Schedulability analysis for a combination of preemptive strict periodic tasks and sporadic tasks

Mohamed Marouf, Laurent George, Yves Sorel *

1 Introduction

We consider the problem of scheduling tasks with strict periods combined with sporadic tasks. Both types of task have fixed priorities and are preemptive. For a task with a strict period, the difference between its starting time and its release time must be identical for every job. Tasks with strict periods are typically in charge of controlling the activities of a system (sensor/actuator, feedback control, ect.). The freshness of the information they use and/or the reactivity of the system are constrained. Indeed, for control tasks, it might be important to control their jitters (the difference between the worst case and the minimum response time) to ensure the stability of the control loop. In this paper, we consider for controlled tasks, the solution satisfying property 1 that minimizes the jitter of tasks with strict periods.

Property 1 *For any task with strict period, (i) the start time of any job of the task must be equal to its release time and (ii) the Worst Case Response Time (WCRT) of the task must be equal to its Worst Case Execution Time (WCET).*

In this paper, we provide a sufficient schedulability condition for the schedulability of tasks with strict periods. We show how to define their first release times such that property 1 is met (based on paper [1]). Tasks with strict periods have the same fixed priority, the highest one. Sporadic tasks all have a lower priority than any task with a strict period. We show in this paper how to define the worst case scenario for the schedulability of sporadic tasks in the presence of tasks with strict periods. Then we propose a schedulability condition for sporadic tasks based on the worst case response time computation.

2 Scheduling tasks with strict periods

A task $\tau_i(C_i, T_i, D_i)$ with the strict period T_i is characterized by (i) a first start time S_i^1, (ii) a strict period T_i equal to the deadline D_i such as the start time of the k^{th} job of task τ_i is given by $S_i^k = S_i^1 + (k \cdot T_i)$, and (iii) a WCET $C_i \leq T_i$.

It has been proved in [2] that a set of tasks Γ is schedulable (sufficient condition) if

$$\sum_{i=1}^{n} C_i \leq g \tag{1}$$

*{Mohamed.Marouf, Laurent.George, Yves.Sorel}@inria.fr. INRIA Paris-Rocquencourt, Domaine de Voluceau BP 105, 78153 Le Chesnay Cedex, France.

where g is the *gcd* of the periods. The start time of each task is given by

$$S_1 = 0,\ S_i = \sum_{j=1}^{i-1} C_j,\ \ i \geq 2 \tag{2}$$

In [1] we gave *a local schedulability condition* when a set of tasks Γ do not satisfy condition (1). In order to prove that Γ is schedulable, we first schedule a sub-set of tasks according to condition (1). Then, for each remaining (candidate) task, we apply the following local schedulability condition iteratively, such that this candidate task, added to a sub-set of tasks already scheduled, leads to a schedulable set of tasks. A candidate tasks $\tau_c(C_c, T_c, D_c)$ is thus schedulable if

$$C_c \leq C_i \cdot \delta\left[T_c mod(T_i) \cdot (T_c mod(2g) + T_i mod(2g))\right] \tag{3}$$

where *mod* is the modulo function and δ is the Kronecker symbol: $\delta(i) = \begin{cases} 1\ if\ \ i = 0 \\ 0\ otherwise \end{cases}$

The start times of a candidate task τ_c are given by

$$\begin{cases} If\ (T_c mod(T_i) = 0)\ then\ S_c^1 = S_i^1 + kg,\ \ 1 \leq k \leq \frac{T_c}{T_i} - 1 \\ If\ (T_i mod(2g) = T_c mod(2g) = 0)\ then\ S_c^1 = S_i^1 + (2k+1)g,\ \ 0 \leq k \leq \frac{T_c}{2g} - 1 \end{cases} \tag{4}$$

Theorem 1 *The scheduling of strict periodic tasks in the time interval* $[kL, (k+1)L]$, $k \in \mathbb{N}^*$, *with start times satisfying equations (2) or (4), is identical to the scheduling obtained in the time interval* $[0, L]$. *L is the LCM (Least Common Multiple) of all the tasks periods.*

Proof According to the conditions (2, 4), $0 \leq S_i^1 < T_i$. The k^{th} start time of a task τ_i is given by $S_i^k = S_i^1 + kT_i$. From the definition of S_i^k, $k \leq 1$, a task released at time S_i^k ends at time $S_i^k + C_i$. Thus, the scheduling of tasks in the time interval $[kL, (k+1)L]$, $k \in \mathbb{N}^*$, is identical to the scheduling obtained in the time interval $[0, L]$, . ⊡

We define the set Ψ which contains all the start times of the strict periodic jobs on a time interval $[0, L[$. Thus, $\Psi = \{S_i^1 + k \cdot T_i,\ k = 0..(\frac{L}{T_i} - 1),\ i = 1..n\}$.

