MISTA 2009

Schedulability Analysis for non Necessarily Harmonic Real-Time Systems with Precedence and Strict Periodicity Constraints using the Exact Number of Preemptions and no Idle Time

Patrick Meumeu Yomsi · Yves Sorel

1 Introduction

Scheduling theory as it applies to hard real-time environments with precedence and strict periodicity constraints — environments where the failure to satisfy any constraint may have disastrous consequences [1], [2] — seems currently to be enjoying a renaissance. The most widely studied problems in this field concern periodic non-preemptive tasks for systems where data are collected throught captors and the control is performed throught actuators [3], [4]. The tasks corresponding to captors and actuators must satisfy a strict periodicity constraint. For reasons of consistency and predictability, we assume in this paper that all tasks are subject to this constraint. The domains considered here include automobiles, avionics, mobile robotics, telecommunications, etc. Although preemptive scheduling algorithms are able to successfully schedule some systems that cannot be scheduled by any non-preemptive scheduling algorithm, the cost of preemption may not be negligible. Therefore, when preemption is allowed, its exact cost has to be explicitly considered in the schedulability conditions in order to avoid wasting resources and provide safety in terms of guaranteeing the right behavior of the system at run-time. In this paper, we address the scheduling problem of hard real-time systems composed of **dependent, strictly periodic,** preemptive tasks in the monoprocessor case. The strictly periodic constraint implies that, for such a system, any task starts its execution at the beginning of its period whereas the dependence constraint implies that any task cannot start its execution before the end of another task preceding it. We assume here that no jitter is allowed at the beginning of each task. To clearly distinguish between the specification level and its associated model, we shall use the term *operation* rather than the commonly used "*task*" [5] which is too closely related to the implementation level.

For systems with the above-mentioned constraints, in [6] we proved that some of them can be eliminated because they are definitely not schedulable, then we solved the problem

INRIA Paris-Rocquencourt,
Domaine de Voluceau BP 105, 78153 Le Chesnay Cedex - France
E-mail: patrick.meumeu@inria.fr

INRIA Paris-Rocquencourt,
Domaine de Voluceau BP 105, 78153 Le Chesnay Cedex - France
E-mail: yves.sorel@inria.fr

for systems with harmonic periods [1] in [7]. Here, we first generalize these results to the case of systems with periods that are not necessarily harmonic. Then, we provide a necessary and sufficient schedulability condition which takes into account the exact number of preemptions for a system with such constraints when no idle time is allowed. That means the processor always executes an operation if there is one to execute. Indeed, even though the cost α of one preemption — the context switching time including the storage as well as the restoration of the context that the processor needs when a preemption occurs — is easy to know for a given processor, it remains a challenging problem to count the exact number of preemptions of each instance for a given operation [8], [9], [7]. As in [10], we consider only predictable processors without cache or complex internal architecture. We consider a set of n strictly periodic preemptive operations $\tau_i, 1 \leq i \leq n$ with precedence constraints. Each operation τ_i is an infinite sequence of instances [2] τ_i^k, $k \in \mathbb{N}^+$, and is characterized by a Worst Case Execution Time (WCET) C_i, not including any approximation of the cost of preemption, as is usually the case in the classical real-time scheduling theory [11], [12], [13], [14], and a period T_i. In each instance τ_i^k, operation τ_i has to execute for C_i time units. The context switching time corresponding to the activation as well as the termination of an operation is constant, and thus is included in the WCET. Regarding the constraints, we have the following information.

The **precedence** constraint is given by a partial order on the execution of the operations. An operation τ_i preceding an operation τ_j is denoted by $\tau_i \prec \tau_j$ which means that $s_i^k \leq s_j^k$, $\forall k \geq 1$ thanks to the result given in [4], s_i^k denotes the start time of τ_i^k. In that paper it was proved that given two operations $\tau_i = (C_i, T_i)$ and $\tau_j = (C_j, T_j)$: $\tau_i \prec \tau_j \Longrightarrow T_i \leq T_j$. Consequently, the operations must be scheduled in an increasing order of their periods corresponding to classical fixed priorities, using Rate Monotonic (RM), where the shorter the period the higher the priority [7], [4]. We re-index operations in such a way that $\tau_1 \prec \tau_2 \prec \cdots \prec \tau_n$, that is to say τ_1 precedes τ_2, τ_2 precedes τ_3 and so on. In the context of this paper we shall use the term "level" rather than priority, level 1 which corresponds to operation τ_1 being the highest, and level n which corresponds to operation τ_n being the lowest.

