Exact best-case response time analysis of real-time tasks under fixed-priority pre-emptive scheduling for arbitrary deadlines

Reinder J. Bril

Techn. Universiteit Eindhoven (TU/e)
Den Dolech 2, 5600 AZ Eindhoven,
The Netherlands
r.j.bril@tue.nl

Liliana Cucu-Grosjean

INRIA Nancy-Grand Est, TRIO team
615 rue du Jardin Botanique, Villers les Nancy,
54600, France
Liliana.Cucu@loria.fr

Joël Goossens

Université Libre de Bruxelles (ULB)
Boulevard du Triomphe - C.P.212, 1050 Brussels,
Belgium
Joel.Goossens@ulb.ac.be

Abstract

In this paper, we present a conjecture for exact best-case response times of periodic released, independent real-time tasks with arbitrary deadlines that are scheduled by means of fixed-priority pre-emptive scheduling (FPPS). We illustrate the analysis by means of an example. Apart from having a value on its own whenever timing constraints include lower bounds *on response times of a system to events, the novel analysis allows for an improvement of existing end-to-end response time analysis in distributed systems, i.e. where the finalization of one task on a processor activates another task on another processor.*

1. Introduction

Real-time systems are systems that provide *correct* and *timely* responses to events in their environment. The term timely means that the timing constraints imposed on these responses must be met. The real-time software of these systems is typically designed as a set of tasks and a scheduling algorithm that determines the order in which the tasks are executed. In such a setting, the timing constraints on the responses of the system give rise to derived timing constraints on the responses of the tasks. In this paper, we consider fixed-priority pre-emptive scheduling (FPPS), which is currently considered to be a de-facto standard for real-time scheduling in industry. Typically, timing constraints are interpreted as *upper bounds* on response times of a system and its tasks to events, i.e. responses should not be *too late*. Accordingly, the vast majority of books and papers addressing systems based on FPPS focus on methods for worst-case analysis in general and worst-case response time analysis in particular.

Whenever timing constraints include *lower bounds* on response times of a system to events, i.e. when responses should not be *too early*, methods for best-case analysis become important as well[1]. A well-known example is an airbag, which must neither be inflated too early nor too late upon a collision. Another example is WiseMAC [4], where information must be sent in intervals of time during which the receiver is awake. Notably, the seminal work on response time analysis for FPPS by Harter [7, 8] already covers both worst-case and best-case response time analysis. The need for best-case response time analysis has later also been identified by others in the area of (finalization) jitter of periodic tasks in general and in the area of distributed systems in particular [3, 9, 10, 17, 18].

Worst-case response time analysis for FPPS has been addressed extensively in the literature, and many restrictions of the original scheduling model [13] have been lifted in later work. As examples, [16] introduced the notion of a sporadic task next to a periodic task, [12] address (worst-case) relative deadlines smaller than periods and [11, 22] (worst-case) relative deadlines larger than periods, [20] lifted independent tasks to tasks with mutual access to

[1] Although a minimum delay of a response can also be guranteed by means of additional mechanisms such as buffering, these mechanisms will increase the complexity and potentially the cost of a system [17].

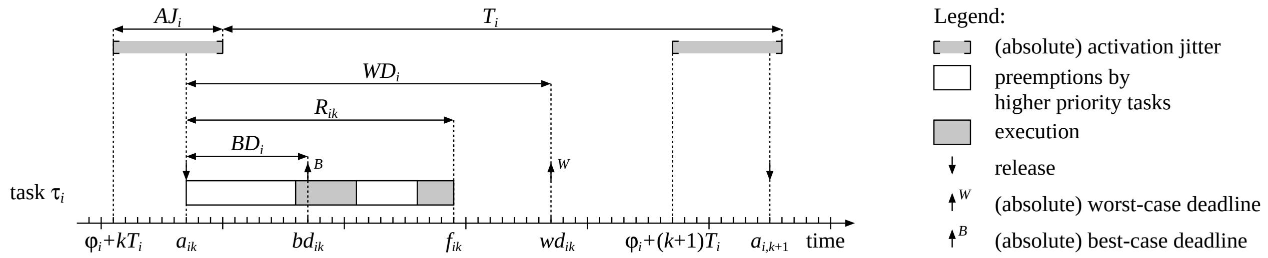

Figure 1. Basic model for a periodic task τ_i with (absolute) activation jitter AJ_i.

