Towards optimal priority assignments for real-time tasks with probabilistic arrivals and execution times

Dorin Maxim

Towards optimal priority assignments for real-time tasks with probabilistic arrivals and execution times

Dorin Maxim
INRIA Nancy Grand Est
615 rue du Jardin Botanique
54600 Villers les Nancy
dorin.maxim@inria.fr

***Abstract*—In this paper we present the problem of optimal priority assignments in fixed priority preemptive single processor systems where tasks have probabilistic arrivals and execution times. We show that Rate Monotic is not optimal for our problem.**

I. Introduction

In embedded real-time systems there is a strong demand for new functionality that can only be met by using advanced high performance microprocessors. Building real-time systems with reliable timing behaviour on such platforms represents a considerable challenge. Deterministic analysis for these platforms may lead to significant overprovision in the system architecture, effectively placing an unnecessarily low limit on the amount of new functionality that can be included in a given system. An alternative approach is to use probabilistic analysis. Probabilistic analysis techniques rather than attempting to provide an absolute guarantee of meeting the deadlines, provide the probability of meeting the deadlines.

II. Model and notations

In this paper, we consider a task set of n synchronous tasks $\{\tau_1, \tau_2, \ldots, \tau_n\}$. Each task τ_i is characterized by three parameters $(\mathcal{C}_i, \mathcal{T}_i, \mathcal{D}_i)$ where $\mathcal{T}_i$ is the inter-arrival time (commonly known as period), $\mathcal{D}_i$ the relative deadline, and $\mathcal{C}_i$ the worst-case execution time. The parameters are described by random variables[1].

A random variable $\mathcal{X}_i$ describing a parameter of τ_i is assumed to have a known probability function (PF) $f_{\mathcal{X}_i}(\cdot)$ with $f_{\mathcal{X}_i}(x) = P(\mathcal{X}_i = x)$ giving the probability that τ_i has the mentioned parameter equal to x. The values of $\mathcal{X}_i$ are assumed to belong to the interval $[x_i^{\min}, x_i^{\max}]$.

For instance the worst-case execution time $\mathcal{C}_i$ can be written as follows:

$$\mathcal{C}_i = \begin{pmatrix} C_i^0 = C_i^{\min} & C_i^1 & \cdots & C_i^{k_i} = C_i^{\max} \\ f_{\mathcal{C}_i}(C_i^{\min}) & f_{\mathcal{C}_i}(C_i^1) & \cdots & f_{\mathcal{C}_i}(C_i^{\max}) \end{pmatrix}, \quad (1)$$

where $\sum_{j=0}^{k_i} f_{\mathcal{C}_i}(C_i^j) = 1$.

[1]In this paper we will use a calligraphic typeface to denote random variables.

For example for a task τ_i we might have a worst-case execution time $\mathcal{C}_i = \begin{pmatrix} 2 & 3 & 25 \\ 0.5 & 0.45 & 0.05 \end{pmatrix}$; thus $f_{C_i}(2) = 0.5$, $f_{C_i}(3) = 0.45$ and $f_{C_i}(25) = 0.05$.

All jobs are assumed to be independent of other jobs of the same task and those of other tasks, hence the execution time of a job does not depend on, and is not correlated with, the execution time of any previous job.

For example, given a tas-kset $\tau = \{\tau_1, \tau_2\}$ composed of two tasks, where $\tau_1 = \left(\begin{pmatrix} 3 & 4 \\ 0.1 & 0.9 \end{pmatrix}, \begin{pmatrix} 1 & 2 \\ 0.7 & 0.3 \end{pmatrix}, \begin{pmatrix} 3 & 4 \\ 0.1 & 0.9 \end{pmatrix}\right)$, and $\tau_2 = \left(\begin{pmatrix} 6 & 8 \\ 0.2 & 0.8 \end{pmatrix}, \begin{pmatrix} 2 & 4 \\ 0.6 & 0.4 \end{pmatrix}, \begin{pmatrix} 6 & 8 \\ 0.2 & 0.8 \end{pmatrix}\right)$, the first random variable of the task representing its probabilistic execution time and the second one, its release distribution.

Let us presume that τ_1 has the higher priority and τ_2 the lower priority. In this case there are multiple scenarios that can occur. We present in Figure 1 and in Figure 2 two of this possible scenarios. In the first one, τ_1 has four jobs represented andτ_2 has two jobs, the first one having an execution time equal to 2 and being released at $t = 0$ and the second job having an execution time of 4 and being released at $t = 6$. Both of these job finish execution before their respective deadlines. In the second scenario, there are represented two jobs of τ_1 and only one job of τ_2, but in this case the job of tau_2 misses its deadline. The difference now is that $\tau_{2,1}$ has an execution time equal to 4 which makes it miss its deadline at $t = 6$.

There are multiple scenarios like this and, in consequence, many questions that arise. We present some of these questions in the next section.

