Response Time Analysis for Fixed-Priority Tasks with Multiple Probabilistic Parameters

Dorin Maxim[1,2,3], Liliana Cucu-Grosjean[1,2,3]
[1] Universite de Lorraine, LORIA, UMR 7503, F-54506, France
[2] CNRS, LORIA UMR 7503, Vandoeuvre-les-Nancy, F54519, France
[3] INRIA Nancy-Grand Est, Villers-les-Nancy, F-54600, France
dorin.maxim@inria.fr, liliana.cucu@inria.fr

***Abstract*—In this paper, we consider a probabilistic model for real-time task systems with probabilistic worst-case execution times, probabilistic minimum inter-arrival times and probabilistic deadlines. We propose an analysis computing response time distributions of the tasks scheduled on one processor under a task-level fixed-priority preemptive scheduling policy. The complexity of our method is analyzed and it is improved by re-sampling techniques on worst-case execution time distributions and/or minimal inter-arrival time distributions. The improvements are shown through experimental results. Also, experiments are conducted in order to investigate the improvement obtained by using a probabilistic model in terms of precision and schedulability gained as opposed to a deterministic worst-case reasoning.**

***Keywords*—probabilistic real-time, fixed-priority, probabilistic worst-case execution time, probabilistic minimum inter-arrival times, probabilistic deadlines**

I. Introduction

Critical real-time embedded systems integrate complex architectures that evolve constantly in order to provide new functionality required by the end users of the systems (automotive, avionics, railway, etc). These new architectures have a direct impact on the variability of the timing behavior of the real-time system. For instance, the use of aggressive hardware acceleration features like caches and deep memory hierarchies imply an acceptable average execution time for a task, but extremely large worst case values may be observed [1]. This variability leads to important over-provisioning if the design of the system is based only on worst case reasoning.

Since the seminal paper of Edgar and Burns [2], probabilistic approaches propose solutions based on the probability of occurrence of the worst case values in order to avoid over provisioning while satisfying real-time constraints. These approaches avoid systematically rejecting systems that are unfeasible according to a deterministic approach[1], when in practice the probability of failure is extremely small compared to the maximum tolerated failure level. For instance, in the aerospace industry, one may compare the maximum allowed probability of failure of 10^{-9} per hour of operation (required by the certification authorities [3]) and the probability of failure of 10^{-15} that the system might experience per hour of operation [1].

Important examples where the worst case analysis is not efficient include event triggered systems that interact with the real world and for which it is difficult to place a bound on the arrival rate of jobs generated by interrupts from external sensors or network interfaces; systems for which their parameters remain unknown until deployment; adaptive systems that need to change their parameters due to changes in the environment in which they are running.

Different industries may also build probabilistic real-time systems on purpose by introducing some parameters with random behaviour. For instance, some automotive manufacturers have randomised the sampling frequency for the reverse parking ultra sound sensor in order to avoid the situation when two vehicles reverse back-to-back as in Figure 1 and they both have the same sampling frequency reducing the efficiency of their parking sensors [4]. By randomising the sampling frequency, the jobs that are generated by the sensor have a random arrival pattern. These jobs belong to a task that can be seen as a sporadic task with its period equal to the minimum inter arrival time (MIT) amongst its jobs, but knowing that job arrivals are random then we may describe the MITs by distributions, i.e., the arrival distribution of the generated jobs.

The use of a probability distribution to describe at least one parameter of the system requires a probabilistic analysis to decide the feasibility of the system. In this paper we introduce for the first time a response time analysis for systems that have both worst case execution times and minimal inter-arrival times described by probability distributions.

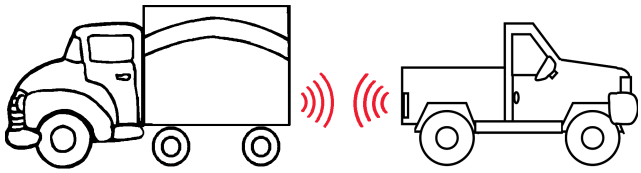


Fig. 1: When two vehicles reverse back-to-back, their respective parking sensors may sample at the same frequency, making them less efficient, or even blind to one another.

Related work: The contribution of this paper belongs to the realm of probabilistic real-time analysis. These analyses exist mainly for real-time systems with random variables describing the duration of a task's execution [5]–[9]. The first papers use the term *probabilistic* or *stochastic execution time* to indicate that a random variable contains all possible values for the execution time of a task. The use of this notion was somehow confusing as the proposed analyses required

[1]We use deterministic as opposed to a probabilistic approach.

1052-8725/13 $26.00 © 2013 IEEE
DOI 10.1109/RTSS.2013.30

independent random variables to perform convolution[2] and the execution times of a task or several tasks may be dependent. The work of Edgar and Burns [2] uses the term *probabilistic worst case execution time* to indicate that the random variable provides the value of the worst case execution time of a task for a given probability of failure. This notion fulfills the hypothesis of independent random variables (Definition 1) as this random variable is common to all instances of a task [10].

Different models are considered for systems with probabilistic inter-arrival times [5], [11]–[15] but only few papers [14], [13], [15] exist for the case that we consider here: systems with probabilistic execution times and probabilistic inter-arrival times. In [14] Lehoczky considers parameters that are identically distributed, i.e., all described by the same probability function. This hypothesis may be restrictive for systems that need a more general model. The recent work of Abeni et al. [15] is probably the closest to our paper. The authors present an analysis framework for tasks with probabilistic execution times and random arrivals, running on a preemptive single processor according to a Constant Bandwidth Server based on Earliest Deadline First. In [13], the authors extend the contribution of [5] to tasks with probabilistic number of arrivals within a time interval t_Δ[3]. Our work extends the results presented in [5] to the case of systems with probabilistic minimal inter-arrival times. With respect to this work we consider probabilistic *worst case* execution times as indicated by [2], whereas Diaz et al. [5] consider probabilistic execution times.

One may note a relation between the non-cyclic Generalised Multi-Frame (GMF) task model presented in [16] and our probabilistic model. In the GMF task model, jobs can take different values for their execution time, inter-arrival time and deadline, be it in a cyclic manner [17], [18] or in a non-cyclic manner [16]. In some respects, a non-cyclic GMF task is a probabilistic task for which we do not provide the associated probabilities of its parameters. Also, a probabilistic task has different conditions of feasibility, i.e., a probabilistic task may have deadline misses provided that their probability of occurrence is below a maximum allowed threshold, e.g., 10^{-9} for aerospace industry.

The random variables describing the parameters of a real-time system may be obtained using techniques like those proposed for obtaining worst case execution times [1], [2], [19] or minimal inter-arrival times [20]–[22]. This paper considers these random variables known and obtaining them is beyond its scope.

Our contribution In this paper, we introduce a novel response time analysis for probabilistic real-time tasks with probabilistic minimum inter-arrival times and probabilistic worst case execution times scheduled preemptively on a single processor according to a task-level fixed-priority scheduling policy. The model that we introduce is general, as there are no constraints on the distributions of the parameters and the associated response time analysis, as well, can handle any type of distributions. We validate the analysis, and we show that it is bounded in the number of steps. We also provide a means of reducing the analysis duration via re-sampling and show the effect that different re-sampling strategies have on the response times distributions of the analysed tasks.

Organization of the paper: The paper is organized as follows. In Section II we introduce preliminary notations and definitions while in Section III we describe the system model and our problem. Section IV contains the main contribution of the paper, the response time analysis, which is validated in Section V. We present in Section VI the experimental evaluation of our probabilistic worst case response time. We conclude in Section VII.

II. Notations and definitions

A random variable $\mathcal{X}$ has a probability function (PF) $f_\mathcal{X}(\cdot)$ with $f_\mathcal{X}(x) = P(\mathcal{X} = x)$. The possible values of $\mathcal{X}_i$ belong to the interval $[X^{\min}, X^{\max}]$. In this paper we associate the probabilities with the possible values of a random variable using the following notation:

$$\mathcal{X} = \begin{pmatrix} X^0 = X^{min} & X^1 & \cdots & X^k = X^{max} \\ f_\mathcal{X}(X^{min}) & f_\mathcal{X}(X^1) & \cdots & f_\mathcal{X}(X^{max}) \end{pmatrix} \quad (1)$$

where $\sum_{j=0}^{k_i} f_\mathcal{X}(X^j) = 1$. A random variable may also be specified using its cumulative distribution function (CDF) $F_\mathcal{X}(x) = \sum_{z=X^{min}}^{x} f_\mathcal{X}(z)$.

