Re-Sampling for Statistical Timing Analysis of Real-Time Systems

Dorin Maxim, Michael Houston, Luca Santinelli, Guillem Bernat, Robert Davis, Liliana Cucu

Re-Sampling for Statistical Timing Analysis of Real-Time Systems

Dorin Maxim
INRIA Nancy-Grand Est
Nancy, France
dorin.maxim@inria.fr

Mike Houston *
Stashmetrics Ltd.
Leeds, United Kingdom
mike@stashmetrics.com

Luca Santinelli *
ONERA
Toulouse, France
luca.santinelli@onera.fr

Guillem Bernat
Rapita Systems
York, United Kingdom
bernat@rapitasystems.com

Robert I. Davis
University of York
York, United Kingdom
rob.davis@york.ac.uk

Liliana Cucu-Grosjean
INRIA Nancy-Grand Est
Nancy, France
liliana.cucu@inria.fr

ABSTRACT

Guaranteeing timing constraints is the main purpose of analyses for real-time systems. The satisfaction of these constraints may be verified with probabilistic methods (relying on statistical estimations of certain task parameters) offering both hard and soft guarantees. In this paper, we address the problem of sampling applied to the distributions of worst-case execution times. The pessimism of presented sampling techniques is then evaluated at the level of response times.

Categories and Subject Descriptors

C.3 [**Special-purpose and Application-based Systems**]: Real-time and embedded systems; G.3 [**Probability and Statistics**]: Distribution functions; J.7 [**Computers in other Systems**]: Real time

General Terms

Real Time, Probability Distribution

Keywords

Probabilistic Real Time, Real Time Sampling, Pessimism, Probabilistic Execution Time, Re-sampling

1. INTRODUCTION

Nowadays Critical Real-Time Embedded Systems (CRTESs) are prevalent in many sectors and it is estimated that 98% of current processors are embedded. The performances of CRTESs are analyzed not only from the point of view of their correctness, but also from the perspective of time. Moreover, the increasing need for new functionalities imposes the use of modern and complex architectures. These architectures have a direct impact on the time variability of programs and applications exploiting functional and non-functional behavior of the CRTESs.

The timing analysis of such systems has been extensively studied by considering deterministic approaches based on worst-case scenarios that induce a certain pessimism. Unfortunately not all real-time systems can afford this pessimism and the consequent over-provisioning, and for these cases other approaches should be considered. Alternative approaches could be statistical and probabilistic, agent learning, game theory, etc. In this paper we are interested in the statistical and probabilistic approaches. These approaches offer an accurate and enriched representation of the parameters of the CRTESs. For example, a task parameter can be described as a random variable or the function expressing the resource given to a task flow can be modeled in terms of probabilistic bounds.

1.1 Related Work

Probabilistic real-time systems and probabilistic real-time analysis is becoming a common practice in the real-time community, [1]. Papers related to this topic used different terms like *stochastic analyses* [2–4], *probabilistic analyses* [5] or *statistical analysis* [6] to indicate usually that the considered CRTES has at least one parameter defined by a random variable. In this paper we make use of the word *statistic* to indicate that the work is based on the theory of statistics and the word of *probabilistic* to indicate that the work is based on the theory of probability.

In this paper we are interested in the distributions of worst-case execution times (WCETs). We consider these distributions as given and obtaining them is beyond the scope of this paper.

The statistical timing analyses can provide empirical distributions of WCETs from (often) a large mass of data [7, 8]. These distributions are difficult to use because of the impact that the large number of possible values for the WCET has on the complexity of the response time analysis [9].

Moreover the distributions of WCETs could also be obtained using different statistical techniques like extreme values [10–13]. These techniques introduce errors, thus different levels of confidence and pessimism are imposed on the CRTESs designer. We use here the concept of pessimism with the meaning defined in [14].

*The results where obtained while Michael Houston was with Rapita Systems and Luca Santinelli was with INRIA Nancy-Grand Est.

RTNS'12 , November 08 - 09 2012, Pont a Mousson, France

In order to avoid the problems cited before, re-sampling techniques [15] could be applied to the distributions of the WCETs. These techniques should be applied such that the real-time constraints are met and no relevant information is lost. To our best knowledge [16] proposes the unique existing re-sampling technique that decreases the number of possible values for the WCETs.

Contribution of the paper. The main contribution of our paper is a framework for systematic analysis of re-sampling mechanisms for real-time systems. Since we show that there is no absolute optimal re-sampling method, we prove that, according to the purposes and the requirements, some re-sampling methods behave better than others. Moreover, we propose three re-sampling methods that simplify the WCET distributions in order to ease timing analyses of real-time systems. They offer three different perspectives for the analysis and they have different effects on the pessimism carried by the analysis itself.

Organization of the paper. The paper is organized as follows. In Section 2 we motivate the need for re-sampling techniques in reducing the complexity of the real-time analysis. In Section 3 and Section 4 we define the notion of pessimism, and we propose three re-sampling techniques. Sections 5 and 6 outline the effect of re-sampling on the timing analysis in terms of improvements in the processing time required to perform a series of convolutions, thus making the problem tractable, and also in terms of the pessimism introduced by the re-sampling as the changes in the response time or the deadline miss ratio. In Section 7 we present a further implementation level improvement. We conclude and present future work in the last section of this paper.

2. MOTIVATIONS: A PROBABILISTIC MODEL

In this paper we deal with the preemptive fixed-priority scheduling of asynchronous periodic tasks with probabilistic execution times on one processor. We consider $\Gamma = \{\tau_1, \tau_2, \cdots, \tau_n\}$ a set of n periodic tasks ordered according to their priorities, $priority(\tau_i) \geq priority(\tau_j)$ if $i \leq j$. Each task is characterized by an offset O_i, an exact inter-arrival time T_i and a relative deadline D_i. This means that the j-th job of τ_i is released at time instant $O_i+(j-1)T_i$ and it must finish its execution by time instant $O_i+(j-1)T_i+D_i$.

