s$ by job $\tau_{sk}$ and a second sub-interval from the finalization of $Q_s$ by $\tau_{sk}$ till the finalization of the level-s active period. The worst-case finalization time $WF_{sk}^Q$ of the subsystem $S_s$ is defined by

$$WF_{sk}^Q = \sum_{s \geq t \geq RC_s} \left\lceil \frac{WF_{sk}^Q}{P_t} \right\rceil (Q_t + X_t).$$

3) Worst-case response time: Let job $\tau_{sk}$ of $S_s$ access $R_l \in \mathcal{R}$. When $\tau_{sk}$ starts to consume its overrun budget, the subsystems $S_{s-1}$ till $S_{RC_s}$ are already blocked, and only subsystems with a priority higher than $RC_s$ can therefore still pre-empt $S_s$. To determine the worst-case response time $WR_{sk}$ of job $\tau_{sk}$ of $S_s$, we bound the worst-case response time of the fictive subsystem $S_{RC_s}$, i.e. a subsystem that can only be preempted by tasks with a higher priority than $RC_s$. The preemptions during $WF_{sk}^Q$ by subsystems $S_{s-1}$ till $S_{RC_s}$ are treated as additional blocking of $S_{RC_s}$. The worst-case interference of the subsystems $S_{s-1}$ till $S_{RC_s}$ in the interval of length $WF_{sk}^Q$ is denoted by $WI_{RC_s,k}$. Given by

$$WF_{sk}^Q = \sum_{s \geq t \geq RC_s} \left\lceil \frac{WF_{sk}^Q}{P_t} \right\rceil (Q_t + X_t).$$

The worst-case response time $WR_{sk}$ of job $\tau_{sk}$ of subsystem $S_s$ when it accesses $R_l$ is now given by

$$WR_{sk} = \lim_{x \uparrow C_s} WF_{sk}^P(B + k(Q_s + X) + Q),$$

where $WF_{sk}^P$ is the worst-case response time of a fictive subsystem $S'_{sk}$ with a period $P' = (k+1)T_s$, a normal budget $Q' = k(Q_s + X) + Q$, and a maximum blocking time $B$. Using the following equation from [14]

$$\lim_{x \uparrow C_s} WF_{sk}^P(x) = WF_{sk}^P(C_s)$$

we derive

$$WF_{sk}^Q = WF_{sk}^P(B_s + (k + 1)Q_s + kX_s).$$
at every activation; see Figure 3. Note that subsystem $S_3$ will miss its deadline at time $t = 7$ for an infinitesimal increase $\epsilon > 0$ of $X_{3,2}$. The worst-case response time for job $t_{sk}$ is therefore the maximum for all global resources accessed by $S_s$, i.e.

$$WR_{sk} = \max_i WR_{skl}.$$  (18)

Finally, the worst-case response time $WR_s$ of subsystem $S_s$ is given by $^5$

$$WR_s = \max_{0 \leq k < wls} WR_{sk}.$$  (19)

C. Concluding remarks

In this section, we briefly discuss three aspects of the global analysis, i.e. the global analysis is (i) uniform and (ii) sustainable and (iii) will never give worse results than the original analysis. We conclude this section with a remark on the complexity of the analysis.

The analysis for FPDS [14] is not uniform for all tasks, i.e. the analysis for the lowest priority task differs from the analysis of the other tasks. This anomaly is caused by the fact that the lowest priority task cannot be blocked, i.e. its blocking time is zero, and the blocking time of all other tasks is a supremum rather than a maximum. Unlike the analysis for FPDS [14], the global analysis presented in this section is uniform. This is an immediate consequence of the fact that blocking of a global resource $R_i$ by a subsystem $S_s$ is already done during the execution of the normal budget, i.e. before the execution based on the overrun budget starts. As a result, subsystems $S_{s-1}$ till $S_{RC_1}$ cannot preempt $S_s$ at the finalization time of $Q_s$.

As described in [16], a schedulability test is sustainable if any task system deemed schedulable by the test remains so if it behaves ‘better’ than mandated by its system specifications, i.e. sustainability requires that schedulability be preserved in situations in which it should be ‘easier’ to ensure schedulability. Given our scheduling model, we use the following definition for sustainability of our tighter global schedulability test.