Definition 1. *Two random variables $\mathcal{X}$ and $\mathcal{Y}$ are (probabilistically)* **independent** *if they describe two events such that the outcome of one event does not have any impact on the outcome of the other.*

Definition 2. *The sum $\mathcal{Z}$ of two (probabilistically)* **independent** *random variables $\mathcal{X}$ and $\mathcal{Y}$ is the* **convolution** *$\mathcal{X} \otimes \mathcal{Y}$ where $P\{\mathcal{Z} = z\} = \sum_{k=-\infty}^{k=+\infty} P\{\mathcal{X} = k\}P\{\mathcal{Y} = z - k\}$.*

$$\begin{pmatrix} 3 & 7 \\ 0.1 & 0.9 \end{pmatrix} \otimes \begin{pmatrix} 0 & 4 \\ 0.9 & 0.1 \end{pmatrix} = \begin{pmatrix} 3 & 7 & 11 \\ 0.09 & 0.82 & 0.09 \end{pmatrix}$$

A complementary operator to the convolution is the operator $\ominus$, defined by $\mathcal{X} \ominus \mathcal{Y} = \mathcal{X} \otimes (-\mathcal{Y})$.

Definition 3. *The* **coalescion** *of two partial random variables, denoted by the operator $\oplus$ represents the combination of the two partial random variables into a single (partial) random variable so that values that appear multiple times are kept only once gathering the summed probability mass of the respective values.*

$$\begin{pmatrix} 5 & 8 \\ 0.18 & 0.02 \end{pmatrix} \oplus \begin{pmatrix} 5 & 6 \\ 0.72 & 0.08 \end{pmatrix} = \begin{pmatrix} 5 & 6 & 8 \\ 0.9 & 0.08 & 0.02 \end{pmatrix}$$

Definition 4. *[23] Let $\mathcal{X}_1$ and $\mathcal{X}_2$ be two random variables. We say that $\mathcal{X}_1$ is* **greater than** *$\mathcal{X}_2$ if $F_{\mathcal{X}_1}(x) \leq F_{\mathcal{X}_2}(x)$, $\forall x$, and denote it by $\mathcal{X}_1 \succeq \mathcal{X}_2$.*

For example, in Figure 2 $F_{\mathcal{X}_1}(x)$ never goes below $F_{\mathcal{X}_2}(x)$, meaning that $\mathcal{X}_2 \succeq \mathcal{X}_1$. Note that $\mathcal{X}_2$ and $\mathcal{X}_3$ are not comparable.

[2]The convolution is the operation of summation between two random variables, as detailed in Definition 2.

[3]The two models are compared in Appendix A of this paper.

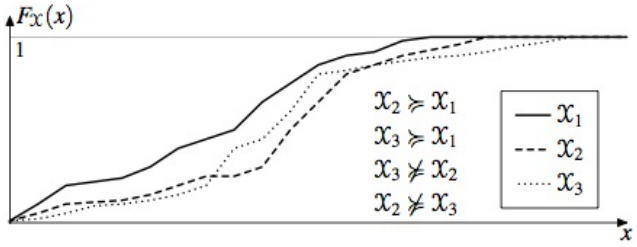


Fig. 2: Possible relations between the CDFs of various random variables

III. Model and problem description

We consider a system of n synchronous tasks $\{\tau_1, \tau_2, \ldots, \tau_n\}$ to be scheduled on one processor according to a preemptive fixed-priority task-level scheduling policy. Without loss of generality, we consider that τ_i has a higher priority than τ_j for $i < j$. We denote by $hp(i)$ the set of tasks' indexes with higher priority than τ_i. By synchronous tasks we understand that all tasks are released simultaneously the first time at $t = 0$.

Each task τ_i generates an infinite number of successive jobs $\tau_{i,j}$, with $j = 1, \ldots, \infty$. All jobs are assumed to be independent of other jobs of the same task and those of other tasks.

Each task τ_i is a generalized sporadic task [24] and it is represented by a probabilistic worst case execution time (pWCET) denoted by $\mathcal{C}_i$[4] and by a probabilistic minimal inter-arrival time (pMIT) denoted by $\mathcal{T}_i$. These notions are defined as follows.

Definition 5. *The probabilistic execution time (pET) of a job of a task describes the probability that the execution time of the job is equal to a given value.*

Definition 6. *The probabilistic worst case execution time (pWCET) of a task describes the probability that the worst case execution time of that task is equal to a given value.*

A safe pWCET $\mathcal{C}_i$ is an upper bound on the pETs $\mathcal{C}_i^j$, $\forall j$ and it may be described by the relation $\succeq$ as $\mathcal{C}_i \succeq \mathcal{C}_i^j$, $\forall j$. Graphically this means that the CDF of $\mathcal{C}_i$ stays under the CDF of $\mathcal{C}_i^j$, $\forall j$.

Following the same reasoning the probabilistic minimal inter-arrival time (pMIT) denoted by $\mathcal{T}_i$ describes the probabilistic minimal inter-arrival times of all jobs.

Definition 7. *The probabilistic inter-arrival time (pIT) of a job of a task describes the probability that the job's arrival time occurs at a given value.*

Definition 8. *The probabilistic minimal inter-arrival time (pMIT) of a task describes the probability that the minimal inter-arrival time of that task is equal to a given value.*

A safe pMIT $\mathcal{T}_i$ is a bound on the pITs $\mathcal{T}_i^j$, $\forall j$ and it may be described by the relation $\succeq$ as $\mathcal{T}_i^j \succeq \mathcal{T}_i$, $\forall j$. Graphically this means that the CDF of $\mathcal{T}_i$ stays below the CDF of $\mathcal{T}_i^j$, $\forall j$.

[4]In this paper, we use calligraphic typeface to denote random variables.

Hence, a task τ_i is represented by a tuple $(\mathcal{C}_i, \mathcal{T}_i)$. A job of a task must finish its execution before the arrival of the next job of the same task, i.e., the arrival of a new job represents the deadline of the current job[5]. Thus, the task's deadline may also be represented by a random variable $\mathcal{D}_i$ which has the same distribution as its pMIT, $\mathcal{T}_i$. Alternatively, we can consider the deadline described by a distribution different from the distribution of its pMIT if the system under consideration calls for such model [11], [25], or the simpler case when the deadline of a task is given as one value. The latter case is probably the most frequent in practice, nevertheless we prefer to propose an analysis as general as possible and in the rest of the paper, we consider tasks with implicit deadlines, i.e., having the same distribution as the pMIT.

As stated in [10], since we consider probabilistic worst case values (for MIT and WCET), then the random variables are (probabilistically) independent.

Definition 9 (Job deadline miss probability)**.** *For a job $\tau_{i,j}$ the deadline miss probability $\mathsf{DMP}_{i,j}$ is the probability that the j^{th} job of task τ_i misses its deadline and it is equal to:*

$$\mathsf{DMP}_{i,j} = P(\mathcal{R}_{i,j} > D_i). \tag{2}$$

where $\mathcal{R}_{i,j}$ is the response time distribution of the j^{th} job of task τ_i.

We show in Theorem 1 that the case when tasks are simultaneously released yields the greatest response time distribution for each task respectively. Here, greatest is defined with respect to the relation $\succeq$ and it indicates that the response time distribution of the first job upper bounds the response time distribution of any other job of that task. Since we are considering synchronous tasks, calculating the response time distribution of the first job of a task provides the worst case response time distribution of the task and, implicitly, its worst case DMP.

Problem: In this paper, we address the problem of computing the response time distributions and, implicitly, Deadline Miss Probabilities of tasks with pMIT and pWCET. The response time of a job is the elapsed time between its release and its completion. Since we consider jobs with probabilistic parameters, the response time of a job is also described by a random variable. The DMP of a job is obtained by comparing the response time distribution of said job and its deadline, be it a probabilistic deadline or a deterministic one. This is a novel problem, and the fact that the system under consideration has more than one task parameter given as a distribution makes it a complex one. The solution that we describe is exponential in the number of tasks and the size of the random variables representing the task parameters. We describe techniques to decrease the analysis duration and to make it tractable. Note that in [26], the authors have shown that there can not be any pseudo-polynomial exact test for fixed-priority task-level scheduling of the non-cyclic GMF task model, or any more general model - that encompasses it - such as the probabilistic task model.

[5]In the analysis of GMF tasks this is known as the frame separation constraint.

IV. Response time analysis

In this section, we introduce an analysis computing the response time distribution of a given task. Since the system under consideration is a task-level fixed-priority preemptive one, then a given task is not influenced by tasks of lower priority but only by those of higher priority. Thus, we consider without loss of generality, the task of interest to be the lowest priority task, τ_n, in a set of n tasks.