A *probabilistic real-time system* is a real-time system with at least one parameter defined by a random variable. In this paper we consider the worst-case execution times to be described by random variables. This is an enriched model compared to having one possible value for the WCET. We denote by $\mathcal{C}_i$[1] the random variable describing the WCETs of a task τ_i (see Equation (1)). The Probability Function PF $\mathcal{C}_i$ is represented as

$$\mathcal{C}_i = \begin{pmatrix} C_i^j \\ \mathbb{P}\{\mathcal{C}_i = C_i^j\} \end{pmatrix}, \quad (1)$$

where $C_i^j \in [C_i^{min}, C_i^{max}]$ and $j \in [0, k_i]$ with $k_i \in \mathbb{N}^*$ the number of values that the random variable $\mathcal{C}_i$ has. We consider that C_i^{min}, C_i^{max} are given, thus known.

It is assumed that the random variables for the worst-case execution time of tasks are independent from one another; this hypothesis allows us to convolve random variables. This hypothesis is not that restrictive and current studies indicate that randomized architectures may fulfill them [17].

We note that C_i^{max} is equal to the value of the WCET that a deterministic worst-case analysis would consider. Thus, the probabilistic representation offers the possibility to ensure absolute guarantees as well as other intermediate levels of constraints.

In conclusion in this paper a probabilistic task τ_i is denoted by $(O_i, \mathcal{C}_i, T_i, D_i)$.

2.1 Probabilistic Real-Time Analysis

In [4, 18] the authors provide the calculation of the probabilistic response time of a job under a preemptive uniprocessor fixed-priority scheduling policy as given by Equation (2).

$$\mathcal{R}_{i,j} = \mathcal{W}_{i,j}(\lambda_{i,j}) \otimes \mathcal{C}_i \otimes \mathcal{I}_i, \quad (2)$$

where all the variables are random variables and the release time $\lambda_{i,j}$ of the job $\tau_{i,j}$ is deterministic. Here $\mathcal{W}_{i,j}(\lambda_{i,j})$ is the backlog at time $\lambda_{i,j}$ obtained as the workload of higher priority jobs than $\tau_{i,j}$ that have not yet been executed immediately prior to $\lambda_{i,j}$. Equation (2) is solved iteratively in [18]. $\mathcal{C}_i$ is the execution time of job $\tau_{i,j}$ and $\mathcal{I}_{i,j}$ is the interference in $\tau_{i,j}$ of all higher priority jobs than $\tau_{i,j}$, $hp(i)$, released at or after $\tau_{i,j}$, $\mathcal{I}_i = \sum_{\tau_{k,l} \in hp(i)} \mathcal{C}_k$.

The analysis takes as input the distributions $\mathcal{C}_i$ of all tasks and computes the response time (as a distribution of values and probabilities) by applying the convolution operator over PFs, here denoted as $\otimes$. The complexity of the response time computation is related to the number of possible values of $\mathcal{C}_i$. For instance, for two discrete random variables $\mathcal{C}_i$ and $\mathcal{C}_j$, during each iteration of the timing analysis we obtain, for any convolution, a new random variable that may have, in the worst-case, $k_i * k_j$ values, where k_i and k_j are the respective numbers of values of the two distributions. This implies that for any convolution $k_i * k_j$ operations are done and it makes the real-time analysis intractable except for simple scenarios, which is not always the case for real-time applications. Indeed, in complex applications the worst-case execution time distributions may have thousands of values [7].

EXAMPLE 2.1. *In this example we investigate the computational costs of the response time analysis provided by Equation (2).*

We used a task-set with 10 tasks, each task having a randomly generated PF *characterizing its worst-case execution time. For this simple experiment, we assumed that the tasks were executed non-pre-emptively and that the periods were long in relation to the response time. Hence the problem simply involved the convolution of the* PF*s of the 10 tasks. The* PF*s contained just* 100 *values spread over a range of approximately 10,000 possible values, intended to simulate execution times measured to an accuracy of 0.1 microseconds, with a largest value of around 1 millisecond (typical of embedded real-time systems).*

Figure 1 shows how the computation time required to compute the response time increases with each additional convolution. Even for this simple example the computational cost becomes prohibitive after about 10 convolutions.

The richness that the probabilistic real-time model offers to the analysis [19] is limited by the complexity of the analysis itself. The complexity problem has a possible solution via a reduction in the number k_i of possible values of $\mathcal{C}_i$ (samples) as long as the real-time constraints are met.

Hence, in this paper we investigate the sampling problems in real-time scenarios, in particular the *simplification* of distributions of worst-case execution time and we give an answer to the question: *given a distribution with n values, how do we select k significant values out of n so that the analysis is still 'accurate'?*

From the original WCET distribution, we derive another distribution of worst-case execution times which offers the same real-time guarantees although with a reduced amount of information

[1] in this paper we utilise calligraphic letters to denote random variables

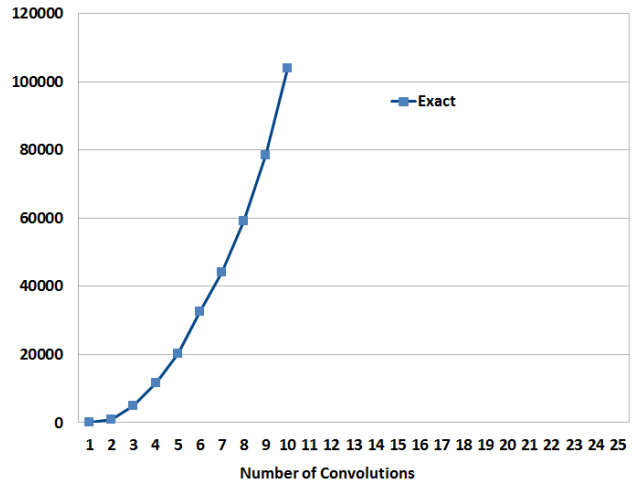


Figure 1: Computation time relative to the number of convolutions

(simplified distribution). Furthermore, we build the basis for an analysis of simplification mechanisms in order to provide a way of judging them in terms of their accuracy.