The **strict periodicity** constraint means that the start times s_i^k and s_i^{k+1} of two consecutive instances corresponding to operation τ_i are **exactly** separated by its period: $s_i^{k+1} - s_i^k = T_i$, $\forall k \geq 1$. The instance started at time $s_i^1 + kT_i$ has $s_i^1 + (k+1)T_i$ as its deadline.

For such a system of operations with precedence and strict periodicity constraints, we propose a method to compute on the one hand the exact number of preemptions, and on the other hand the schedule of the system when no idle time is allowed, i.e. the processor will always execute an operation as soon as it is possible to do so. Although idle time may help the system to be schedulable, when no idle time is allowed it is easier to find the start times of all the instances of an operation according to the precedence relation.

For the sake of readability and without any loss of generality, from now on, although it is not entirely realistic, we will consider the cost of one preemption for the processor to be $\alpha = 1$ time unit. It is worth noticing that the analysis performed here would work even if the preemption cost were not a constant.

The remainder of the paper is structured as follows: section 2 describes the model and gives the notations used throughout this paper. Section 3 provides the definitions we need to take into account the exact number of preemptions in the schedulability analysis presented in section 4. That section explains in detail, on the one hand, our scheduling algorithm

[1] A sequence $(a_i)_{1 \leq i \leq n}$ is harmonic if and only if there exists $q_i \in \mathbb{N}$ such that $a_{i+1} = q_i a_i$. Notice that we may have $q_{i+1} \neq q_i \quad \forall i \in \{1, \cdots, n\}$.

[2] Throughout the paper all subscripts refer to operations whereas all superscripts refer to instances.

which counts the exact number of preemptions and, on the other hand, derives the new schedulability condition. We conclude and propose future work in section 5.

2 Model

Throughout the paper, we assume that all timing characteristics are non-negative integers, i.e. they are multiples of some elementary time interval (for example the "CPU tick", the smallest indivisible CPU time unit). We denote by $\tau_i = (C_i, T_i)$: an operation, T_i: Period of τ_i, C_i: WCET of τ_i without any preemption approximation, $C_i \leq T_i$, α: Temporal cost of one preemption for a given processor, τ_i^k: The k^{th} instance of τ_i, $N_p(\tau_i^k)$: Exact number of preemptions of τ_i in τ_i^k, $C_i^k = C_i + N_p(\tau_i^k) \cdot \alpha$: Preempted Execution Time (PET) of τ_i including its exact preemption cost in τ_i^k, s_i^1: Start time of the first instance of τ_i, $s_i^k = s_i^1 + (k-1)T_i$: Start time of τ_i^k, R_i^k: Response time of τ_i^k.

A *valid schedule* $\mathcal{S}$ for the system taking into account the exact number of preemptions will be yielded by the set of start times of the first instance for all operations: $\mathcal{S} = \{(s_1^1, s_2^1, \cdots, s_n^1)\}$. Since all the operations except the one with the shortest period w.r.t. the precedence relations may be preempted, the execution time of an operation may vary from one instance to another due to the number of preemptions. Therefore, the *preempted execution time* (PET) [10] which corresponds to the WCET augmented with the exact cost due to preemptions for each instance of an operation may also vary from one instance to another. Consequently, the PET denoted C_i^k for instance τ_i^k depends on the instance and on the number of preemptions occurring in that instance. Its computation will be detailed below.

Because we intend to take into account the exact number of preemptions, and because all operations may be preempted, except the first one, i.e. the one with the shortest period, all instances of all operations must be considered since the number of preemptions may be different from one instance to another. We give a schedulability condition for each operation individually according to operations with shorter periods. For each operation, our scheduling algorithm first provides the start time of the first instance, then computes the exact number of preemptions per instance. This individual operation analysis leads, at the end, to a schedulability condition for all operations.

It has been shown in [4], [7] that systems with precedence and strict periodicity constraints repeat identically after a time called the *hyperperiod* which corresponds to the Least Common Multiple (LCM) of the periods of all the operations.

3 Definitions

All the definitions and terminologies used in this section are directly inspired by [10] and are applied here to the case of a model with precedence and strict periodicity constraints.