Before we proceed to the response time analysis for tasks that have pWCET as well as pMIT, we first recall here the response time analysis for tasks that have only the WCET described by a random variable [5]. The response time $\mathcal{R}_{i,j}$ of a job $\tau_{i,j}$ that is released at time instant $\lambda_{i,j}$ is computed using the following equation:

$$\mathcal{R}_{i,j} = \mathcal{B}_i(\lambda_{i,j}) \otimes \mathcal{C}_i \otimes \mathcal{I}_i(\lambda_{i,j}), \tag{3}$$

where $\mathcal{B}_i(\lambda_{i,j})$ is the accumulated *backlog* of higher priority tasks released before $\lambda_{i,j}$ and still active (not completed yet) at time instant $\lambda_{i,j}$. $\mathcal{I}_i(\lambda_{i,j})$ is the sum of the execution times of higher priority tasks arriving after $\lambda_{i,j}$ and that could preempt the job under analysis, $\tau_{i,j}$. The operator $\otimes$ is the convolution between two random variables. In the case of synchronous tasks the backlog of the first job of τ_n is equal to $\mathcal{B}_n = \bigotimes_{i \in hp(n)} \mathcal{C}_i$.

Equation (3) is solved iteratively, integrating new possible preemptions by modifying the tail of the response time distribution $\mathcal{R}_{i,j}$ at each iteration. The iterations stop once either there are no more preemptions to be integrated or all the newly obtained values in the tail of the response time distribution are larger than the tasks' deadline.

Intuitively: In the case when the MIT is also given as a random variable, we need to modify Equation (3) to take into account the fact that a preemption can occur at different time instants with different probabilities. We do so by making a copy of $\mathcal{R}_{i,j}$ for each value in the pMIT distribution of the preempting task and scaling each copy with the probability of its respective value. We then modify the tail of each copy in order to integrate, as in Equation (3), the execution requirement of the preempting task. The distributions obtained are then coalesced and the process is repeated until either there are no more preemptions to be integrated or the newly obtained values in the tails of each copy of the response time distribution are larger than the tasks' deadline. Note that, if the MIT of the preempting task is deterministic, then the analysis is the same as Equation (3). Furthermore, our analysis can handle any combination of probabilistic and deterministic parameters, and in the case that all parameters are deterministic the returned result is the same as the one provided by the worst case response time analysis in [27].

We present first a numerical example of the analysis before introducing it formally.

Example 1. *We introduce here an example of a task system and the response time computation for the lowest priority task of the system. In order to better understand the implication of the probabilistic parameters, let us start with a deterministic task set and slowly move our way to a probabilistic version of it.*

Given a task system $\tau = \{\tau_1, \tau_2\}$ scheduled under task-level fixed-priority scheduling policy with τ_1 at higher priority and τ_2 at lower priority, and the tasks described by the following worst case values: $\tau_1 = (C_1 = 2, T_1 = 5)$ and $\tau_2 = (C_2 = 4, T_2 = 7)$, with $D_i = T_i, \forall i$. A deterministic analysis of this task system would conclude that it is unschedulable, since the response time of $\tau_{2,1}$ (the first job of τ_2) is greater than its deadline.

First generalization (pMIT): *Let us now assume that, after studying the system, we have extra information about its behaviour, namely that not all the jobs of τ_1 arrive with 5 units of time between them, but instead they follow a distribution equal to $\mathcal{T}_1 = \begin{pmatrix} 5 & 6 \\ 0.2 & 0.8 \end{pmatrix}$ meaning that an instance of τ_1 has a 20% probability of arriving 5 units of time after the previous instance, and a 80% probability of arriving 6 time units after the previous instance. All other parameters of the system keep their worst case representation.*

In this case, $\tau_{2,1}$ also has an 80% probability of finishing execution before its deadline, i.e. the cases when $\tau_{1,2}$ arrives at $t = 6$.

Second generalization (pWCET): *Going back to the deterministic task system that we started with this time let us assume that we have extra information about the execution requirement of τ_2, namely that it follows the distribution $\mathcal{C}_2 = \begin{pmatrix} 3 & 4 \\ 0.9 & 0.1 \end{pmatrix}$ meaning that only 10% of the jobs generated by τ_2 have an execution requirement of 4 time units and the other 90% require only 3 time units. All other parameters of the system are considered with their worst case values.*

In this case as well $\tau_{2,1}$ has a high probability, 90%, of finishing execution before its deadline, namely the cases when it requires 3 units of execution time.

Combining the two cases: *If we now combine both cases presented above, taking into consideration the probabilistic nature of $\mathcal{T}_1$ and of $\mathcal{C}_2$, we note that $\tau_{2,1}$ misses its deadline only when the two worst case scenarios happen at the same time, i.e. $\tau_{1,2}$ arrives at $t = 5$ and $\tau_{2,1}$ needs to execute for 4 units of time. The probability of this happening is the combined probability of the two scenarios, namely $\tau_{2,1}$ has a probability of $DMP_2 = 0.2 \times 0.1 = 0.02 = 2\%$ of missing its deadline.*

For this example we have found the deadline miss probability of $\tau_{2,1}$ by (manually) exploring all the possible combinations of inter-arrival times and execution times of the two tasks. This is not always possible to do, considering that a system can have many tasks and each parameter distribution may have tens, hundreds or even thousands of values, leading to a large number of possible combinations.

The analysis that we introduce computes the worst case response time distribution of a task by means of convolution of random variables, ensuring in this way that all scenarios have been taken into account without needing to explicitly investigate all of them.

The analytical response time computation: *The probabilistic representation of the system under analysis is $\tau = \{\tau_1 = (\mathcal{C}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \mathcal{T}_1 = \begin{pmatrix} 5 & 6 \\ 0.2 & 0.8 \end{pmatrix}), \tau_2 = (\mathcal{C}_2 = \begin{pmatrix} 3 & 4 \\ 0.9 & 0.1 \end{pmatrix}, \mathcal{T}_2 = \begin{pmatrix} 7 \\ 1 \end{pmatrix})\}$ and we are interested in finding the response time distribution $\mathcal{R}_{2,1}$ of $\tau_{2,1}$ by applying our analysis.*

The computation starts by initialising the response time distribution $\mathcal{R}_{2,1}$ with the combined execution time requirements of higher priority tasks, in this case $\mathcal{C}_1$, and adding to it the execution requirement of the task under analysis:

$$\mathcal{R}_{2,1} = \mathcal{C}_1 \otimes \mathcal{C}_2 = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 3 & 4 \\ 0.9 & 0.1 \end{pmatrix} = \begin{pmatrix} 5 & 6 \\ 0.9 & 0.1 \end{pmatrix}.$$

The possible preemption that can occur from $\tau_{1,2}$ can be either at $t = 5$ with probability 0.2 or at $t = 6$ with probability 0.8. For each of these two cases we make a copy of $\mathcal{R}_{2,1}$ and proceed in the following way:

$$\mathcal{R}^1_{2,1} = \left(\begin{pmatrix} 5 \\ 0.9 \end{pmatrix} \oplus \begin{pmatrix} 6 \\ 0.1 \end{pmatrix} \otimes \begin{pmatrix} 2 \\ 1 \end{pmatrix}\right) \otimes \begin{pmatrix} 0 \\ 0.2 \end{pmatrix} = \left(\begin{pmatrix} 5 \\ 0.9 \end{pmatrix} \oplus \begin{pmatrix} 8 \\ 0.1 \end{pmatrix}\right) \otimes \begin{pmatrix} 0 \\ 0.2 \end{pmatrix} = \begin{pmatrix} 5 & 8 \\ 0.9 & 0.1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 0.2 \end{pmatrix} = \begin{pmatrix} 5 & 8 \\ 0.18 & 0.02 \end{pmatrix}$$

represents the case that $\tau_{1,2}$ arrives at $t = 5$ preempting $\tau_{2,1}$. In this case, $\tau_{1,2}$ can only affect the tail of the distribution, i.e. $\tau_{2,1}$ did not finish execution by $t = 6$. Two units of time are added to the tail of the distribution, and the entire resulting distribution is updated with the probability 0.2 of $\tau_{1,2}$ arriving at $t = 5$.

$$\mathcal{R}^2_{2,1} = \begin{pmatrix} 5 & 6 \\ 0.9 & 0.1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 0.8 \end{pmatrix}$$

represents the case that $\tau_{1,2}$ arrives at $t = 6$ and so it does not preempt $\tau_{2,1}$. The tail of the distribution is not affected, but the entire distribution is updated with the probability 0.8 of $\tau_{1,2}$ arriving at $t = 6$.