3. RE-SAMPLING: SIMPLIFYING DISTRIBUTIONS

DEFINITION 3.1 (**RE-SAMPLING**). *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, while not being optimistic.*

First re-sampling techniques have been presented in [16, 18, 20], stating the necessity and the basic mechanism, but without a thorough analysis of its effects on the response time. The re-sampling techniques we propose simplify the original distribution of WCETs but also take into account the pessimism that the operation introduces to the analysis. In this section we improve previous results presenting three new re-sampling techniques and studying their effect on the worst-case execution time distribution. In the next section we study the effects of the re-sampling on the response time distribution.

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

- *Selecting the k samples to be kept in the reduced distribution.*
- *Re-distribution of the probabilities from the values that are not kept.*

Both steps differentiate and characterize a re-sampling method, although in our opinion it is the selection of the remaining values that differentiates them most. Therefore the selection step is the place where the re-sampled distribution may be improved.

a) Selecting the k Samples.

In this phase, the question to answer is *how do we select k samples out of n?* For an input distribution of worst-case execution times, we know that the largest value has to remain in the re-sampled distribution in order to ensure the real-time analysis. Thus the worst-case value is guaranteed as well as the worst-case analysis. Deciding which are the $k-1$ samples to stay is the decision that impacts the most on the response time analysis. For instance in [16] the $k-1$ samples are chosen randomly, while in [20] they are chosen among those with the largest probabilities. We present in the next section three different selection techniques and study them in detail.

b) Probability re-distribution.

Once the k values to be kept have been selected, the probabilities of the un-drafted samples have to be re-distributed among these k values. This is imposed by the fact that the final result of re-sampling has to be a distribution. This means that the cumulative probability of the re-sampled distribution $\mathcal{C}'_i$ has to be equal to 1, $\sum_j \mathbb{P}\{\mathcal{C}'_i = C^j_i\} = 1$. Therefore we need to answer the question *How are the probabilities of the non selected samples re-distributed to the k selected samples?* In order to have a pessimistic re-sampled distribution, and hence a safe analysis[2], 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 probabilities of a larger value. We note that in [16] the probabilities of deleted values are accumulated and added to the probability of the highest value. In [20] the remaining probabilities are added to the probabilities of the k selected values. The later approach increases the accuracy of the analysis and at the same time decreases the pessimism. Therefore we use in this paper the probability re-distribution methods presented in [20].

Another issue that a re-sampling mechanism has to take into account is how large k should be. It is intuitive that k plays a key role in the accuracy of re-sampling methods. A large value of k means less probability mass to re-distribute, so more accuracy in the resulting distribution. On the other hand, a large computation time to get results from the analysis. So, k comes from a trade-off of complexity versus accuracy we want to achieve. In this paper we consider k as given.

EXAMPLE 3.2. *To better understand the re-sampling we present an example. We consider a task with execution time given by a random variable with 10 values. Without loss of generality we consider these ten values as the integer numbers from* 1 *to* 10*. The probabilities of these values are as follows:*

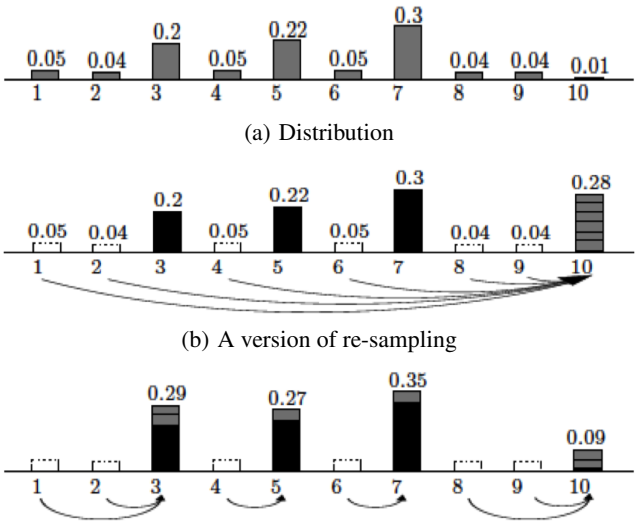
$$\mathcal{C} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 0.05 & 0.04 & 0.2 & 0.05 & 0.22 & 0.05 & 0.3 & 0.04 & 0.04 & 0.01 \end{pmatrix},$$

as can be seen in Figure 2(a). 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. According to [20], these* 4 *resulting values will be the last value of the original distribution (*10*-th value) and the* 3 *values with the highest probabilities, i.e., the* 7*-th value which has a probability of* 0.3*, the* 5*-th value which has a probability of* 0.22 *and the* 3*-rd value which has a probability of* 0.2*.*

The samples according to the method in [16] should result from a random selection. In order to compare the results with the next method, we keep the same values that we would have chosen with that solution. It is notable that the largest value of the distribution gets all the mass-probabilities of the values that were not selected to be kept in the new random variable. Because of that, the largest value has a higher probability associated with it compared to the original distribution, the resulting distribution being equal to $\mathcal{C}^I = \begin{pmatrix} 3 & 5 & 7 & 10 \\ 0.2 & 0.22 & 0.30 & 0.28 \end{pmatrix}$*, represented in Figure 2(b).*