From the point of view of any operation τ_i, we define the *hyperperiod at level i*, H_i, which is given by $H_i = LCM\{T_j\}_{\tau_j \in sp(\tau_i)}$, where $sp(\tau_i)$ is the set of operations with a period shorter than or equal to that of operation τ_i. The set $sp(\tau_i)$ may include τ_i. It is obvious that H_i time units after the first start time s_i^1 of operation τ_i, the start time of the next instance is exactly the same as that of s_i^1 w.r.t. the start time of the first instances of operations preceding τ_i. This characteristic derives from both the precedence and the strict periodicity constraints. Without any loss of generality we assume that the first operation τ_1 starts its execution at time $t = 0$. Since at each level i the schedule of τ_i repeats indefinitely, it is sufficient to perform the scheduling analysis in the interval $[s_i^1, s_i^1 + H_i]$ for τ_i and $[0, s_n^1 + H_n]$ for the whole set of operations. Therefore, τ_i starts σ_i times in each hyperperiod at level i starting from 0, with $\sigma_i = \dfrac{H_i}{T_i} = \dfrac{LCM\{T_j\}_{\tau_j \in sp(\tau_i)}}{T_i}$.

Because operation τ_i may only be preempted by the set of operations with a period shorter than τ_i denoted $sp(\tau_i)$, then there are exactly σ_i different PETs for operation τ_i. In other words, from the point of view of any operation τ_i, we can define the function π, inflating C_i, as $\pi : \mathbb{N}^+ \times \mathbb{N}^+ \longrightarrow \mathbb{N}^{+\sigma_i} \times \mathbb{N}^+$, where $\pi(C_i, T_i) = \pi(\tau_i) = ((C_i^1, C_i^2, \cdots, C_i^{\sigma_i}), T_i)$, which maps the WCET C_i of operation τ_i into its respective PET C_i^k in each instance τ_i^k when τ_i is schedulable.

Because of the precedence constraints among operations and because we proceed the schedule from the operation with the shortest period towards the operation with the longest period. At each priority level the goal is to fill available time units in the previous schedule thus far obtained, with slices of the WCET of the current operation taking into account the exact number of preemptions, and hence we obtain the next current schedule. Consequently, we represent the previous schedule of every instance τ_i^k of the current operation $\tau_i = (C_i, T_i)$ by an ordered set of T_i time units where some are already executed because of the execution of operations with shorter periods relatively to $\prec$, and the others are still available for the execution of operation τ_i in that instance. We call this ordered set which describes the state of each instance τ_i^k the $\mathcal{M}_i^k$ *T_i-mesoid*. We denote a time unit already executed by an "e" and a time unit still available by an "a". The switch from an a to an e represents a preemption if the WCET of the current operation is strictly greater than the cardinal of the sub-set corresponding to the first sequence of a. Depending on the remaining execution time while filling available time units, this situation may occur again leading therefore to several preemptions which themselves may result in causing others. The cardinal of a sub-set corresponding to a sequence of consecutive time units already executed is called a *consumption*. It will be denoted by its value inside brackets. We enumerate the sequence of available time units according to natural numbers. This enumeration is done from the end of the first sequence of time units already executed in that instance. Each of these natural numbers corresponds to the number of available time units since the end of the first consumption. They represent all the possible PETs of the operation under consideration in the corresponding instance. Each of these natural numbers a_i is called an *availability*. For example, the 13-mesoid $\{e,e,e,a,a,a,e,e,a,a,e,a,a\}$ will be represented by $\{(3),1,2,3,(2),4,5,(1),6,7\}$, $(3),(2),(1)$ are consumptions and $1,2,3,4,5,6,7$ are availabilities. More details on the definition of a *T_i-mesoid* are given in [10].

From the point of view of the current operation $\tau_i = (C_i, T_i)$, there are as many T_i-mesoids as instances in the hyperperiod H_i at level i. Therefore, there are σ_i T_i-mesoids in H_i which will form a sequence of T_i-mesoids. We call $\mathcal{L}_i^b = \left\{\mathcal{M}_i^{b,1}, \mathcal{M}_i^{b,2}, \cdots, \mathcal{M}_i^{b,\sigma_i}\right\}$ the sequence of σ_i *T_i-mesoids* **before** τ_i is scheduled in the current schedule. The process used to build the sequence $\mathcal{L}_i^b$ of operation τ_i will be detailed later. We define for each T_i-mesoid $\mathcal{M}_i^{b,k}$ the corresponding *universe* X_i^k to be the ordered set, compatible with that of the corresponding mesoid, which consists of all the availabilities of $\mathcal{M}_i^{b,k}$. That is to say, all the possible values that C_i^k can take in $\mathcal{M}_i^{b,k}$. Recall that C_i^k denotes the PET of τ_i in τ_i^k, the k^{th} instance of τ_i. Operation τ_i will be said to be *potentially schedulable* if and only if