**Definition 1:** A schedulability test for our real-time scheduling model for subsystems is sustainable if any system deemed schedulable by the schedulability test remains schedulable when the parameters of one or more individual job[s] are changed in any, some, or all of the following ways: (i) decreased normal budgets; (ii) decreased overrun budgets, (iii) later arrival times; and (iv) larger relative deadlines.

With this definition, sustainability of our global schedulability test immediately follows from (6), i.e. $WR_s \leq P_s = D_s$ and the fact that

- the maximum number $wls$ of jobs of subsystem $S_s$ in a level-$s$ active period, and
- the worst-case finalization time $WF^Q_{sk}$ in (14), the worst-case interference $WF_{RC_1,k}$ in (15), and the worst-case response time $WR_{skl}$ in (16)

are strictly non-increasing for decreasing normal budgets, decreasing overrun budgets, and increasing budget periods of subsystems.

We will prove that the tighter global analysis will never give worse results than the original analysis i.e., it will give better results or in the worst case the same results as the original analysis. Looking at (12), if $wls > 1$, then the system will be unschedulable using the original analysis because the first job will miss its deadline according to the original analysis. While using the modified analysis, the same systems can be schedulable. If $wls = 1$ then $k = 0$ and $X_s$ will not have any effect in (14) since $k = 0$. For modified analysis, only the subsystems with a higher priority than the resource ceiling of the resource being locked are able to preempt. Taking this into account can reduce the amount of interference considered due to higher priority subsystems in general and for $k = 0$ in particular. Which in turn can improve the results in terms of response time, schedulability and the CPU-resource requirement.

Finally, comparing the new global analysis (11-19) with the existing analysis (4) it is clear that the new analysis is more complex. For static systems, for which the set $S$ of subsystems does not change during runtime, this additional complexity will not be very important since the analysis will be performed during the integration phase of the system, i.e. off-line. For dynamic systems, for which subsystems can be added or removed during runtime, the new analysis can be used when the original analysis fails to find a feasible solution.

VI. TIGHTER LOCAL ANALYSIS

Both the existing global schedulability analysis and the new global schedulability analysis assume a deadline for a subsystem $S_s$ equal to its period $P_s$ for the sum of the normal budget $Q_s$ and the overrun budget $X_s$. The existing local schedulability analysis for the tasks of $S_s$ is exclusively based on $Q_s$, however. Hence, when a system is feasible from a global scheduling perspective, the latest finalization time of $Q_s$ is guaranteed to be at least $X_s$ before the next activation of $S_s$. Hence, we can use the supply bound function $sb(t)$ of the EDP resource model $\Omega_t(P_s, Q_s, \Delta_s)$ for overrun without payback rather than $sb(t)$ of $\Gamma_s(P_s, Q_s)$

Figure 3. Subsystem $S_3$ just meets it deadline at $t = 7$ for $X_{3,2} = 0.4$.

$^5$The interested reader is referred to [13], which explains the improvement in detail by means of a variety of timelines.
in (7), where $\Delta_s = P_s - X_s$. Because $X_s \geq 0$ for all subsystems (by definition), $\text{sbf}_{\Gamma_s}(t) \leq \text{sbf}_{\Omega_s}(t)$ for all subsystems. As a result, a subsystem may be schedulable according to the local analysis based on $\text{sbf}_{\Gamma_s}(t)$, but not be schedulable based on $\text{sbf}_{\Gamma_s}(t)$.

Figure 4 shows an example of the supply bound functions $\text{sbf}_{\Omega_s}(t)$ and $\text{sbf}_{\Gamma_s}(t)$ for subsystem $S_2$ of system $S_{YS}$ where $Q_2 = 1.8$ and $X_2 = 2.4$.

![Supply bound functions](image)

Figure 4. Supply bound functions $\text{sbf}_{\Omega_s}(t)$ and $\text{sbf}_{\Gamma_s}(t)$ for $S_2$ with $Q_2 = 1.8$ and $X_2 = 2.4$.

VII. Evaluation

In this section, we evaluate the modified overrun without payback analysis (MONP), including both local and global new tighter analyses, with respect to CPU resource. We compare MONP with the traditional overrun without payback mechanism (ONP) using the notion of system load [2], as system load provides an indication of the system CPU requirement in the presence of shared resources. The comparison is carried out by means of simulation experiments. To show the performance of MONP relative to alternative approaches.

We start this section by briefly explaining the notion of system load and how it should be adapted for MONP.