[2]Safety and pessimism are clarified in sections 3.1 and 3.2 respectively

Applying the mechanism presented in [20], the re-sampled distribution becomes $\mathcal{C}^{II} = \begin{pmatrix} 3 & 5 & 7 & 10 \\ 0.29 & 0.27 & 0.35 & 0.09 \end{pmatrix}$*, Figure 2(c). With this method the probability of the unused samples are distributed to the closest larger sample in the new distribution. For instance here the probabilities associated with the values* 1 *and* 2 *go to value* 3 *which results in a probability of* $0.29 = 0.2+0.05+0.04$ *in the new random variable. The last sample ends up with a probability of* 0.09*, much lower than the one obtained using the mechanisms presented in [16], but still larger than in the original distribution. Since that value represents the largest execution time, then assigning a lower probability to it makes the second re-sampling method less pessimistic in terms of response time and the resulting deadline miss ratio (see Section 5). Nonetheless, this is not sufficient to judge the quality of re-sampling methods in real-time scenarios.*



(a) Distribution

(b) A version of re-sampling

(c) An improved version of re-sampling

Figure 2: The example distribution with two different basic re-samplings.

3.1 Safeness at Execution Time Level

A safe real-time analysis has to guarantee the results in any possible conditions including those assured by the worst-case values in the deterministic scenario. Since the re-sampling has an important impact on the real-time analysis, it is mandatory that re-sampling does not introduce optimism since the safety of real-time analysis has to be guaranteed. Such a requirement affects the way of doing re-sampling.

- If all random variables have a single value, then the re-sampling has to keep that value so that the analysis could find the same result as existing deterministic analyses; this is the *limit condition* [21] for re-sampling algorithms as well as probabilistic real-time analysis.
- A first constraint when re-sampling a worst-case execution time distribution is that the largest value of the original distribution has to be included in the final distribution being critical for the safety of the analysis. This is the *worst-case condition* [21] for both the analysis and the re-sampling (that the analysis requires and the re-sampling has to provide). Indeed, removing that value would provide optimistic results with respect to the exact analysis. The selection of the remaining values for the re-sampled distribution does not affect the safety because the selected samples are a subset of the original ones.
- The probability re-distribution among the new samples has an effect on the safety of the new distribution. In order to be safe, a distribution has to be pessimistic with respect to the original one which is considered exact. This is again the *worst-case condition*.

The input distribution of worst-case execution times (the original one) is the one that results in the precise analysis, so we consider it as the reference. We then say that a re-sampled distribution is safe, from the real-time analysis perspective if it is pessimistic with respect to the original distribution. In order to compare two distributions (original and re-sampled or two re-sampled distributions), we recall here the ordering relationship defined in [18].

DEFINITION 3.3 (ORDER AMONGST RANDOM VARIABLES [18]). *Let* $\mathcal{X}$ *and* $\mathcal{X}'$ *be two random variables.* $\mathcal{X}'$ *is greater than or equal to* $\mathcal{X}$ *(or alternatively,* $\mathcal{X}$ *is less than or equal to* $\mathcal{X}'$*), and denote it by* $\mathcal{X}' \succeq \mathcal{X}$ *(alternatively,* $\mathcal{X} \preceq \mathcal{X}'$*) if* $\mathbb{P}\{\mathcal{X}' \leq D\} \geq \mathbb{P}\{\mathcal{X} \leq D\}$ *(alternatively* $\mathbb{P}\{\mathcal{X}' \leq D\} \leq \mathbb{P}\{\mathcal{X} \leq D\}$*) for any* D*, and the two random variables are not identically distributed.*

The same ordering holds with the Cumulative Distribution Function (CDF) of PFs, $F(x) = \mathbb{P}\{\mathcal{X} \leq x\}$.

DEFINITION 3.4. *[18] The random variable* $\mathcal{X}'$ *is* greater than or equal to *the random variable* $\mathcal{X}$ *(*$\mathcal{X}' \succeq \mathcal{X}$*), if* $F_{\mathcal{X}'}(x) \leq F_{\mathcal{X}}(x)$*, for any* x*, and the two random variables are not identically distributed.*

COROLLARY 3.5 (SAFE RE-SAMPLING). *By moving probability mass from smaller execution times to larger execution times, we obtain a distribution that is greater than the original one. This new distribution is safe from the perspective of the execution time.*

Indeed, the method of accumulating the probabilities from discarded values up to the nearest larger values selected, so rightward with increasing ordered values, results in a distribution $\mathcal{X}^{re-sampled}$ 'greater than or equal to' the exact one $\mathcal{X}^{original}$. This ends up in a pessimistic distribution which is safe with respect to the original one.

From Corollary 3.5, the definition of ordering among distributions and the results in [16] we can conclude that any probability re-distribution 'from-left-to-right' is safe because the unselected probabilities are moved to larger execution time values. On the other hand, the re-distribution right-to-left produces optimistic results in terms of response time.

3.2 Metric of Pessimism

Once re-sampling has been performed on a distribution, we know that pessimism (inaccuracy) has been introduced due to the loss of information (with respect to the original distribution), but the question is *how much?* In order to have an answer to this question we need to define a metric that computes the pessimism introduced by the re-sampling, and maybe a metric or a method to compare two re-sampling techniques to see which one is better.

An effective approach is to multiply the probability and execution time values at each point together to obtain the overall 'weight' (linear combination) of the distribution and then compare the re-sampled distributions to the original. In the probability theory, this is the *expectation* $E(\mathcal{X})$ of a distribution $\mathcal{X}$, and is defined

as $\sum_{x \in val(\mathcal{X})} x \cdot \mathbb{P}\{\mathcal{X} = x\}$, where $val(\mathcal{X})$ is the set of values of $\mathcal{X}$. The weight W_i of a distribution i with k_i values is given by

$$W_i = \sum_{j=1}^{k_i} v_j \cdot p_j \quad (3)$$

where v_j is the value C_i^j at the j-th position in the distribution and p_j is it's corresponding probability, $\mathbb{P}\{\mathcal{C}_i = C_i^j\}$.