$$\begin{cases} C_i \in X_i^k \quad \forall k \in \{1, \cdots, \sigma_i\} \\ \mathcal{M}_i^{b,k} \text{ starts with an available time unit for each } k \in \{1, \cdots, \sigma_i\} \end{cases} \tag{1}$$

The first σ_i equations of (1) verify that C_i belongs to each universe at level i. Then, the next σ_i equations verify that every $\mathcal{M}_i^{b,k}$ starts with an availability as no idle time is allowed.

These verifications are necessary for the strict periodicity constraints to be satisfied. As a matter of fact, if a T_i-mesoid starts with a consumption it is not possible to fill the previous schedule with slices of the WCET of the current operation τ_i taking into account the cost of preemption as it belongs to a lower level than those already scheduled w.r.t $\prec$. Therefore its start time is postponed to the end of the consumption in the previous schedule, and thus does not satisfy the strict periodicity constraint of τ_i. In this case the system is not schedulable. Notice that when this situation arises the three non schedulability conditions given in [6] hold. Since $C_i \in \{1, 2, \cdots, T_i\}$, $\forall 1 \leq i \leq n$, let us define the following binary relation on each instance.

$$\mathcal{R}\text{: "availability } a_{i_1} \text{ leads to the same number of preemptions as availability } a_{i_2}\text{",}$$
$$a_{i_1}, a_{i_2} \in \{1, 2, \cdots, T_i\}$$

$\mathcal{R}$ is clearly an equivalence relation on $\{1, 2, \cdots, T_i\}$ (reflexive, symmetric, transitive). Now, since $X_i^k \subseteq \{1, 2, \cdots, T_i\}$, $\forall 1 \leq k \leq \sigma_i$, thus $\mathcal{R}$ is also an equivalence relation on X_i^k and each X_i^k together with $\mathcal{R}$ is a *setoid* [3]. From now on, we consider only the restriction of $\mathcal{R}$ on $X_i^k, k = 1, \cdots, \sigma_i$ because X_i^k represents all the available time units in instance τ_i^k.

Each T_i-mesoid consists of a sequence of time units already executed, i.e. consumptions, due to the schedule of operations with shorter periods, followed or preceded by a sequence of times units still available, i.e. availabilities. Since each switch from an available time unit to an already executed time unit possibly corresponds to a preemption, then according to the value of C_i several preemptions may occur. Among the possible values that C_i can take, those which will lead to the same number of preemptions will be said to be equivalent w.r.t. to $\mathcal{R}$, and thus will belong to the same *equivalence class*. Therefore, the *equivalence classes* of each universe correspond to the subsets of availabilities determined by two consecutive consumptions in the associated mesoid. The start time of the first instance s_i^1 of operation τ_i occurs at least after the end time of that of operation τ_{i-1} in order to satisfy the strict periodicity constraint. Moreover, s_i^1 occurs as soon as possible since no idle time is allowed. The latter statement implies that operation τ_i starts Δ_{i-1} time units after the start time s_{i-1}^1 of the first instance of operation τ_{i-1}. The computation of Δ_{i-1} will be detailed later on. Already, it is worth noticing that Δ_{i-1} is longer than or equal to the response time of τ_{i-1} in its first instance because when the last piece of the PET of τ_{i-1} of t-0.08443 function f transforms a time unit already executed (resp. still available) in the sequence $\mathcal{L}_{i-1}^{a}$ into a time unit already executed (resp. still available) in the sequence $\mathcal{L}_{i}^{b}$ by following an index ψ which enumerates according to natural numbers, the time units (already executed or still available) in the sequence $\mathcal{L}_{i-1}^{a}$ of operation τ_{i-1} after τ_{i-1} is scheduled. ψ starts from the first available time unit of the first mesoid $\mathcal{M}_{i-1}^{a,1}$ towards the last time unit of the last mesoid $\mathcal{M}_{i-1}^{a,\sigma_{i-1}}$