**System load:** System load is defined as a quantitative measure to represent the minimum amount of CPU allocations necessary to guarantee the global schedulability of the system $S$.

For ONP, system load $\text{load}_{\text{sys}}$ is calculated as follows:

$$\text{load}_{\text{sys}} = \max_{S \in S} \{ \alpha_s \}$$

where

$$\alpha_s = \min_{0 < x \leq P_s} \left\{ \frac{\text{RBF}_s(x)}{x} \mid \text{RBF}_s(x) \leq x \right\}$$

and

$$\text{RBF}_s(x) = B_s + (Q_s + X_s) + \sum_{t < s} \left[ \frac{x}{P_t} \right] (Q_t + X_t).$$

Note that $x$ can be selected within a finite set of scheduling points [17], and that $\alpha_s$ is the smallest fraction of the CPU resource required to schedule a subsystem $S_s$ (satisfying the schedulability condition presented in Section IV-A), assuming that the global resource supply function is $\alpha_s x$.

One can think of system load as decreasing the speed of the processor by the factor $\text{load}_{\text{sys}}$, which will increase the subsystems’ normal budgets, the overrun budgets, and blocking times by a factor $1/\text{load}_{\text{sys}}$.

For MONP, evaluating system load is more complex than e.g., for ONP, because it has more than one response time equation for global schedulability analysis (see Section V), unlike the case for ONP which has only one equation. To perform the schedulability analysis for MONP, firstly, the value of $w_s$ should be evaluated in order to evaluate the range of $k$ that is used by the other equations. However, we cannot evaluate the value of $w_s$ in (12) without having the value of system load known. Without having the range of $k$, we can not use the equations (14) - (19) that are required in the calculation of system load. We solve this problem by using a binary search algorithm, such that the system load is selected by the search algorithm and corresponding system schedulability is checked. To do this we multiply all subsystems normal budgets, maximum overrun budgets, and blocking times in equations (12) - (19) by a factor $1/\text{load}_{\text{sys}}$. If the system is schedulable, then the algorithm will select a lower system load and try again. If the system is unschedulable, then the algorithm will select a higher system load. The algorithm terminates if the selected system load $\text{load}_{\text{sys}} > 1$ and the system is unschedulable, or when the difference between the previous and the current system load is less than a given acceptance limit. Since we have used a binary search algorithm for MONP, the complexity of evaluating the system load is higher compared with ONP. However, note that we use the system load for comparison purposes only, hence it does not have any relationship with the complexity of the schedulability analysis.

The efficiency of MONP is measured by the amount of system load required for schedulability, relative to ONP.

Both the new tighter local and global analysis in MONP can decrease the system load. For the tighter local analysis, it has the potential to decrease the subsystem normal budget for certain subsystems, which in turn, can decrease the system load, since it decreases the effect of the interference from higher priority subsystems and the required normal budget of the subsystem itself, in equations (12) - (19). However, there is no guarantee that the improved local analysis can decrease the subsystem’s normal budget. Note that $\text{sbf}_{\Gamma_s}(x) < \text{sbf}_{\Omega_s}(x)$ for $A P_s - 2Q_s - X_s < x < A P_s - Q_s$ where $A \in \mathbb{N}, A \geq 2$, and $\text{sbf}_{\Gamma_s}(x) = \text{sbf}_{\Omega_s}(x)$ otherwise. So looking at (7), the new local analysis depends on where the value of $x$ (that makes the left hand side of the equation, which represent the resource demand, equal to the right hand side, which is the supply bound function) is located in the above mentioned ranges. If that value of $x$ is in the range that makes $\text{sbf}_{\Gamma_s}(x) < \text{sbf}_{\Omega_s}(x)$ then it will decrease the subsystem normal budget, otherwise, it will not. The amount of improvement when using the new tighter local analysis compared with the original analysis, on the system load, depends on many factors such as the size of $X_s$, the subsystem period and the difference between...
the subsystem period and tasks’ deadlines. The higher the value of \( X_s \), the more improvement can be achieved. Also, if the difference between the subsystem’s period and its tasks’ deadlines is low, then the improvement in system load becomes higher. If the difference between the subsystem period and its tasks’ deadlines is high, then the \( x \) that makes the left and right hand side of (7) equal becomes very far from the subsystem period. In this case a small increment in the subsystem’s normal budget will be enough to cover the difference between \( \text{sbf}_{\text{r,s}}(x) \) and \( \text{sbf}_{\text{f,s}}(x) \), which also affects the improvement in the system load.