Once the two copies of $\mathcal{R}_{2,1}$ have been obtained they are coalesced and the final result is obtained:

$$\mathcal{R}_{2,1} = \mathcal{R}^1_{2,1} \oplus \mathcal{R}^2_{2,1} = \begin{pmatrix} 5 & 8 \\ 0.18 & 0.02 \end{pmatrix} \oplus \begin{pmatrix} 5 & 6 \\ 0.72 & 0.08 \end{pmatrix} = \begin{pmatrix} 5 & 6 & 8 \\ 0.9 & 0.08 & 0.02 \end{pmatrix}.$$

The value 8 of the response time distribution is not possible since $\tau_{2,1}$ will not be allowed to continue its execution past $t = 7$. Any value strictly greater that the jobs deadline is replaced by "DMP" and their summed probability mass represents the Deadline Miss Probability of the job: $\mathcal{R}_{2,1} = \begin{pmatrix} 5 & 6 & DMP \\ 0.9 & 0.08 & 0.02 \end{pmatrix}$.

Since the earliest arrival of $\tau_{1,3}$ is at $t = 10$ then this job cannot preempt $\tau_{2,1}$ and the analysis stops here. We have obtained the worst case response time distribution of τ_2 and its Deadline Miss Probability, 0.02, which is exactly the same as the one obtained earlier by enumerating all possible scenarios.

Third generalization (probabilistic deadline): *In order to show the effect of a probabilistic deadline, we further generalize the task system by considering that the arrival distribution and hence the deadline distribution of τ_2 is equal to $\mathcal{T}_2 = \begin{pmatrix} 7 & 8 \\ 0.3 & 0.7 \end{pmatrix}$. In this case $\tau_{2,1}$ will miss its deadline if and only if the following worst case scenario happens: $\tau_{1,2}$ arrives at $t = 5$ (probability 0.2), $\tau_{2,1}$ executes for 4 units of time (probability 0.1) and $\tau_{2,2}$ arrives at $t = 7$ (probability 0.3). The probability of this scenario happening is $DMP_2 = 0.2 \times 0.1 \times 0.3 = 0.006$.*

This probability can be obtained directly by applying Equation (9):

$$\mathcal{B}_2 = \mathcal{R}_{2,1} \ominus \mathcal{D}_2 = \begin{pmatrix} 5 & 6 & 8 \\ 0.9 & 0.08 & 0.02 \end{pmatrix} \ominus \begin{pmatrix} 7 & 8 \\ 0.3 & 0.7 \end{pmatrix} = \begin{pmatrix} -3 & -2 & -1 & 0 & 1 \\ 0.63 & 0.83 & 0.24 & 0.014 & 0.006 \end{pmatrix}.$$

The value that are less or equal to zero are discarded, since they represent the cases when the job finishes execution before or at the deadline. The strictly positive values are kept and their added probabilities represent the tasks' DMP, in this case the probability of the value 1 which is 0.006 as found by the descriptive method above.

We introduce now a formal description of our analysis.

Formally: The arrival time of the j^{th} job of a task τ_n is computed for $j \geq 1$ as follows

$$\mathcal{T}_{n,j} = \mathcal{T}_n \otimes ... \otimes \mathcal{T}_n, \quad j - 1 \ times \tag{4}$$

and for $j = 0$ we have $\mathcal{T}_{n,0} = 0$.

The worst case response time of task τ_n is initialized as:

$$\mathcal{R}^0_n = \mathcal{B}_n \otimes \mathcal{C}_n \tag{5}$$

where the backlog at the arrival of τ_n is equal to

$$\mathcal{B}_n = \bigotimes_{i \in hp(n)} \mathcal{C}_i \tag{6}$$

After adding the execution time of the task under analysis to the backlog accumulated at its arrival, its response time is updated iteratively with the possible preemptions as follows:

$$\mathcal{R}^i_n = \bigoplus_{j=1}^{k} \mathcal{R}^{i,j}_n \tag{7}$$

where i is the current iteration, k is the number of values in the random variable representing the pMIT distribution of the preempting task, j is the current value taken into consideration from the pMIT distribution of the preempting task, and $\mathcal{R}^{i,j}_n$ is the j^{th} copy of the response time distribution and it integrates the possible preemption in the following way:

$$\mathcal{R}^{i,j}_n = (\mathcal{R}^{i-1,head}_n \oplus (\mathcal{R}^{i-1,tail}_n \otimes \mathcal{C}^{pr}_m)) \otimes \mathcal{P}_{pr} \tag{8}$$

where:

- n is the index of the task under analysis;
- i is the current step of the iteration;
- j represents the index of the current value taken into consideration from the pMIT distribution of the preempting task;
- $\mathcal{R}^{i-1,head}_n$ is the part of the distribution that is not affected by the current preemption under consideration;
- $\mathcal{R}^{i-1,tail}_n$ is the part of the distribution that may be affected by the current preemption under consideration;
- m is the index of the higher priority task that is currently taken into account as a preempting task;
- $\mathcal{C}^{pr}_m$ is the execution time distribution of the currently preempting task;

- $\mathcal{P}_{pr}$ is a fake random variable used to scale the j^{th} copy of the response time with the probability of the current value i from the pMIT distribution of the preempting task. This variable has one unique value equal to 0 and its associated probability is equal to the i^{th} probability in the pMIT distribution of the preempting job.

For each value $v^j_{m,i}$ in $\mathcal{T}_{(m,j)}$ for which there exists at least one value $v_{n,i}$ in $\mathcal{R}^{i-1}_n$ so that $v^i_n > v^j_{m,i}$, the distribution $\mathcal{R}^{i-1}_n$ is split in two parts:

- $\mathcal{R}^{i-1,head}_n$ which contains all values $v^-_{n,i}$ of $\mathcal{R}^{i-1}_n$ that are less or equal than $v^j_{m,i}$, i.e., $v^-_{n,i} \leq v^j_{m,i}$, and
- $\mathcal{R}^{i-1,tail}_n$ which contains all values $v^+_{n,i}$ of $\mathcal{R}^{i-1}_n$ that are greater than $v^j_{m,i}$, i.e., $v^+_{n,i} > v^j_{m,i}$.

The iterations end when there are no more arrival values $v^j_{m,i}$ of any job i of any higher priority task τ_m that is smaller than any value of the response time distribution at the current step. A stopping condition may be explicitly placed in order to stop the analysis after a desired response time accuracy has been reached. For example, the analysis can be terminated once an accuracy of 10^{-9} has been reached for the response time. In our case, the analysis stops when new arrivals of the preempting tasks are beyond the deadline of the task under analysis, i.e., the type of analysis required for systems where jobs are aborted once they reach their deadline.

Once the jobs' response time distribution can be computed, the Deadline Miss Probability can be obtained by comparing the response time distribution with that of the deadline, as follows:

$$\mathcal{B}_i = \mathcal{R}_i \ominus \mathcal{D}_i = \mathcal{R}_i \oplus (-\mathcal{D}_i), \tag{9}$$

where the $\ominus$ operator indicates that the values of the distribution are negated. We use the notation $\mathcal{B}_i$ even though the resulting distribution is not a backlog distribution in the strict sense for the model we consider, but it is still the formula for computing backlog for systems where jobs are allowed to execute past their deadline.

The DMP of the job under analysis is given by the probability mass corresponding to the values strictly greater than 0, i.e. the job would need more units of time to finish its execution. The probability mass corresponding to the values less or equal to 0 gives the probability that the job finishes execution before its deadline and the next release.

A. Critical instant of a task with multiple probabilistic parameters

Lemma 1. *We consider a task system of n tasks with τ_i described by deterministic C_i and probabilistic $\mathcal{T}_i, \forall i \in \{1, 2, \cdots, n\}$. The set is ordered according to the priorities of the tasks and the system is scheduled preemptively on a single processor. The response time distribution $\mathcal{R}_{i,1}$ of the first job of task τ_i is greater than the response time distribution $\mathcal{R}_{i,j}$ of any j^{th} job of task τ_i, $\forall i \in \{1, 2, \cdots, n\}$.*

Proof. The response time distribution $\mathcal{R}_{i,j}$ of a job within a probabilistic system may be obtained by composing response time values $R^k_{i,j}$ of jobs within all corresponding deterministic systems obtained by considering all values of the minimal inter-arrival times and the probability associated with the respective scenario k and we have $\begin{pmatrix} R^k_{i,j} \\ p_{scenario_k} \end{pmatrix}$. For each of these deterministic systems we know from [28] that the critical instant of a task occurs whenever the task is released simultaneously with its higher priority tasks. Thus we have that $R^k_{i,1} \geq R^k_{i,j}, \forall k, j > 1$ and we obtain $\mathcal{R}_{i,1} \succeq \mathcal{R}_{i,j}$ as the associated probabilities of $R^k_{i,1}$ and$R^k_{i,j}, \forall k$ are the same. □

Theorem 1. *We consider a task system of n tasks with τ_i described by probabilistic $\mathcal{C}_i$ and $\mathcal{T}_i, \forall i \in \{1, 2, \cdots, n\}$. The set is ordered according to the priorities of the tasks and the system is scheduled preemptively on a single processor. The response time distribution $\mathcal{R}_{i,1}$ of the first job of task τ_i is greater than the response time distribution $\mathcal{R}_{i,j}$ of any j^{th} job of task τ_i, $\forall i \in \{1, 2, \cdots, n\}$.*