Since the schedule proceeds from the operation with the shortest period corresponding to the highest level, to the one with the longest period corresponding to the lowest level, then for every potentially schedulable operation, we determine its schedule thanks to those with shorter periods. At each priority level i, the basic idea consists in filling availabilities in each mesoid of the sequence $\mathcal{L}_i^b$, before operation τ_i is scheduled, with slices (cardinal of equivalence classes) of its inflated WCET while taking into account the cost of the exact number of preemptions. At each preemption occurrence, α time units add to the remaining execution time of the instance of the operation under consideration. This situation may occur again w.r.t. the remaining execution time, leading therefore to several preemptions which themselves may cause others. This is why it is crucial to calculate the exact number of preemptions. Finally, we obtain for each mesoid the PET, and then the corresponding response time. Determining the worst case among these response times allow us to conclude on the schedulability of operation τ_i w.r.t. $\prec$. When τ_i

3: Determine the start time of the first instance of operation τ_i: $s_i^1 = s_{i-1}^1 + \Delta_{i-1}$ where Δ_{i-1} is the consumption before the first available time unit in the sequence $\mathcal{L}_{i-1}^a$ of operation τ_{i-1}.

4: Build the sequence $\mathcal{L}_i^b = f(\mathcal{L}_{i-1}^a)$ of T_i-*mesoids* of operation τ_i before it is scheduled. This construction consists of σ_i T_i-*mesoids* $\mathcal{M}_i^{b,k}$ with $k = 1, \cdots, \sigma_i$, and is based on a modulo T_i arithmetic using index ψ on the sequence $\mathcal{L}_{i-1}^a$ without forgetting to start at the first available time unit rather than the first time unit as in [10].

5: For each T_i-*mesoid* $\mathcal{M}_i^{b,k}$ resulting from the previous step, build the corresponding universe X_i^k which consists of the ordered set of all availabilities of $\mathcal{M}_i^{b,k}$. Notice that this set corresponds to the set of all possible values that the PET C_i^k of operation τ_i can take in $\mathcal{M}_i^{b,k}$.

6: Build all the equivalence classes for each universe X_i^k. An equivalence class of X_i^k is composed of the subset of availabilities determined by two consecutive consumptions in the associated mesoid $\mathcal{M}_i^{b,k}$. $m \in \mathbb{N}$ in expression $[m]^k$ denotes the subset of X_i^k composed of the availabilities which are preempted m times.

7: Compute both the exact number of preemptions and the PET C_i^k of operation τ_i in each universe $X_i^k, 1 \leq k \leq \sigma_i$, resulting from the previous step thanks to the algorithm inlined in this step. Since τ_i is potentially schedulable, i.e. its WCET C_i belongs to one and only one equivalence class $[\theta_1]^k$ in each universe X_i^k , we must verify that it is actually schedulable given that some preemptions may occur. The recursive inflation of the execution time of the operation, due to preemptions, starts from the value of the WCET. Indeed, the current inflated WCET is obtained by adding the previous inflated WCET and the cost of preemptions incurred by this latter WCET. This explains the following fixed-point algorithm.

$$\begin{cases} \theta_0 = 0 \\ C_i^{k,0} = C_i \in [\theta_1]^k \\ C_i^{k,m} = C_i^{k,m-1} + (\theta_m - \theta_{m-1}) \cdot \alpha \in [\theta_m]^k \quad \forall m \geq 1 \end{cases}$$

This computation stops as soon as either the PET is reached, i.e. two consecutive values of $C_i^{k,j}, j \geq 1$ are equal , or there exists $\mu_2 \geq 1$ such that $C_i^{k,\mu_2} > card(X_i^k)$. In the latter case, operation τ_i is not schedulable. In the first case, $C_i^k = C_i + \sum_{j=1}^{l} (\theta_j - \theta_{j-1}) \cdot \alpha = C_i + N_p(\tau_i^k) \cdot \alpha$.

8: Deduce the image $\tau_i' = \pi(\tau_i) = \left(\left(C_i^1, C_i^2, \cdots, C_i^{\sigma_i}\right), T_i\right)$ of operation τ_i resulting from the previous step.