3 Scheduling sporadic tasks

A sporadic task $\tau_i(C_i, T_i, D_i)$ is defined by its WCET C_i, its minimum inter-arrival time T_i and its relative deadline $D_i \leq T_i$. In order to study the schedulability of sporadic tasks when tasks with strict periods have already been scheduled, we have to determine the critical instants for a sporadic job τ_i^j. A critical instant is defined as the instant where a job will have the largest response time if its release time r_i^j is set to it. It has been proved in [3] that a critical instant occurs when the release time of a sporadic job is equal to the release time of a highest priority job. For all sporadic tasks, this results in first releasing all tasks with priority higher than τ_i at the same time as the first release time of τ_i. To consider tasks with strict periods, we have to study all cases of release times $S_i \in \Psi$ following the scenario defined in section 2. However, rather than testing all the start times of Ψ, some start times are useless and will be thus pruned from Ψ according to the following lemma.

Lemma 2 *For a sequence of consecutive executions of strict periodic jobs, only the first release times in the sequence should be considered.*

Thus, if $\exists S_i^k,\ S_j^l \in \Psi$ such that $S_i^k - S_j^l = C_j$, S_i^k is removed from Ψ. Consider Ψ the release times of tasks with strict periods obtained applying equations (2, 4). Consider that task τ_i is first released at time $S \in \Psi$. $W_i(t)$ denotes the sum of the computational requirements at time t (w.r.t time S) of all the tasks with higher priority, plus one execution of τ_i starting from time S. It is given by

$$W_i(t) = C_i + \sum_{\tau_j \in \Gamma_P^{NS}/i} \left\lceil \frac{t}{T_j} \right\rceil C_j + \sum_{\tau_j \in \Gamma^S} \left\lceil \frac{t - s_j}{T_j} \right\rceil C_j \tag{5}$$

where s_j is the relative start time S_j^k according to a release time S of τ_i. s_j is given by

$$s_j = S_j^1 + \left\lceil \frac{S - S_j^1}{T_j} \right\rceil T_j - S \tag{6}$$

The WCRT of τ_i is the solution of $r_i = W(r_i)$ computed by iteration [3].

4 Example

We use the Rate-Monotonic algorithm to schedule the following tasks. Let us consider three tasks with strict periods: $\tau_1(1,4,4)$, $\tau_2(1,6,6)$ and $\tau_3(1,12,12)$. τ_1 and τ_2 satisfy condition (1) and are schedulable with $S_1^1 = 0$, $S_2^1 = 1$. τ_3 and τ_1 satisfy condition (3), thus τ_1 is schedulable with $S_3^1 = 6$. $L = LCM(T_1, T_2, T_3) = 12$ thus $\Psi = \{0, 1, 4, 6, 7, 8\}$. After eliminating the successive jobs according to lemma 2, $\Psi = \{0, 4, 6\}$.
Let $\tau_4(2, 6, 8)$, $\tau_5(2, 12, 12)$ be sporadic tasks to be scheduled.
We have: $W_4(t) = 2 + \sum_{j=1}^{3} \left\lceil \frac{t-s_j}{T_j} \right\rceil C_j$ and $W_5(t) = C_5 + \left\lceil \frac{t}{T_4} \right\rceil C_4 + \sum_{j=1}^{3} \left\lceil \frac{t-s_j}{T_j} \right\rceil C_j$.

S	s_1	s_2	s_3	r_4	r_5
0	0	1	6	4	**12**
4	0	3	2	**6**	8
6	2	1	0	5	**12**

As $r_4 \leq D_4$ and $r_5 \leq D_5$, τ_4 and τ_5 are schedulable, as shown in figure 1.

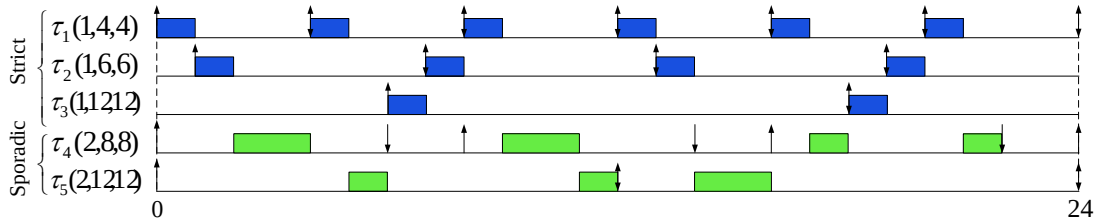

Figure 1: Scheduling diagram of strict periodic and sporadic tasks

References

[1] M. Marouf and Y. Sorel. *Schedulability conditions for non-preemptive hard real-time tasks with strict period*, In Proceedings of 18th International Conference on Real-Time and Network Systems, RTNS10, Toulouse, France, November 2010.

[2] O. Kermia and Y. Sorel. *Schedulability Analysis for Non-Preemptive Tasks under Strict Periodicity Constraints*, In Proceedings of 14th International Conference on Real-Time Computing Systems and Applications, RTCSA'08, Kaohsiung, Taiwan, August 2008.

[3] M. Joseph and P. Pandya. *Finding response times in a real-time system*, British Computer Society Computer Journal, vol. 29, no. 5, pp. 390-395. 1986.