Now we will explain the impact of the new tighter global analysis on the system load, and we will use Figure 5 for illustration. When the \( X_s/\text{load}_{\text{sys}} \) part in Figure 5 is as large as possible, the new global analysis contributes with a larger improvement. The reason for this is that during this part there will be no (or limited) preemptions from higher priority subsystems. Hence, the difference between this part and the other part \( (I + Q_s)/\text{load}_{\text{sys}} \) should be low to achieve a greater improvement, where \( I \) is the interference from higher priority subsystems (including the sum of \( Q_r + X_r \) of the higher priority subsystems) and also the blocking from lower priority subsystems. We can distinguish some cases in which the new global analysis can not reduce the system load. First, if the subsystem period of all subsystems are equal, then there will be no preemptions from higher priority subsystems during the overrun time. Since the tighter global analysis is based on removing the interference from higher priority subsystems during the overrun from the global analysis, the new tighter global analysis can not decrease the system load. Another case where the new global analysis can not decrease the system load, is when the subsystem that requires maximum CPU resources (i.e., the system load was computed based on its CPU requirement), is the highest priority subsystem or does not access a global shared resource.

Finally, combining both the local improvement and the global improvement can require lower system load. As mentioned previously, the new tighter local analysis has a potential to decrease the subsystem budget which will decrease the interference from higher priority subsystems and the budget of the subsystem itself \((I + Q_s)\).

For simplicity, we assume that the internal resource ceilings of the globally shared resources are equal to the highest task priority in each subsystem (i.e., \( r_{\text{c,sil}} = 1 \)), and \( T_l = D_l \) for all tasks. For each simulation study the following settings are changed and a new 1000 systems is generated:

1) Critical section execution time \( CS_s \). It specifies the maximum absolute time that a task may access a global shared resource. Changing this parameter does not require to generate new 1000 system, since changing only this parameter will not have effect on the other task parameters as we will show later.
2) Subsystem period \( P_s \) and task period \( T_{si} \). The subsystem/task period is specified as a range with a lower and upper bound. The simulation program generates a subsystem/task period randomly within the specified range, following a uniform distribution.
3) Number of subsystems \( N \).
4) System utilization \( U^S \). The sum of the utilization of all tasks in the system, is specified to a desired value.

The given system utilization is divided randomly among the subsystems. The assigned utilization to each subsystem is in turn divided randomly to the tasks that belong to that subsystem. Since the task period is generated to a value within the interval as specified, the execution time is derived from the desired task utilization. The critical section execution time is given as an input parameter, however, its value can not be greater than the execution time of its task. The critical section is therefore set to the minimum value of the task execution time and the given critical section execution time, i.e., \( c_{\text{sil}} = \min(CS_s, CS_i) \). All randomized system parameters are generated following uniform distributions.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure5.png}
\caption{Considering MONP analysis for \( S_s \).}
\end{figure}

\section{Simulation setting}

The simulation is performed by applying the modified overrun without payback analysis (MONP), including the new tighter local and global analysis, on 1000 different randomly generated systems. Initially, we assumed that each system consists of 5 subsystems and each subsystem contains 4 tasks. A task is assumed to access at most one globally shared resource and 2 tasks in each subsystem access globally shared resources and we assume that there is only one global shared resource.

\subsection*{B. Simulation results}

We have performed 4 different simulation studies and selected some range of values in order to highlight some properties of the new analysis as described below:

- **Study 1** is specified having critical section execution time \( CS_s \in \{2, 4, 6, 8\} \), task periods \( T_{si} \in \{140, 1000\} \), subsystem periods \( P_s \in \{40, 70\} \), \( U^S = 20\% \) and \( N = 5 \).
- **Study 2** increase the range of the subsystem periods \( P_s \) and task periods \( T_{si} \) (compared to Study 1) to
  - A.) \( P_s \in \{50, 200\} \) and \( T_{si} \in \{400, 1000\} \).
  - B.) \( P_s \in \{100, 200\} \) and \( T_{si} \in \{400, 1000\} \).
- **Study 3** change the number of subsystems (compared to Study 1) to \( N \in \{4, 6, 8\} \).
Study 4 change the system utilization (compared to Study 1) to $U^S \in \{10\%, 30\%\}$ with $CS_s = 2$.