We note that the closer the weight of a re-sampled distribution is to the weight of the original distribution, the better the re-sample is at representing the distribution as a whole.

4. RE-SAMPLING: TECHNIQUES

In this section we present three of the re-sampling techniques that we have implemented and we currently use in our analysis tool which has been developed at Rapita Systems [22]. We compare these three methods in order to underline the impact that re-sampling has on the probabilistic response time analysis. The first one is a 'uniform spacing' re-sampling technique that is frequently used in practice and provides good results. The second one, which we introduce in this paper, is a more complex technique that provides better results, i.e., introduces less pessimism, compared with other techniques that we have studied. The third technique is 'domain quantisation' which reduces the number of distinct values produced by convolution and so increases performance.

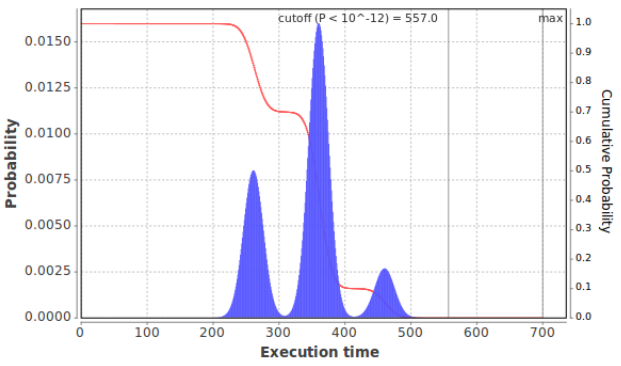


Figure 3: Original distribution

4.1 Uniform Spacing Re-sampling

We present here a first re-sampling method that we use and we have analyzed in order to see its effects on the response time analysis of probabilistic real time tasks. The technique is called *uniform spacing* and it is frequently used in the probabilistic real-time domain as it is easy to implement.

The re-sampling is done by choosing equally distanced values out of the initial distribution. Algorithm 1 describes the steps of this re-sampling technique. Here $\mathcal{C}$ is the initial distribution, $\mathcal{C}^{new}$ is the distribution obtaining after the re-sampling process. We denote by $\mathcal{C}.size$ the number of values of $\mathcal{C}$, and by $\mathcal{C}_j$ the j-th value of $\mathcal{C}$. The probability associated with the j-th value of $\mathcal{C}$ is denoted by $\mathcal{C}_j.prob$.

For example, by applying the *uniform re-sampling technique* (with the objective of obtaining 50 values) to the distribution presented in Figure 3, that has 650 values, then the new distribution contains every 13-th value of the original. This value gathers the probability mass of the 12 values that precedes it. The resulting distribution can be seen in Figure 4. Although it keeps the general form of the original distribution, it gathers high peeks, the largest probability being close to 0.2 compared to 0.016 in the initial distribution.

Algorithm 1 Uniform Spacing Re-sampling Algorithm

Input: $\mathcal{C}$ a distribution and k the number of values to be selected
Output: $\mathcal{C}^{new}$

```
m = 1;
p = 0;
q = ceil(C.size/k)
for i = 0; i ≤ C.size; i++ do
  p = p + C_j.prob;
  if i mod q = 0 or i = C.size then
    C_m^new.value = C_j.value
    C_m^new.prob = p
    p = 0;
    m = m + 1;
  end if
end for
```

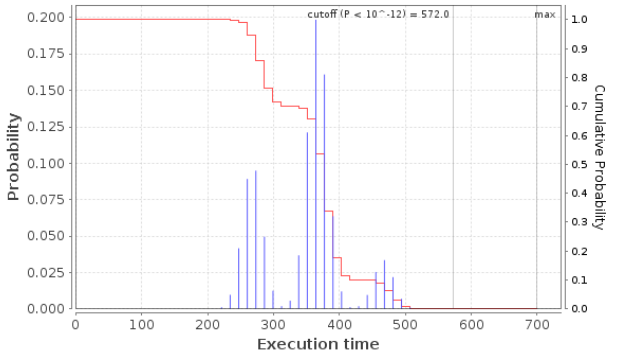


Figure 4: Re-sampling obtained by Uniform Spacing

4.2 Domain Quantization

We now introduce an enhanced re-sampling technique, which we refer to as Domain Quantisation. The rationale for this technique is as follows. When convolving two distribution that have m and respectively n values each, the resulting distribution can have up to $m \times n$ values. This is true when the two distributions that are convolved are very different from one another. In the best case the resulting distribution can have as little as $m + n - 1$ values. For large distributions, having only $m + n - 1$ values instead of $m \times n$ values can make a big difference, reducing the total amount of re-sampling required since the number of values per distribution does not increase so fast.

For example, by convolving two distributions that have respectively the values $(1\ 4\ 7)$ and $(2\ 6\ 19)$ we obtain a distribution that has the values$(3\ 6\ 7\ 9\ 10\ 13\ 16\ 19\ 22)$, i.e. by convolving two distributions that have three values each we obtain a distribution with nine values. If, on the other hand, the second distribution has the values $(9\ 12\ 15)$, so the distance between values is the same as for the first input distribution, then we obtain a resulting distribution

that has only five values, namely (10 13 16 19 22). By virtue of having less values, the resulting distribution does not need to be re-sampled and so less pessimism is introduced.

This suggests that a way of decreasing the number of values in the resulting distribution is to quantize the values of the input distributions using a re-sampling strategy so that they have the same spacing between them. In this way, a first re-sampling is necessary in order to obtain the same quantization for the input distributions. Further re-sampling of the results with the same quanta will then be unnecessary.