shared resources (other than the processor) by presenting the priority ceiling protocol, [1, 22] address tasks with activation jitter, [6, 14, 15, 21] consider tasks with a specific phasing rather than arbitrary phasing, [5] introduced FPPS with varying priorities, and [19, 23] address scheduling with pre-emption thresholds. The scheduling models for best-case response time analysis [3, 8, 18] are considerably less advanced, however. Compared to the original scheduling model, the following advancements are facilitated: (worst-case) relative deadlines are also allowed to be smaller than periods and tasks can have activation jitter.

In this paper, we improve existing analysis by presenting a conjecture for exact best-case response time analysis for tasks with *arbitrary deadlines*. We illustrate the analysis by means of an example.

This paper is organized as follows. We present our scheduling model for FPPS in Section 2 and we briefly recapitulate existing best-case response analysis in Section 3. Our conjecture for exact best-case response analysis for arbitrary deadlines is the topic of Section 4. In Section 5, we present an example illustrating our novel analysis. The paper is concluded in Section 6.

2. A basic scheduling model for FPPS

We assume a uniprocessor system 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, i.e. tasks are scheduled using FPPS.

Each task τ_i generates an infinite sequence of *jobs* ι_{ik} with $k \in \mathbb{Z}$. The inter-activation times of τ_i are characterized by a (fixed) *period* $T_i \in \mathbb{R}^+$ and an (*absolute*) *activation jitter* $AJ_i \in \mathbb{R}^+ \cup \{0\}$, where $AJ_i < T_i$. Moreover, τ_i is characterized by a *best-case computation time* $BC_i \in \mathbb{R}^+$, a *worst-case computation time* $WC_i \in \mathbb{R}^+$, 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 $\mathcal{T}$. The deadlines BD_i and WD_i are relative to the activations.

Note that the activations of τ_i do not necessarily take place strictly periodically with period T_i, but somewhere in an interval of length AJ_i that is repeated with period T_i. The activation times a_{ik} of τ_i satisfy $\sup_{k,\ell}(a_{ik}(\varphi_i) - a_{i\ell}(\varphi_i) - (k-\ell)T_i) \leq AJ_i$, where φ_i denotes the start of the interval in which job zero is activated, i.e. $\varphi_i + kT_i \leq a_{ik} \leq \varphi_i + kT_i + AJ_i$. A task with activation jitter equal to zero is termed a *strictly periodic* task.

The *active interval* of job ι_{ik} is defined as the time span between the activation time a_{ik} of that job and its finalization time f_{ik}, i.e. $[a_{ik}, f_{ik})$. The *response time* R_{ik} of job ι_{ik} is defined as the length of its active interval, i.e. $R_{ik} = f_{ik} - a_{ik}$. Figure 1 illustrates the above basic notions for an example job of a periodic task τ_i.

We assume that we do not have control over the phasing φ, so we assume that any arbitrary phasing may occur. We also assume other standard basic assumptions [13], i.e. tasks are ready to run upon their activation and do no 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, i.e. each job of a task must be completed after its best-case deadline and before its worst-case deadline. Hence, a set $\mathcal{T}$ of n tasks can be scheduled if and only if

$$BD_i \leq R_{ik} \leq WD_i \tag{1}$$

for all $i = 1, \ldots, n$ and all $k \in \mathbb{Z}$.

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.

3. Existing best-case response time analysis

The *best-case response time* BR_i of a task τ_i is the smallest (relative) response time of any of its jobs, i.e.

$$BR_i \stackrel{\text{def}}{=} \inf_{\varphi,k} R_{ik}(\varphi). \tag{2}$$

For worst-case deadlines at most equal to periods minus activation jitter, i.e. $WD_i \leq T_i - AJ_i$, the best-case response time BR_i is given by the *largest* $x \in \mathbb{R}^+$ that satisfies

$$x = BC_i + \sum_{j<i} \left(\left\lceil \frac{x - AJ_j}{T_j} \right\rceil - 1 \right)^+ BC_j. \tag{3}$$

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 can not become negative. To calculate BR_i, we can 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.