<table>
<thead>
<tr>
<th>$CS_s$</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 load_sys ONP</td>
<td>0.905</td>
<td>0.664</td>
<td>0.763</td>
<td>0.886</td>
</tr>
<tr>
<td>Median load_sys ONP</td>
<td>0.832</td>
<td>0.747</td>
<td>0.849</td>
<td>0.940</td>
</tr>
<tr>
<td>Q3 load_sys ONP</td>
<td>0.560</td>
<td>0.771</td>
<td>0.929</td>
<td>$&gt;1$</td>
</tr>
<tr>
<td>schedulable ONP</td>
<td>100%</td>
<td>100%</td>
<td>89.7%</td>
<td>67.1%</td>
</tr>
<tr>
<td>Q1 load_sys MONP</td>
<td>0.475</td>
<td>0.631</td>
<td>0.792</td>
<td>0.960</td>
</tr>
<tr>
<td>Median load_sys MONP</td>
<td>0.495</td>
<td>0.665</td>
<td>0.770</td>
<td>0.840</td>
</tr>
<tr>
<td>Q3 load_sys MONP</td>
<td>0.516</td>
<td>0.696</td>
<td>0.837</td>
<td>0.940</td>
</tr>
<tr>
<td>schedulable MONP</td>
<td>100%</td>
<td>100%</td>
<td>98.3%</td>
<td>84.7%</td>
</tr>
<tr>
<td>MONP/ONP med. improv.</td>
<td>7.3%</td>
<td>9.4%</td>
<td>10.2%</td>
<td>11.1%</td>
</tr>
<tr>
<td>MONP/ONP max. improv.</td>
<td>13.2%</td>
<td>19.6%</td>
<td>25.7%</td>
<td>30.0%</td>
</tr>
</tbody>
</table>

### Table III

RESULTS OF STUDY 1

Table III shows results of Study 1. The results of each method (ONP and MONP) are shown using the median, lower quartile (Q1) and the higher quartile (Q3) of the system load values of the 1000 generated systems. It also shows the percentage of schedulable systems out of the 1000 generated systems. In addition, it shows the percentage of improvement in the system load based on the evaluated median (explained above) and the maximum improvement when using MONP compared with ONP. It is calculated as $100 \times (load_{sys}^{ONP} - load_{sys}^{MONP}) / load_{sys}^{MONP}$, where load\_sys\_ONP is the median or maximum system load, depending on what is required to be evaluated.

For the case of $CS_s = 8$, some of the systems are unschedulable (i.e., having load\_sys > 1) using both ONP and MONP, because the systems that have load\_sys > 100%. It is not important to find the actual system load for unschedulable systems.

Looking at the results in Table III, it is clear that MONP can give better results compared to traditional ONP in terms of a lower system load and more schedulable systems when increasing $CS_s$ (same results are shown in Tables IV and V). In this study, the ratio $CS_s/P_s$ is relatively high and this is the reason why MONP performs significantly better than ONP. This is the main characteristics that we are looking for.

In Study 2, we decrease the ratio $CS_s/P_s$ by increasing the range of subsystem period. In Table IV, the subsystem period is selected as $P_s = [50, 200]$. In this case, the improvement that MONP can achieve is less than in Study 1 because $X_s/load\_sys$ becomes less significant within the subsystem period $P_s$. In Table V, we change the range of subsystem period to $P_s = [100, 200]$. This causes MONP to give better results compared to the results in Table IV. The reason for this improvement is that when the difference between the minimum and maximum subsystem period is decreased, then also the maximum number of interferences (preemptions) from higher priority subsystems is decreased. This will decrease the contribution of the higher priority subsystems in equations (12) - (19) which in turn decreases the required subsystem load when using MONP.

Looking at the MONP/ONP med. improv. line in Table IV and Table V, the rate of the improvement when increasing $CS_s$ from 2 to 4 is lower when increasing $CS_s$ from 4 to 6 and when increasing $CS_s$ from 6 to 8 (for example in Table IV, the difference between 4.9% - 3.1% is higher than 5.6% - 4.9% ). The reason for this is that increasing $CS_s$ of tasks will increase the required subsystems normal budgets for their subsystems (see (7)) which, in turn, will increase the interference from higher priority subsystems in (12) - (19) and that limits the improvement that MONP can achieve.