Choosing the quanta to be used is an important problem, since it determines the number of samples to be kept per distribution, scaling a large distribution to a large quanta means that few values are kept out of its initial number of values, and so the loss in precision is potentially large; on the other hand, scaling a large distribution to a small quanta results in too many values of the distribution to be kept, which makes the re-sampling inefficient.

This problem can be solved by taking advantage of the fact that the convolution is commutative, so, when there are multiple distributions to be convolved with each other, which is often the case in probabilistic response time analysis, first the small distributions (representing tasks with relatively short execution times) are convolved amongst themselves until they become bigger and they can be convolved with larger distributions. To facilitate this, we set the quanta for each distribution to the smallest power of 2 (e.g. 1,2,4,8...) that results in at most k samples. We note that this form of re-sampling has much in common with the Piece-wise Constant Approximation (PCA) method used in re-sampling time-series data [23].

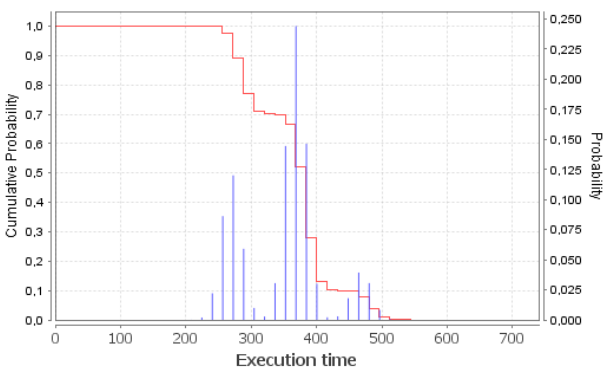


Figure 5: Re-sampling obtained by Domain Quantization

4.3 Reduced-Pessimism Re-sampling

A uniform selection of values for re-sampling will, most of the time, not result in a satisfactory reallocation of probability mass. Some of the selected values will accumulate a disproportionate share of the overall probability. This can lead to large amounts of pessimism in the re-sampled profile, and this pessimism is compounded by later applications of re-sampling which may occur during response time computation.

We have implemented an improved algorithm for the selection of re-sampling values which we call *Reduced-Pessimism Re-sampling*. It works by considering ranges of values and calculating the pessimism that would be introduced if the range of values were to be aggregated into a single entry with the highest value in the range taking all of the probability mass.

Algorithm 2 describes the method we use for computing the pessimism, which is based on the relative probability 'weight' as described in Section 3.2: the execution time multiplied by the probability.

Algorithm 3 provides the method by which re-sampling values are selected. Starting with the entire distribution, ranges are examined to identify the range with the highest pessimism. The selected range is then split into two sub-ranges. When sufficient sub-ranges have been generated, the upper bound of each range is used to perform re-sampling using the method described in Section 4.1.

Algorithm 2 Algorithm to compute the pessimism associated with replacing a range with a single value

Input: $\mathcal{C}$ and (x, y) a range of values;
Output: p pessimism;
$shifted = (\sum_{i=x}^{y} \mathcal{C}_i.probability) \times \mathcal{C}_y.value$;
$original = \sum_{i=x}^{y} (\mathcal{C}_i.probability \times \mathcal{C}_i.value)$;
$p = shifted - original$;

Algorithm 3 Algorithm for selecting values which create the least pessimism when re-sampling

Input: $\mathcal{C}$ a distribution and k the number of values to be selected
Output: $\mathcal{C}^{new}$
$\mathcal{Q} = \varnothing$; // Priority queue
$r = (1, \mathcal{C}.size)$; // Full range of $\mathcal{C}$
$p = pessimism(\mathcal{C}, r)$;
$\mathcal{Q}.add(r, p)$; // Add r to $\mathcal{Q}$ with priority p
while $\mathcal{Q}.size < n$ **do**
  $r = \mathcal{Q}.remove\_first$;
  $(a, b) = split(r)$; // Split into two equal sub-ranges
  $\mathcal{Q}.add(a, pessimism(\mathcal{C}, a))$;
  $\mathcal{Q}.add(b, pessimism(\mathcal{C}, b))$;
end while
$\mathcal{C}^{new} = resample(\mathcal{C}, \mathcal{Q}.upper\_bounds)$;

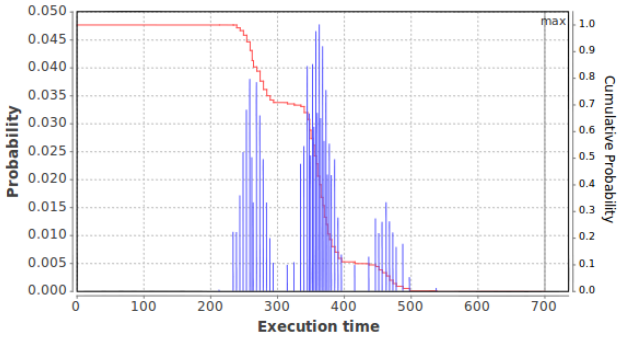


Figure 6: Re-sampling obtained with Reduced-Pessimism method

4.4 Comparison

To compare the three methods at execution time level we make use of the metric presented in section 3.2. The original distribution depicted in Figure 3 has a weight of 340, while the distribution obtained using Uniform Spacing and Reduced-Pessimism

Re-sampling have weights of 346 and 342 respectively. The distribution obtained using Domain Quantization has the highest weight, equal to 348.

At execution time level the Reduced-Pessimism Re-sampling technique performs best, having a weight that is closer to that of the original distribution.

We will further analyze the three re-sampling methods in the next section, where they are applied in the response time analysis and compared via the pessimism introduced in the exceedence functions, i.e. at the level of deadline miss probabilities.