As described and illustrated in [18], the largest solution of (3) is a lower bound for worst-case deadlines larger than periods minus activation jitter, i.e. $WD_i > T_i - AJ_i$. For $T_i - AJ_i \geq WR_i$, we know that a job of task τ_i can never delay a next job, and the existing best-case response time analysis therefore remains exact.

4. A conjecture for arbitrary deadlines

When the worst-case relative deadline WD_i of a task τ_i is larger than its period T_i minus its activation jitter AJ_i, the execution of a job of τ_i may be delayed by the previous job. The longest interval of time in which jobs of a task can delay subsequent jobs is the worst-case length WL_i of a so-called level-i active period [2], which is found for the smallest $x \in \mathbb{R}^+$ that satisfies the following equation

$$x = \sum_{j \leq i} \left\lceil \frac{x + AJ_j}{T_j} \right\rceil WC_j. \tag{4}$$

Such a smallest value exists when either (*i*) the utilization factor $U^{\mathcal{T}}$ is smaller than 1 or (*ii*) $U^{\mathcal{T}}$ is equal to 1, the activation jitter of all tasks of $\mathcal{T}$ are equal to zero, and the least common multiple of all tasks of $\mathcal{T}$ exists [2]. To calculate WL_i, we can use an iterative procedure based on recurrence relationships. The maximum number $w\ell_i$ of jobs of task τ_i in a level-i active period is given by

$$w\ell_i = \left\lceil \frac{WL_i + AJ_i}{T_i} \right\rceil. \tag{5}$$

For best-case response time analysis of tasks under FPPS, we only need to consider the last job of a task τ_i in a level-i active period, because that job is the only job in the active period with a response time at most equal to T_i [2]. We now determine the best-case response time of a task τ_i by reusing (3) for $w\ell_i$ fictive tasks τ'_i with best-case computation times $(k+1) \cdot BC_i$, where $0 \leq k < w\ell_i$.

Conjecture 1 *The* best-case response time BR_i *of task* τ_i *with* $T_i - AJ_i < WD_i$ *is given by*

$$BR_i = \max_{0 \leq k < w\ell_i} \left(BR'_i((k+1) \cdot BC_i) - \begin{cases} 0 & k = 0 \\ kT_i + AJ_i & k > 0 \end{cases} \right), \tag{6}$$

where $w\ell_i$ is the worst-case number of jobs of τ_i in a level-i active period, and $BR'_i((k+1)BC_i)$ is the best-case response time of a fictive task τ'_i with a best-case computation time $BC'_i = (k+1)BC_i$, a period equal to its worst-case deadline, i.e. $T'_i = WD'_i$, and a worst-case deadline WD'_i equal to

$$WD'_i = WD_i + \begin{cases} 0 & k = 0 \\ kT_i - AJ_i & k > 0 \end{cases}, \tag{7}$$

and a best-case deadline BD'_i equal to $BD_i + k \cdot BC_i$.

We can start the calculation with $k = w\ell_i - 1$ and use WL_i as initial value for the iterative procedure to determine $BR'_i(w\ell_i \cdot BC_i)$. For next steps, we can use the previously found BR'_i value as initial value, obviating the need to determine WR'_i for each fictive task τ'_i. Note that for $w\ell_i = 1$, (6) becomes equal to the solution of (3). Hence, the conjecture therefore applies for tasks with arbitrary deadlines.

5. An example

For illustration purposes, we use an example task set $\mathcal{T}_1$ with characteristics given in Table 1, and determine the best-case response time BR_3 of task τ_3. In this example, best-case computation times are equal to worst-case computation times. The processor utilization $U^{\mathcal{T}_1} = \frac{69}{70} < 1$, hence

task	T	AJ	C	WR	BR
τ_1	4	0	2	2	2
τ_2	5	0	1	3	1
τ_3	7	0.6	2	8.6	2.4

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

the smallest value of (4) exists for all three tasks of $\mathcal{T}_1$. The worst-case length WL_3 of the level-3 active period is equal to 20, and we therefore find $w\ell_3 = \lceil \frac{WL_3 + AJ_3}{T_3} \rceil = \lceil \frac{20.6}{7} \rceil = 3$. Using Conjecture 1, we get $BR_3 = \max(17 - (14 + 0.6), 9 - (7 + 0.6), 2) = \max(2.4, 1.4, 2) = 2.4$. A time-line for $\mathcal{T}_1$ with a best-case response time $BR_3 = 2.4$ for task τ_3 is shown in Figure 2.