Proof. The response time distribution $\mathcal{R}_{i,j}$ of a job within a probabilistic system is obtained by convolving response time distributions $\mathcal{R}^l_{i,j}$ of jobs within all corresponding probabilistic systems obtained by considering tasks described by $C_i, \mathcal{T}_i, \forall i$. Then within each scenario l we have from Lemma 1 that $\mathcal{R}^l_{i,1} \succeq \mathcal{R}^l_{i,j}$. We have then $\mathcal{R}_{i,1} = \otimes_{l=1}^{nb_of\,scenarios} \mathcal{R}^l_{i,1} \succeq \otimes_{l=1}^{nb_of\,scenarios} \mathcal{R}^l_{i,j} = \mathcal{R}_{i,j}$. □

V. Validation of the method

Our response time analysis provides safe, but pessimistic results with respect to an exact analysis that would provide the response time of any job of a task. We leave the reduction of pessimism as future work, for now we just note that a pessimistic result is safe, and since this is the first analysis framework of its kind, we do not strive for absolute accuracy, only for safeness. We validate our method by ensuring the following three conditions: limit condition, insurance of the worst case scenario and convergence (conditions originally presented in [29]).

Limit condition: To ensure the limit condition we need to prove that our method provides the same or larger than the worst case response time of a corresponding deterministic task system obtained by considering only the worst case values for all parameters of the tasks.

By applying the analysis to a deterministic task system, the convolution specific to random variables becomes the summation of deterministic values, the coalescing operator $\oplus$ that combines two random variables is no longer necessary since there is no $head$ and $tail$ sections of random variables, the splitting into $head$ and $tail$ is replaced with the corresponding inequality verification, checking if the next arrival of a higher priority task will preempt the job under execution. The analysis stops when there are no higher priority jobs left that can preempt the job under analysis.

Insurance of the worst case scenario: In order to obtain the worst case response time of the given probabilistic task system, the existing deterministic analysis [27] is applied to the deterministic task system obtained by considering from each task its minimum inter-arrival time value and its maximum worst case execution time value. Besides the worst case response time, the probabilistic analysis framework that we propose also provides the best case response time of the task, and all possible response time values between the best case and the worst case, each with its probability of occurring.

Convergence: In order to ensure the convergence condition we need to prove that the analysis of a task system does not

run indefinitely without, i.e., sooner or later returning a result. This condition follows from these two assumptions over the system model:

a) the distribution representing the tasks' parameters are of finite length and so the analysis will complete a loop of integrating new preemptions in the tail of the response time distribution in a finite time.

b) the system under consideration makes use of an *a*bort on deadline policy, which means that the analysis stops when the returned response time values are larger than the largest possible deadline of the task under analysis, i.e. the largest value in the pMIT distribution.

VI. Implementation and evaluation of the method

We implemented our response time analysis in MATLAB. The pseudo-code for the associated steps is presented in Algorithms 1 and 2 and the complete scripts are available[6].

Before we proceed with the description of the simulations performed we recall here the concept of re-sampling[7].

Definition 10. *[30] [**Re-sampling**] Let $\mathcal{X}_i$ be a distribution with n values describing a parameter of a task τ_i. The process of **re-sampling to** k **values** or k**-re-sampling** consists of reducing the initial distribution $\mathcal{X}_i$ from n values to a distribution $\mathcal{X}_i^*$ with k values.*

The re-sampling is safe with respect to the response time analysis as the response time $\mathcal{R}_i$ of any task τ_i of the initial system is greater than the response time $\mathcal{R}_i^*$ of the considered task within the re-sampled task system.

The re-sampling of a real-time distribution is performed in the following two sequential steps: *1*) selection of the k samples to be kept in the reduced distribution and *2*) redistribution of the probabilities from the values that are not kept.

Re-sampling for pWCET differs from re-sampling for pMIT in the second step, namely, as larger values of pWCET produce greater probabilistic response times as well as smaller values of pMIT produce greater probabilistic response times.

A. Experiment 1: Complexity

One may be concerned when mentioning probabilistic analyses by their complexity as operations like convolutions of random variables are involved. Our analysis is tested first with respect to the complexity.

In Figure 3, a 3D plot of the analysis duration is presented. On the z-axis the analysis duration is given in seconds, on the x-axis is the variation of the number of values per random variable, from 2 to 16 values, and on the y-axis is the number of tasks per task system, also from 2 to 16 tasks. Every point on the surface corresponds to the average analysis duration of 100 task sets. The worst case utilization of each considered task is between 1.5 and 2 and the expected utilization is between 0.5 and 1. The pWCETs are decreasing distributions while the pMITs are increasing distributions.

[6]The scripts are available at http://www.loria.fr/~maxim or uppon request to the authors

[7]Note that in statistics, re-sampling has a different meaning from that used in real-time systems. For an example of re-sampling in real-time systems see Appendix B or refer to [30].

Algorithm 1 Worst case response time distribution computation

Input: Γ a task set and $target$ the index of the task we analyze
Output: $\mathcal{R}_{target}$ the worst case response time distribution of τ_{target}

$\mathcal{R}_{target} = \mathcal{C}_{target}$; //initialize the response time with the execution time of the task under analysis
for $(i = 1; i < target; i++)$ **do**
  $\mathcal{R}_{target} = \mathcal{R}_{target} \otimes \mathcal{C}_i$; //add the execution times of all higher priority tasks
end for
for $(i = 1; i < target; i++)$ **do**
  $\mathcal{A}_i = \mathcal{T}_i$; //initialize the arrivals of each higher priority task with their inter-arrival times distribution
end for
for $(i = 1; i < max(\mathcal{T}_{target}); i++)$ **do**
  for $(j = 1; j < target; j++)$ **do**
    if $max(\mathcal{R}_{target}) > min(\mathcal{A}_j)$ and $min(\mathcal{A}_j) = i$ **then**
      $\mathcal{R}_{target} = doPreemption(\mathcal{R}_{target}, \mathcal{A}_j, \mathcal{C}_j)$; //update the response time with the current possible preemption
      $\mathcal{A}_j = \mathcal{A}_j \otimes \mathcal{T}_j$; //the next arrival of τ_j
    end if
  end for
end for
$\mathcal{R}_{target} = sort(\mathcal{R}_{target})$
Output: $\mathcal{R}_{target}$

Algorithm 2 doPreemption function

Input: $\mathcal{R}$ the current response time,
$\mathcal{A}$ the arrival distribution of the preempting job and
$\mathcal{C}$ the execution time distribution of the preempting job
Output: $\mathcal{R}$ the response time distribution updated with the current preemption

$\mathcal{R}_{intermediary}$ = *empty*;
$\mathcal{A}_{fake}$ = *empty*;
for $(i = 1; i < length(\mathcal{A}); i++)$ **do**
  //constructing the fake random variable giving the probability of the preemption occurring
  $\mathcal{A}_{fake}$.*value* = 0; //the value of the fake random variable
  $\mathcal{A}_{fake}$.*probability* = $\mathcal{A}$(i).*probability*; //the probability of the fake random variable
  Split $\mathcal{R}$ into *head* and *tail* according to the preemption value;
  if *tail* != *empty* **then**
    tail = *tail* $\otimes \mathcal{C}$;
  end if
  $\mathcal{R}_{intermediary}$ = *head* $\oplus$ *tail*;
  $\mathcal{R}_{intermediary} = \mathcal{R}_{intermediary} \otimes \mathcal{A}_{fake}$;
  $\mathcal{R}$ = $\mathcal{R}$ + $\mathcal{R}_{intermediary}$;
  $\mathcal{R} = sort(\mathcal{R})$
end for
$\mathcal{R}$ = sort($\mathcal{R}_{intermediary}$)
Output: $\mathcal{R}$

We note that the analysis duration of a task set with 16 tasks, each of its random variables having 16 values, takes in average 140 seconds, i.e. the highest point on the z-axis.