5. IMPACT OF RE-SAMPLING ON THE RESPONSE TIME ANALYSIS

In probabilistic analysis, where the response times are random variables, the real-time constraints are expressed in terms of probabilities of deadline misses. The response time $\mathcal{R}$ is one of the metrics applied to evaluate the probabilistic real-time analysis. Based on that, Diaz et al. in [18] have defined relationships in order to tackle the pessimism in probabilistic analyses.

DEFINITION 5.1 (EXACT PROBABILISTIC ANALYSIS). *An exact random variable defines the exact results of a probabilistic analysis.*

A probabilistic response time $\mathcal{R}'$ is pessimistic when it is greater than or equal to the exact one $\mathcal{R}$, $\mathcal{R}' \succeq \mathcal{R}$. The challenge when using re-sampling techniques is how to perform approximations with probabilistic analysis parameters, while guaranteeing that the resultant response time is pessimistic or identically distributed, thus making the approximated analysis safe.

The pessimism introduced when re-sampling execution times is propagated through the analysis to the response time distribution. A good re-sampling technique should not introduce so much pessimism that the system is deemed unfeasible when in reality it is feasible, i.e., a more precise analysis done on the system without any re-sampling would deem the system feasible. In other words, the response time distribution obtained when re-sampling is performed should be as close as possible to that obtained when re-sampling is not performed.

A way of computing the relative pessimism introduced at the response time level is to use the weight presented in the previous section, Equation (3). A simpler way is to compare the exceedence functions and deadline miss probability of the jobs [24] for which we have computed the response time.

DEFINITION 5.2 (JOB DEADLINE MISS PROBABILITY). *For a job $\tau_{i,j}$ and a priority assignment Φ, the deadline miss probability $\mathsf{DMP}_{i,j}$ is the probability that $\tau_{i,j}$ misses its deadline:*

$$\mathsf{DMP}_{i,j}(\Phi) = \mathbb{P}\{\mathcal{R}_{i,j}(\Phi) > D_i\}. \quad (4)$$

When re-sampling is done, the pessimism introduced translates in an increased DMP. A decreased DMP in the re-sampled results with respect to the original results means that optimism has been introduced and the result is no longer safe.

6. EXPERIMENTAL INVESTIGATION

In this section, we provide details of three experiments used to investigate the performance and precision of the three re-sampling techniques described in section 4.

As the basis of these experiments, we used randomly generated sets of tasks with a PF characterizing the worst-case execution time of each task. We assumed that the tasks were executed non-preemptively and that their periods were long in relation to the response time. Hence the problem simply involved the convolution of the PFs of the tasks. (Thus each convolution effectively represents the addition of a further task; or equivalently, the computation of the response time of a further lower priority task). This assumption does not change the nature of the problem, it simply means that less convolutions need to be performed to compute the response time of a job. We have opted for a simplified problem since the task-sets that we analyze are randomly generated and taking into account preemptions would have meant a more strict control on the generated task-sets, in order to avoid cases when a job suffers many preemptions, making the problem highly intractable, or even that the mentioned job never finishes execution. Nevertheless, the effects of the re-sampling strategies proposed and the results presented are valid for any number of convolutions and tasks in a task-set, regardless on whether preemptions are present or not.

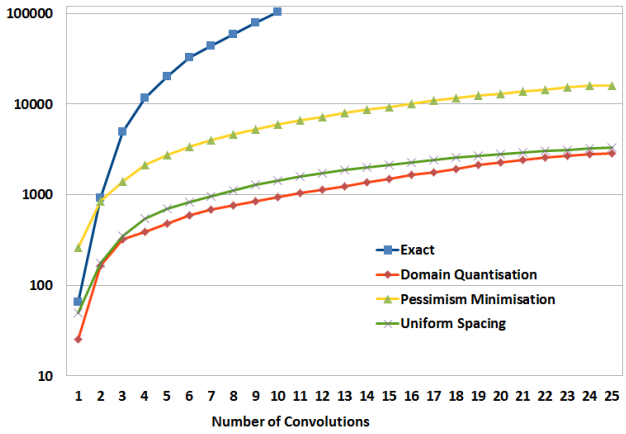


Figure 7: Computation time in the exact case and with different re-samplings

The PFs contained 100 values spread over a range of approximately 10,000 possible values, intended to simulate execution times measured to an accuracy of 0.1 microseconds, with a largest value of around 1 millisecond, typical of embedded real-time systems.

All the data was obtained from a prototype java implementation running on an i7-2720QM CPU @ 2.20GHz computer.

In the first experiment, we examined the computational cost of the three re-sampling techniques, compared to an exact approach. For this experiment, 10 task-sets were randomly generated with 25 tasks each. When re-sampling was enabled, once the PF of each task was convolved, the resulting distribution was re-sampled to a maximum of 1000 values (the re-sampling threshold). Figure 7 plots the total computation time required in milliseconds (averaged over 10 task-sets) versus the number of convolutions from 1-25. Note the log scale on the graph. From Figure 7 it is clear that the exact approach initially exhibits exponential growth, with the computation time required approaching 100,000ms for 10 convolutions. All of the re-sampling techniques exhibit significantly improved performance with respect to the exact analysis; with an improvement of around a factor of 100 for 10 convolutions for the Domain Quantisation technique. We observe that the more complex reduced pessimism re-sampling takes significantly longer than the uniform spacing and domain quantisation techniques.