\begin{table}
<table>
<thead>
<tr>
<th>$CS_s$</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 load_sys ONP</td>
<td>0.444</td>
<td>0.538</td>
<td>0.607</td>
<td>0.659</td>
</tr>
<tr>
<td>Median load_sys ONP</td>
<td>0.462</td>
<td>0.562</td>
<td>0.644</td>
<td>0.704</td>
</tr>
<tr>
<td>Q3 load_sys ONP</td>
<td>0.481</td>
<td>0.588</td>
<td>0.683</td>
<td>0.755</td>
</tr>
<tr>
<td>schedulable ONP</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>99.8%</td>
</tr>
<tr>
<td>Q1 load_sys MONP</td>
<td>0.430</td>
<td>0.512</td>
<td>0.573</td>
<td>0.620</td>
</tr>
<tr>
<td>Median load_sys MONP</td>
<td>0.448</td>
<td>0.536</td>
<td>0.609</td>
<td>0.661</td>
</tr>
<tr>
<td>Q3 load_sys MONP</td>
<td>0.467</td>
<td>0.563</td>
<td>0.643</td>
<td>0.708</td>
</tr>
<tr>
<td>schedulable MONP</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>MONP/ONP med. improv.</td>
<td>6.1%</td>
<td>4.9%</td>
<td>6.8%</td>
<td>6.2%</td>
</tr>
<tr>
<td>MONP/ONP max. improv.</td>
<td>8.1%</td>
<td>12.3%</td>
<td>16.2%</td>
<td>16.8%</td>
</tr>
</tbody>
</table>

### Table IV

RESULTS OF STUDY 2A, $P_s \in [50, 200]$

In Study 3, we investigate the effect of changing the number of subsystems. The results are shown in Table VI. We can see that increasing $N$ will decrease the improvement that MONP can achieve over ONP. The reason for this is that increasing the number of subsystems will increase the interference $I$ of the higher priority subsystems which, in turn, will decrease the improvement as explained in the previous section.

Finally, in Study 4 we investigate the effect of changing the system utilization on the performance of MONP. The results in Table VII show that increasing the value of $U^S$ will decrease the improvement that MONP can achieve over ONP. The reason for this is that increasing the value
of $U^S$ will increase the subsystem normal budget for all subsystems which increases the contribution of the higher priority subsystems in (12) - (19) and will limit the potential improvement of MONP as explained previously.

<table>
<thead>
<tr>
<th>$N$</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 load$_{sys}$ ONP</td>
<td>0.459</td>
<td>0.505</td>
<td>0.560</td>
<td>0.674</td>
</tr>
<tr>
<td>Median load$_{sys}$ ONP</td>
<td>0.483</td>
<td>0.531</td>
<td>0.590</td>
<td>0.708</td>
</tr>
<tr>
<td>Q2 load$_{sys}$ ONP</td>
<td>0.606</td>
<td>0.660</td>
<td>0.717</td>
<td>0.743</td>
</tr>
<tr>
<td>schedulable ONP</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Q1 load$_{sys}$ MONP</td>
<td>0.430</td>
<td>0.475</td>
<td>0.526</td>
<td>0.637</td>
</tr>
<tr>
<td>Median load$_{sys}$ MONP</td>
<td>0.448</td>
<td>0.495</td>
<td>0.549</td>
<td>0.669</td>
</tr>
<tr>
<td>Q3 load$_{sys}$ MONP</td>
<td>0.467</td>
<td>0.516</td>
<td>0.575</td>
<td>0.702</td>
</tr>
<tr>
<td>schedulable MONP</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>MONP/PONP med. improv.</td>
<td>7.8%</td>
<td>7.4%</td>
<td>7.3%</td>
<td>5.9%</td>
</tr>
</tbody>
</table>

Table VI
RESULTS OF STUDY 3 FOR $CS_s = 2$

VIII. CONCLUSION

In this paper we have shown that existing global and local schedulability analysis of synchronization protocols based on SRP and overrun without payback for two-level hierarchical scheduling based on FPPS is pessimistic. We presented a new tighter global and local analysis assuming that the deadline of a subsystem holds for the sum of its normal budget and its overrun budget, and shown that the global analysis is both uniform and sustainable. We have illustrated the improvements by means of examples, and have evaluated the improvement through an extensive simulation study. The evaluation results show that our novel analysis can improve the CPU requirement significantly for certain cases especially when the ratio between $X_s/P_s$ is high, which makes the performance of the existing analysis low.

REFERENCES