The analysis duration increases both with respect to the number of tasks per task system and with respect to the number

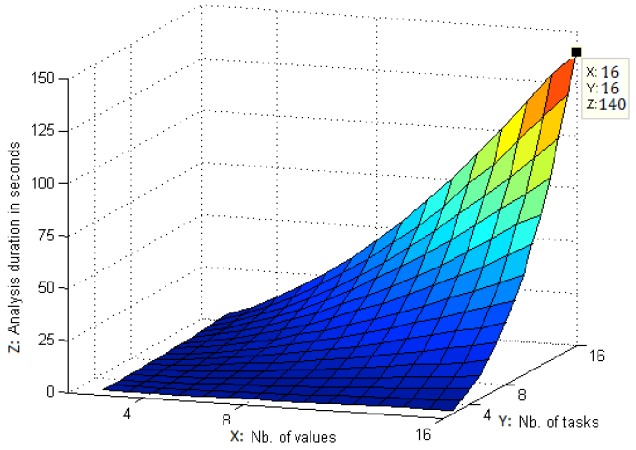


Fig. 3: Analysis duration of random task system

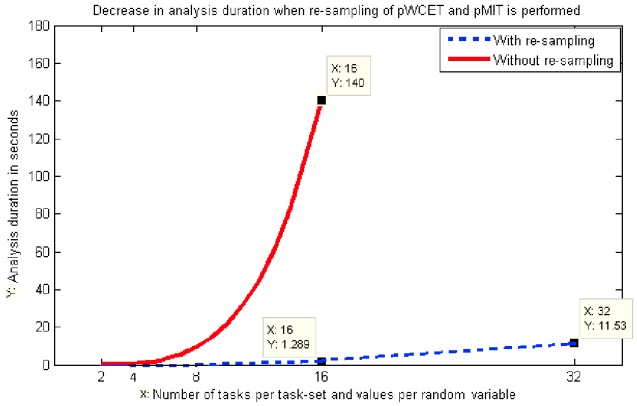


Fig. 4: Analysis duration of random task systems , increasing both the number of tasks per task system and the number of values per random variable in the same time

of values per random variable, indicating the exponential complexity of the analysis. Nevertheless, solutions exist to make such analysis affordable even for large task systems with parameters described by large random variables.

A solution to decreasing the probabilistic analysis duration is re-sampling, which reduces the analysis duration while introducing minimal pessimism [30].

In Figure 4, the diagonal of the surface from Figure 3 is represented by a solid line, having an exponential behaviour. The same analysis is performed with re-sampling of the pWCET to 50 values and of the pMIT to 5 values both done after each iteration. The improvement is shown in the same figure, represented by the dotted line; this time the average analysis duration over 100 task sets each having 16 task with 16 values per random variable is only 1.29 seconds, compared to 140 seconds when no re-sampling was performed. We note the important gain in speed when the analysis is performed with re-sampling, even for systems that have 32 tasks and each random variable has 32 values it takes 11 seconds to perform the analysis, indicating that it is affordable even for considerable larger systems. We show in the next experiment that distributions with 5 values for pMIT bring significant increase in precision with respect to the worst case response time analysis.

B. Experiment 2: Improvement with respect to existing analysis

The second set of experiments that we performed show the precision that is gained by having tasks' parameters given as random variables. In order to do so we randomly generated probabilistic task systems to which we applied our analysis with different levels of re-sampling applied either at pWCET level or at pMIT level.

1) Precision gained by having a more detailed pMIT distribution: To show the increase in precision brought by a more detailed pMIT distribution, we repeated three times the analysis on the generated task system, each time varying the re-sampling level of the pMIT, using 10 values, 5 values and 1 value, respectively. A pMIT distribution with only one value is in fact a worst case MIT. The pWCET distribution was not re-sampled.

Figure 5 shows the task Deadline Miss Probability averaged over 100 task systems of 10 tasks. We note that the deterministic reasoning that considers one value for the MIT (and no re-sampling for pWCET) provides a DMP equal to 0.0167 (left bar in Figure 5) - this is the case of the existing analysis presented in [5]. By considering a probabilistic reasoning with 10 values for the pMIT we decrease by a factor of 3 the DMP. Then further increasing the number of values within the pMIT only marginally decreases the DMP and this is shown by comparing the values of DMP for 1, 5 and 10.

Nevertheless, having a 10-value pMIT does not bring much increased precision over a 5-value pMIT, as can also be seen in Figure 5 where their respective DMPs are almost equal. We note that it is not necessary to have a large pMIT distribution to have a precise analysis, depending on the system under analysis just 5 values can be sufficient.

This increase in precision comes at a cost, namely an increase in the analysis duration: the larger the distribution, the more time it takes to perform the analysis. In Figure 6 we show the analysis duration of the three cases described above. The duration of the 10 values analysis is close to that of no-re-sampling, where the duration of 5 values analysis is decreased. In this case the 5 values analysis seems to be a comfortable compromise between the duration and the gained DMP.

2) Precision gained by having a more detailed pWCET distribution: We performed a set of experiments to show the difference in precision when the tasks' pWCET distribution has 1000 values, 100 values, 10 values or only 1 value. A pWCET distribution with only one value is a deterministic WCET. The analysis was performed on the randomly generated task systems on which there were applied, in turn, different levels of re-sampling to the pWCET distribution. The pMIT distribution was not re-sampled.

In Figure 7 the difference in DMP between the four cases is depicted. Note that having 1 or 10 values in the pWCET distribution returns a task DMP equal to 1 which means that the system would be deemed unfeasible. This is not necessarily true, as can be seen from the bar representing the case when the pWCET distribution has 100 values. In this case the average DMP values of the analysed tasks does not surpass 0.02 which

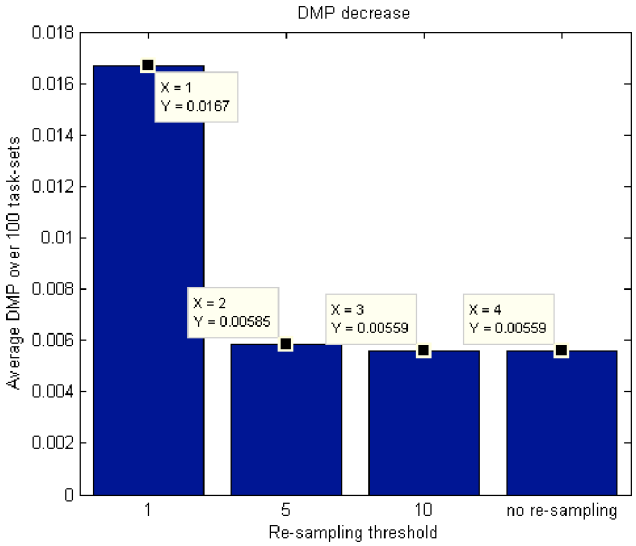


Fig. 5: The difference in DMP when the tasks' pMIT distribution has 1, 5 and respectively 10 values. Here 1 value indicates that only the worst case value of the pMIT distribution is considered.

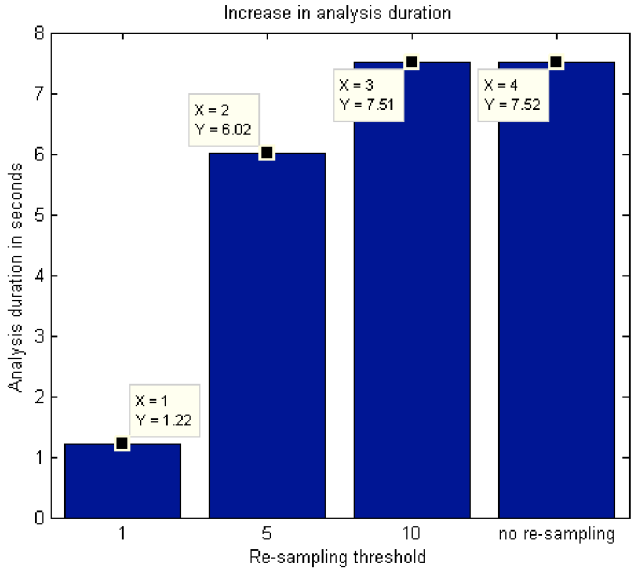


Fig. 6: The difference in seconds of the analysis duration when the tasks' pMIT distribution has 10, 5 and respectively 1 value, i.e. only the worst case value of the pMIT distribution.

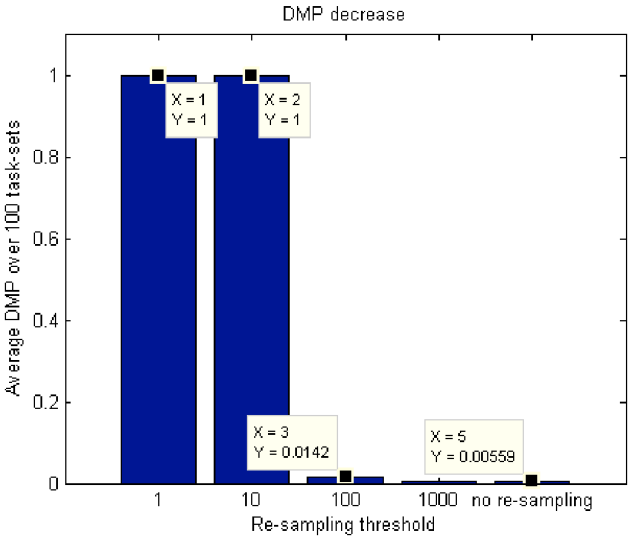


Fig. 7: The difference in DMP when the tasks' pWCET distribution has 1000, 100, 10 and respectively 1 value, i.e. only the worst case value of the pMIT distribution.