9: Determine the response time $R_i^k, 1 \leq k \leq \sigma_i$ of operation τ_i in its k^{th} instance, i.e. in the k^{th} T_i-*mesoid*. This is obtained by summing C_i^k with all the consumptions prior to C_i^k in the corresponding mesoid. Deduce the worst-case response time R_i of operation τ_i: $R_i = max_{\{1 \leq k \leq \sigma_i\}}(R_i^k)$. Operation τ_i is schedulable if and only if $R_i \leq T_i$.

10: If $R_i \leq T_i$ then build the sequence $\mathcal{L}_i^a = g(\mathcal{L}_i^b)$, increment i, and go back to step 2 as long as there remain potentially schedulable operations in the system.

11: If $R_i > T_i$, then the system $\{\tau_i = (C_i, T_i)\}_{1 \leq i \leq n}$ is not schedulable.

12: **end for**

Thanks to the above algorithm, a system $\{\tau_i = (C_i, T_i)\}_{1 \leq i \leq n}$, with precedence and strict periodicity constraints where no idle time is allowed and which takes into account the exact number of preemptions, is schedulable if and only if $R_i \leq T_i$ for all $i \in \{1, 2, \cdots, n\}$.

5 Conclusion and future work

We are interested in hard real-time systems with precedence and strict periodicity constraints where it is mandatory to satisfy these constraints. We are also interested in preemption which offers great advantages when seeking schedules. Since classical approaches are based on an approximation of the cost of the preemption in WCETs, possibly leading to an incorrect real-time execution, we proposed an approach that takes its exact cost into account. We proposed a scheduling algorithm which counts the exact number of preemptions for a given system and thus gives a stronger schedulability condition than those in the literature.

Currently, we are adding the latency constraints to our model and we are planning to study the same problem when jitter is allowed on the periods of operations and then, the complexity of our approach. Afterwards, because idle time may increase the possible schedules, we also plan to allow idle time, even though this would increase the complexity of the scheduling algorithm.

References

1. Joseph Y.-T. Leung and M. L. Merrill. A note on preemptive scheduling of periodic, real-time tasks. *Information Processing Letters*, 1980.
2. Ray Obenza and Geoff. Mendal. Guaranteeing real time performance using rma. *The Embedded Systems Conference, San Jose, CA*, 1998.
3. K. Ramamritham. Allocation and scheduling of precedence-related periodic tasks. *IEEE Transactions on Parallel and Distributed Systems*, 1995.
4. L. Cucu and Y. Sorel. Schedulability condition for systems with precedence and periodicity constraints without preemption. In *Proceedings of 11th Real-Time Systems Conference, RTS'03*, Paris, March 2003.
5. J.H.M. Korst, E.H.L. Aarts, and J.K. Lenstra. Scheduling periodic tasks. *INFORMS Journal on Computing 8*, 1996.
6. P. Meumeu Yomsi and Y. Sorel. Non-schedulability conditions for off-line scheduling of real-time systems subject to precedence and strict periodicity constraints. In *Proceedings of 11th IEEE International Conference on Emerging technologies and Factory Automation, ETFA'06, WIP*, Prague, Czech Republic, September 2006.
7. P. Meumeu Yomsi and Y. Sorel. Schedulability analysis with exact number of preemptions and no idle time for real-time systems with precedence and strict periodicity constraints. In *Proceedings of 15th International Conference on Real-Time and Network Systems, RTNS'07*, Nancy, France, March 2007.
8. Burns A., Tindell K., and Wellings A. Effective analysis for engineering real-time fixed priority schedulers. *IEEE Trans. Software Eng.*, 1995.
9. J. Echagüe, I. Ripoll, and A. Crespo. Hard real-time preemptively scheduling with high context switch cost. *In Proceedings of the 7th Euromicro Workshop on Real-Time Systems*, 1995.
10. P. Meumeu Yomsi and Y. Sorel. Extending rate monotonic analysis with exact cost of preemptions for hard real-time systems. In *Proceedings of 19th Euromicro Conference on Real-Time Systems, ECRTS'07*, Pisa, Italy, July 2007.
11. C.L. Liu and J.W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM*, 1973.
12. J. Goossens. Scheduling of offset free systems. *The Journal of Real-Time Systems*, 2003.
13. J. Goossens and Devilliers R. The non-optimality of the monotonic priority assignments for hard real-time offset free systems. *Real-Time Systems*, 1997.
14. M. Grenier, J. Goossens, and N. Navet. Near-optimal fixed priority preemptive scheduling of offset free systems. *14th International Conference on Real-time and Network Systems*, Poitiers, 2006.