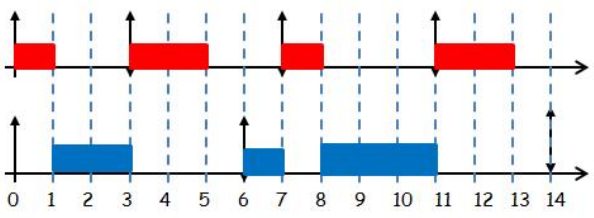

Fig. 1. Scenario 1

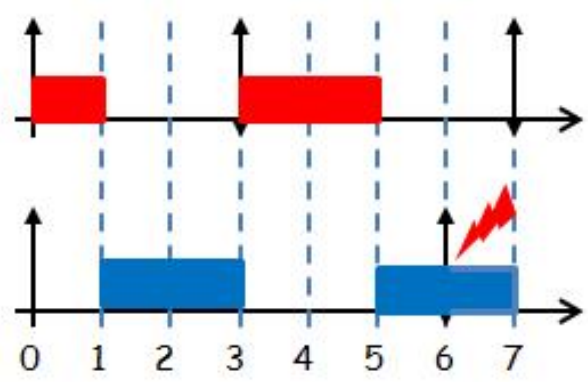

Fig. 2. Scenario 2

III. Open problems

One of the first questions that comes to mind in a probabilistic time system is *how does one analytically compute the response time distributions of the different jobs of given tasks?* This is needed in order to compute the probability that the job misses its deadline, and to know the percentages of deadline misses.

Presuming that such an analytical tool is at hand, the next question that arises, and maybe the most important one in what concerns the design of a real time system is *how should the tasks be scheduled so that each task meets certain conditions referring to its timing failures?* We mention that we are searching for a fixed priority scheduling, in a preemptive context, i.e., all jobs of the same task have the same priority. The timing, or deadline failure, is usually given as a maximum percentage of deadlines that the task can miss in certain time interval. We refer to a priority ordering that meets such a requirement as a feasible priority assignment. We note that this use of the term feasible is an extension of its normal use in the deterministic case, where a feasible priority ordering is one in which the associated schedule has zero probability of timing failure.

The next question that immediately comes to mind is *how does one define a study interval in such a system?* In the deterministic case the study interval is, usually, the hyper-period. Knowing that everything that happens during a hyper-period will repeat for the next ones, it is enough to study one hyper-period to obtain the behavior of the entire system. In the probabilistic case the hyper-period might not be the answer to the question, since what happens in one hyper-period might be completely different than what happened in the previous one and what will happen in the next one. Also, there is the question of how does one compute the hyper-period in a probabilistic system.

In the following we talk about the case of the Rate Monotonic priority assignment algorithm and its compatibility with a probabilistic task-system.

A. Non-optimality of Rate Monotonic

We know from [1] that Rate Monotonic is not optimal for the problem of scheduling tasks according to a fixed-priority policy in the case of tasks with deterministic arrivals and probabilistic executions times.

Following the reasoning applied in [1], it is easy to prove that Rate Monotonic does not provide a feasible scheduling, since it does not take into account the probabilistic character of the tasks. Furthermore, in the case of tasks with probabilistic arrivals it would be difficult to determine the ordering of the tasks since each task may have multiple values representing its arrival time and even if one would apply a convention as considering the minimum value of each arrival time distribution would still not provide a feasible scheduling. For example, considering a task-set $\tau = \{\tau_1, \tau_2\}$ with τ_1 defined by $(\begin{pmatrix} 4 & 10 \\ 0.01 & 0.99 \end{pmatrix}, \begin{pmatrix} 2 & 3 \\ 0.5 & 0.5 \end{pmatrix}, \begin{pmatrix} 4 & 10 \\ 0.01 & 0.99 \end{pmatrix})$ and τ_2 defined by $(\begin{pmatrix} 8 & 9 \\ 0.6 & 0.4 \end{pmatrix}, \begin{pmatrix} 2 & 3 \\ 0.5 & 0.5 \end{pmatrix}, \begin{pmatrix} 8 & 9 \\ 0.6 & 0.4 \end{pmatrix})$, one can easily see that, even if τ_1 has its smallest arrival time smaller than those of τ_2, it very rarely arrives with a period equal to 4, most of the times it has a distance of 10 units of time between instances, which Rate Monotonic does not take into consideration.

Furthermore, since the problem of scheduling tasks according to a fixed-priority policy in the case of tasks with deterministic arrivals and probabilistic executions times is a sub-problem of our problem, we can conclude that Rate Monotonic is not optimal in the case when the arrivals are probabilistic either.

References

[1] D. Maxim, O. Buffet, L. Santinelli, L. Cucu-Grosjean, and R. Davis, "On the optimality of priority assignment for probabilistic real-time systems," in *the 19th International Conference on Real-Time and Network Systems*, 2011.