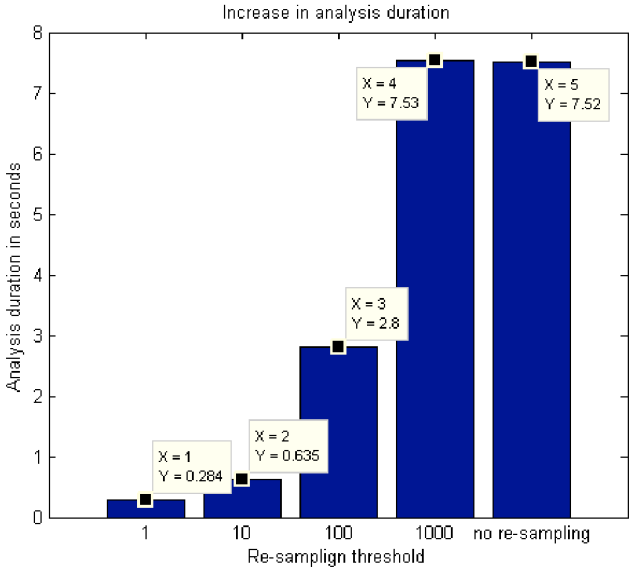


Fig. 8: The difference in seconds of the analysis duration when the tasks' pWCET distribution has 1000, 100, 10 and respectively 1 value, i.e. only the worst case value of the pMIT distribution.

means that the systems could be feasible if they can afford a 0.02 Deadline Miss Probability for their lowest priority tasks. In conclusion we decreased the DMP fifty times by analysing the system with 100 values for the pWCET.

As in the case of re-sampling at pMIT level, the re-sampling at pWCET level also comes with an increase in the analysis duration. Figure 8 depicts the analysis duration of the four cases described above, with the 1000 values for the pWCET having an exponential behavior but also being the distribution that has the most precision at DMP level. For pWCET re-sampling 100 is a compromise level that allows to obtain affordable duration and important increase in DMP.

In [30] a study is performed on different re-sampling strategies and novel re-sampling strategies are proposed that introduce very little pessimism. Also, by combining pWCET re-sampling and pMIT re-sampling, the analysis duration can be decreased considerably while retaining a high level of

accuracy, regardless of the system under analysis.

VII. Conclusion

In this paper, we introduced for the first time a probabilistic worst case response time analysis for tasks with multiple probabilistic parameters scheduled on one single processor under a task-level fixed-priority preemptive scheduling policy. This model of tasks generalizes the sporadic task model.

Our worst case response time analysis provides a probabilistic worst case response time that is described by a probability distribution. Our analysis is safe as we prove that the critical instant of a task occurs whenever the task is released simultanously with its higher priority tasks.

We provide experimental evaluation for our analysis and we compare it against existing results. We show that for a affordable duration analysis we decrease the Deadline Miss Probability by important factor.

This research provides a basis for the proposing of optimal fixed-priority scheduling algorithms for system with probabilistic worst case execution times and probabilistic minimum inter-arrival times.

Acknowledgments

The authors thank Robert Davis (University of York) for helping on an earlier version of this paper. The authors also thank the anonymous reviewers for their useful comments.

References

[1] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzeti, E. Quinones, and F. Cazorla, "Measurement-based probabilistic timing analysis for multi-path programs," in *the 24th Euromicro Conference on Real-Time Systems (ECRTS)*, pp. 91–101, 2012.

[2] S. Edgar and A. Burns, "Statistical analysis of WCET for scheduling," in *the 22nd IEEE International Real-Time Systems Symposium (RTSS)*, pp. 215–224, 2001.

[3] A. 653, "An avionics standard for safe, partitioned systels," in *Wind River Systems/IEEE Seminar*, 2008.

[4] D. Buttle, "Real-time in the prime-time," in *Keynote talk at the 24th Euromicro Conference on Real-Time Systems (ECRTS)*, 2012.

[5] J. L. Díaz, D. F. García, K. Kim, C.-G. Lee, L. Lo Bello, J. M. López, S. L. Min, and O. Mirabella, "Stochastic analysis of periodic real-time systems," in *the 23rd IEEE Real-Time Systems Symposium (RTSS)*, pp. 289–300, 2002.

[6] M. K. Gardner and J. W. Liu, "Analyzing stochastic fixed-priority real-time systems," in *5th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 44–58, Springer, 1999.

[7] A. Atlas and A. Bestavros, "Statistical rate monotonic scheduling," in *the 19th IEEE Real-Time Systems Symposium (RTSS)*, pp. 123–132, 1998.

[8] T.-S. Tia, Z. Deng, M. Shankar, M. Storch, J. Sun, L.-C. Wu, and J.-S. Liu, "Probabilistic performance guarantee for real-time tasks with varying computation times," in *IEEE Real-Time and Embedded Technology and Applications Symposium (ETFA)*, pp. 164–173, 1995.

[9] D. Maxim, O. Buffet, L. Santinelli, L. Cucu-Grosjean, and R. I. Davis, "Optimal priority assignments for probabilistic real-time systems," in *the 19th International Conference on Real-Time and Network Systems (RTNS)*, pp. 129–138, 2011.

[10] L. Cucu-Grosjean, "Independence - a misunderstood property of and for (probabilistic) real-time systems," in *"Real-Time Systems: the past, the present, and the future" conference organized in celebration of Professor Alan Burns sixtieth birthday*, March 14th, 2013.

[11] L. Abeni and G. Buttazzo, "QoS guarantee using probabilistic deadlines," in *the 11th Euromicro Conference on Real-Time Systems (ECRTS)*, pp. 242–249, 1999.

[12] L. Cucu and E. Tovar, "A framework for the response time analysis of fixed-priority tasks with stochastic inter-arrival times," *SIGBED Review*, vol. 3, no. 1, pp. 7–12, 2006.

[13] G. A. Kaczynski, L. Lo Bello, and T. Nolte, "Deriving exact stochastic response times of periodic tasks in hybrid priority-driven soft real-time systems," in *the 12th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 101–110, 2007.

[14] J. P. Lehoczky, "Real-time queueing theory," in *the 17th of the IEEE Real-Time Systems Symposium (RTSS)*, pp. 186–195, 1996.

[15] L. Abeni, N. Manica, and L. Palopoli, "Efficient and robust probabilistic guarantees for real-time tasks," *Journal of Systems and Software*, vol. 85, no. 5, pp. 1147–1156, 2012.

[16] N. T. Moyo, E. Nicollet, F. Lafaye, and C. Moy, "On schedulability analysis of non-cyclic generalized multiframe tasks," in *the 22nd Euromicro Conference on Real-Time Systems (ECRTS)*, pp. 271–278, 2010.

[17] S. Baruah, D. Chen, S. Gorinsky, and A. Mok, "Generalized multiframe tasks," *Real-Time Systems*, vol. 17, no. 1, pp. 5–22, 1999.

[18] V. Berten and J. Goossens, "Sufficient ftp schedulability test for the non-cyclic generalized multiframe task model," in *Nathan Fisher, editor, WiP Session of the 32nd IEEE Real-Time Systems Symposium (RTSS)*, 2011.

[19] J. Hansen, S. A. Hissam, and G. A. Moreno, "Statistical-based WCET estimation and validation," in *9th International Workshop on Worst-Case Execution Time (WCET) Analysis*, 2009.

[20] F. Dewan and N. Fisher, "Efficient admission control for enforcing arbitrary real-time demand-curve interfaces," in *the 33rd IEEE Real-Time Systems Symposium (RTSS)*, pp. 127–136, 2012.

[21] M. Neukirchner, T. Michaels, P. Axer, S. Quinton, and R. Ernst, "Monitoring arbitrary activation patterns in real-time systems," in *the 33rd IEEE Real-Time Systems Symposium (RTSS)*, pp. 293–302, 2012.

[22] C. Maxim, A. Gogonel, D. Maxim, and L. Cucu-Grosjean, "Estimation of probabilistic minimum inter-arrival times using extreme value theory," in *Junior Researcher Workshop on Real-Time Computing (JRWRTC) in conjunction with the 20th International Conference on Real-Time and Network Systems (RTNS)*, 2012.

[23] J. M. López, J. L. Díaz, J. Entrialgo, and D. García, "Stochastic analysis of real-time systems under preemptive priority-driven scheduling," *Real-Time Systems*, vol. 40, no. 2, pp. 180–207, 2008.