Using (3) of the existing analysis for this example yields a value $BR_3 = 2$, which is pessimistic, i.e. too small. Hence, our novel analysis for best-case response times can improve end-to-end response time analysis in distributed systems [3, 9, 10, 17, 18].

6. Conclusion

In this paper, we presented a conjecture for exact best-case response time analysis for periodically released, independent real-time tasks with arbitrary deadlines that are

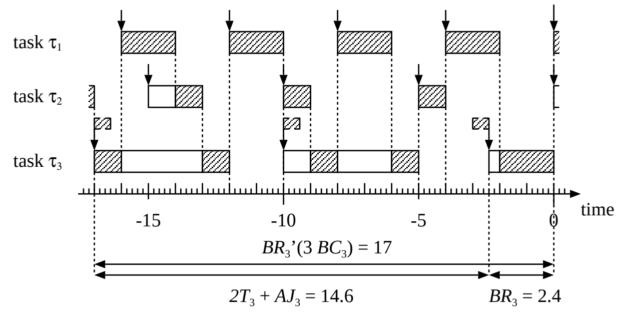

Figure 2. A timeline for $\mathcal{T}_1$ with a best-case response time $BR_3 = 2.4$ for task τ_3.

scheduled using FPPS, and illustrated the analysis by means of an example. Apart from having a value on its own whenever timing constraints include *lower bounds* on response times of a system to events, our novel analysis allows for an improvement of existing end-to-end response time analysis in distributed systems. A formal proof of our conjecture is currently under study.

References

[1] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. Wellings. Applying new scheduling theory to static priority pre-emptive scheduling. *Software Engineering Journal*, 8(5):284–292, 1993.

[2] R. Bril, J. Lukkien, and W. Verhaegh. Worst-case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption. *Accepted for publications in the Real-Time Systems journal (print)*, March 22^{nd} 2009. http://www.springerlink.com/content/f05r404j63424h27 (online), April 28^{th}, 2009.

[3] R. Bril, E. Steffens, and W. Verhaegh. Best-case response times and jitter analysis of real-time tasks. *Journal of Scheduling*, 7(2):133–147, March 2004.

[4] A. El-Hoiydi and J.-D. Decotignie. WiseMAC: An ultra low power MAC protocol for the downlink of infrastructure wireless sensor networks. In *Proc.* 9^{th} *IEEE International Symposium on Computers and Communications, Vol. 1*, pp. 244–251, June/July 2004.

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

[6] 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, Sep. 1997.

[7] P. Harter. Response times in level-structured systems. Technical Report CU-CS-269-84, Department of Computer Science, University of Colorado, USA, 1984. http://www.cs.colorado.edu/department/publications/reports/docs/CU-CS-269-84.pdf.

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

[9] R. Henia, R. Racu, and R. Ernst. Improved output jitter calculation for compositional performance analysis of distributed systems. In *IEEE Int. Parallel and Distributed Processing Symposium (IPDPS)*, pp. 1–8, March 2007.

[10] T. Kim, J. Lee, H. Shin, and N. Chang. Best case response time analysis for improved schedulability analysis of distributed real-time tasks. In *Proc. ICDCS Workshop on Distributed Real-Time Systems*, pp. B14–B20, 2000.

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

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

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

[14] J. Mäki-Turja and M. Nolin. Efficient response-time analysis for tasks with offsets. In *Proc.* 10^{th} *International Conference on Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 462–471, May 2004.

[15] 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)*, pp. 127–136, July 2005.

[16] A.-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.

[17] J. Palencia Gutiérrez, J. Gutiérrez García, and M. González Harbour. Best-case analysis for improving the worst-case schedulability test for distributed hard real-time systems. In *Proc.* 10^{th} *EuroMicro Workshop on Real-Time Systems*, pp. 35–44, June 1998.

[18] 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)*, pp. 165–172, June 2002.

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

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

[21] K. Tindell. Adding time-offsets to schedulability analysis. Report YCS 221, Department of Computer Science, University of York, January 1994.

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

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