[24] A. K.-L. Mok, *Fundamental Design Problems of Distributed Systems for the Hard-Real-Time Environment*. PhD thesis, Laboratory for Computer Science, Massachusetts Institute of Technology, 1983.

[25] L. Palopoli, D. Fontanelli, N. Manica, and L. Abeni, "An analytical bound for probabilistic deadlines," in *Real-Time Systems (ECRTS), 2012 24th Euromicro Conference on*, pp. 179–188, 2012.

[26] M. Stigge and W. Yi, "Hardness results for static priority real-time scheduling," in *the 24th Euromicro Conference on Real-Time Systems (ECRTS)*, pp. 189–198, 2012.

[27] M. Joseph and P. K. Pandya, "Finding response times in a real-time system," *The Computer Journal*, vol. 29, no. 5, pp. 390–395, 1986.

[28] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM (JACM)*, vol. 20, no. 1, pp. 46–61, 1973.

[29] L. Cucu-Grosjean, "Probabilistic real-time schedulability analysis: from uniprocessor to multiprocessor when the execution times are uncertain," *RR-INRIA*, 2009.

[30] D. Maxim, M. Houston, L. Santinelli, G. Bernat, R. I. Davis, and L. Cucu-Grosjean, "Re-sampling for statistical timing analysis of real-time systems," in *the 20th International Conference on Real-Time and Network Systems*, pp. 111–120, ACM, 2012.

[31] I. Broster and A. Burns, "Applying random arrival models to fixed priority analysis," in *the WiP session of the 25th IEEE Real-Time Systems Symposium (RTSS)*, 2004.

APPENDIX A
DIFFERENCES BETWEEN TWO MODELS OF PROBABILISTIC ARRIVALS

In this section, we present the differences between two models of probabilistic arrivals that co-exist in the real-time literature: one associating probabilities to the possible values of the inter-arrival times and a second model associating probabilities to the number of arrivals within a time interval. We recall the first model in Section A-A and in Section A-B we present the second model. In Section A-C we provide a comparison of the two models.

A. Real-time systems with probabilistic MIT

Our paper uses the first model, that has been introduced in papers like [11], [14].

Within this model, for a task τ_i the pMIT $\mathcal{T}_i$ is defined by a distribution as follows:

$$\mathcal{T}_i = \begin{pmatrix} T^0 = T^{min} & T^1 & \cdots & T^k = T^{max} \\ f_{\mathcal{T}_i}(T^{min}) & f_{\mathcal{T}_i}(T^1) & \cdots & f_{\mathcal{T}_i}(T^{max}) \end{pmatrix}$$

For instance τ_1 has a pMIT $\mathcal{T}_1 = \begin{pmatrix} 5 & 10 \\ 0.3 & 0.7 \end{pmatrix}$ indicating that the MIT of τ_1 is equal to 5 with a probability of 0.3 and to 10 with a probability of 0.7 .

B. Real-time systems with probabilistic number of arrivals

This model has been used in papers like [13], [31].

Within this model, for a task τ_i^* the number of possible arrivals $\mathcal{N}_i$ within a time interval of length t_Δ is defined by a distribution as follows:

$$\mathcal{N}_i = \begin{pmatrix} N^0 = N^{min} & N^1 & \cdots & N^k = N^{max} \\ f_{\mathcal{N}_i}(N^{min}) & f_{\mathcal{N}_i}(N^1) & \cdots & f_{\mathcal{N}_i}(N^{max}) \end{pmatrix}$$

For instance if $\mathcal{N}_1 = \begin{pmatrix} 1 & 2 & 4 \\ 0.4 & 0.3 & 0.3 \end{pmatrix}$ for $t_\Delta = 12$, then the task τ_1^* has at most 4 arrivals from $t = 0$ to $t = 12$.

C. Comparing the two models

We present here the main difference between the two models from Section A-A and Section A-B.

We consider here the tasks defined in Sections A-B and A-A. For those tasks only the parameters related to the arrival of the tasks are relevant to our discussion.

The first model provides information to a schedulability analysis, information that the second model does not provide

- **Probabilistic MIT**: The task τ_1 has at most two arrivals before $t = 7$ (with a probability 0.3).
- **Probabilistic number of arrivals**: It is not possible to estimate how many times τ_1^* was released from 0 to 7. Different situations are possible like those described in Figure 9.

The first model can also provide the information that the second model provides to a schedulability analysis

- **Probabilistic MIT**: From $t = 0$ to $t = 12$ there are three scenarios of arrivals for task τ_1:
  - 3 arrivals at $t = 0$, $t = 5$ and $t = 10$ with a probability of 0.21;
  - 2 arrivals at $t = 0$ and $t = 5$ with a probability of 0.09;
  - 2 arrivals at $t = 0$ and $t = 10$ with a probability of 0.7.

  Thus, from $t = 0$ to $t = 12$ the possible number of arrivals of τ_1 is described by $\begin{pmatrix} 2 & 3 \\ 0.79 & 0.21 \end{pmatrix}$.
- **Probabilistic number of arrivals**: from $t = 0$ to $t = 12$ the number of arrivals of τ_1^* $\mathcal{N}_1 = \begin{pmatrix} 1 & 2 & 4 \\ 0.4 & 0.3 & 0.3 \end{pmatrix}$ is provided by the model.

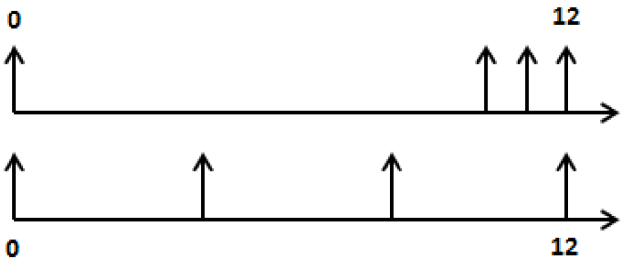


Fig. 9: The arrivals defined using the number of arrivals may correspond to these situations

APPENDIX B
RE-SAMPLING

We present here a summary of the notion of *real-time re-sampling of execution time distributions* that is used in Section V in order to better understand the mechanism. We note that re-sampling of minimum inter-arrival time distribution is similar to re-sampling of pWCET distributions with the exception that probability mass is transferred from larger values to smaller values.

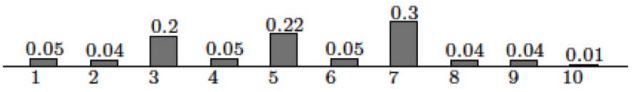
Definition 11 (Re-sampling of pWCET distributions). *Let $\mathcal{C}_i$ be a distribution with n values representing the probabilistic execution times of a task τ_i. The process of **re-sampling to k values** or **k-re-sampling** consists of reducing the initial distribution $\mathcal{C}_i$ from n values to k values.*

A re-sampling technique has two sequential steps, namely:

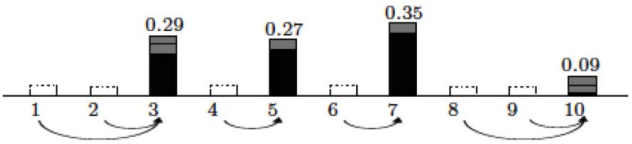
1) Selecting the k samples to be kept in the reduced distribution. The number of samples to be kept, k, comes from a trade-off of complexity versus accuracy that we want to achieve. A constraint while choosing the k samples to be kept is that the largest value has to remain in the re-sampled distribution in order to ensure the real-time analysis.

2) Re-distribution of the probabilities from the values that are not kept. In order to have a pessimistic re-sampled distribution, and hence a safe probabilistic analysis, there are no alternatives other than accumulating *from-left-to-right*. This means that the probabilities of a subset of samples have to be added to the probability of a larger value.

Example 2. *To better understand how re-sampling works we present an example. We consider a task with execution time given by a random variable with 10 values. Without loss of*



(a) Original distribution C



(b) Re-sampled distribution C'

Fig. 10: Re-sampling a probabilistic worst case execution time distribution from 10 values to 4 values.

generality we consider these ten values as the integer numbers from 1 *to* 10*. The probabilities of these values are as follows:*

$$\mathcal{C} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 0.05 & 0.04 & 0.2 & 0.05 & 0.22 & 0.05 & 0.3 & 0.04 & 0.04 & 0.01 \end{pmatrix},$$

as depicted in Figure 10a.

The objective is to reduce this random variable to only $k = 4$ *values instead of* 10 *in order to ease the convolution and hence the real-time analysis.*

For example, if the values $3, 5, 7$ *and* 10 *are kept, then the re-sampled distribution will be* $\mathcal{C}' = \begin{pmatrix} 3 & 5 & 7 & 10 \\ 0.29 & 0.27 & 0.35 & 0.09 \end{pmatrix}$ *which is represented in Figure 10b.*