In the second experiment, we examined the precision of the re-sampling techniques by comparing the exceedence functions $(1 -$

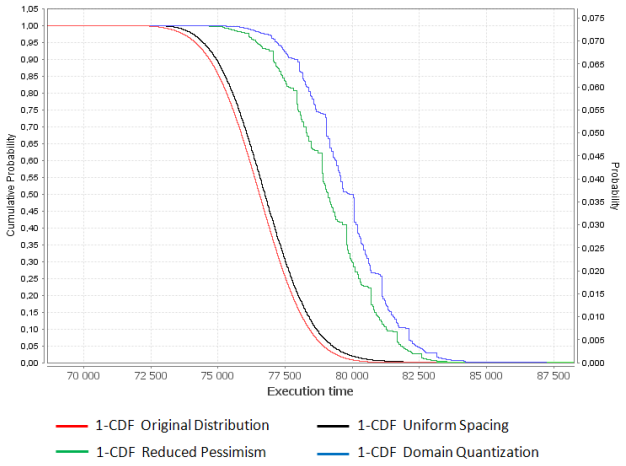


Figure 8: $1 - CDF$ **of re-sampled distributions and exact one**

CDF) of the resulting distribution after 25 convolutions, i.e., the response time of the 25th highest priority task. This experiment used a similar configuration to the first experiment, i.e., a randomly generated task-set with 25 tasks, each task having an execution time given by a random variable with 100 values, but a lower re-sampling threshold of 100 to highlight the differences between the re-sampling techniques. Figure 8 shows the exceedence functions for the resultant distribution. We observe that the reduced pessimism technique results in a distribution that remains very close to the exact distribution at high probabilities, even when the re-sampling threshold is just 100 values. This is because the technique is designed to minimise the increase in the weight of the distribution. However, Figure 9 shows the same exceedence functions on a log scale, thus illustrating the tails of the distributions. This shows that by selecting values to minimise the overall pessimism in the distribution, the shape of the tail is heavily compromised. (The reduced pessimism technique does not provide enough values in this region where the probabilities are small). With the Uniform Spacing and Domain Quantisation techniques although correspondence to the exact distribution is worse at high probabilities, it is much better in the tail of the distribution. This is reflected by the fact that the response time with a 10^{-9} probability of being exceeded is 89,600 for the exact distribution, 94,300 for the Domain Quantisation technique, and 141,500 for Reduced Pessimism re-sampling.

In the third experiment, we examined the trade-off between performance (computation time) and pessimism introduced in terms of the weight of the resulting distributions as a function of varying the re-sampling threshold. The experimental configuration was similar to that of the second experiment, but averaged over 10 task-sets, i.e. 10 runs, 25 tasks per task-set, 100 values per random variable. Figure 10 gives corresponding results showing how the weight of the resultant distributions changes in relation to the exact analysis. The values plotted in Figure 10 are the ratio of the weight of the distribution produced via the re-sampling technique to the weight of the distribution from the exact analysis. Here the Reduced Pessimism technique shows a significant advantage albeit at the expense of precision in the tail of the distribution.

7. FUTURE IMPROVEMENTS

While re-sampling is a good way of reducing the computation time of convolutions, it is not the only one. There are further improvements that can be made at the implementation level so that the

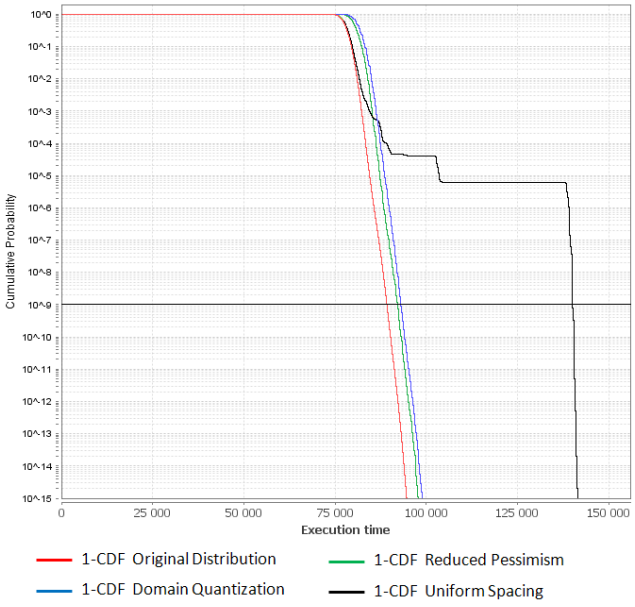


Figure 9: Zoom-in in logarithmic scale on $1 - CDF$ **of re-sampled distributions and exact one**

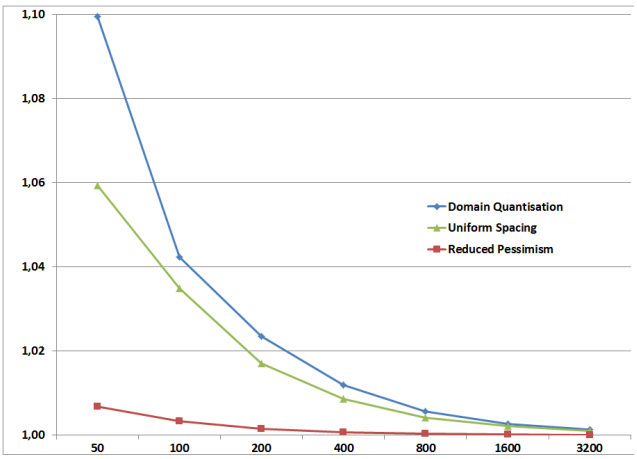


Figure 10: Relative weight of re-sampled distributions with different re-sampling thresholds after 10 **convolutions**

time it takes to perform convolutions and response time analysis is reduced even more.

7.1 Range quantization

One reason why performing convolutions is costly, besides the fact that there are many operations (multiplications, additions) to be performed, is how these operations are performed and, implicitly, the way data is stored and handled by the computer.

In order to represent and to perform operations on very small numbers, of the order of 10^{-k} where k can be tens or even hundreds, one needs to use arbitrary precision arithmetic, which is very expensive.

One solution is to change the representation to something faster and simpler, like floats in the manner of one or few significant figures in the mantissa and a variable exponent. For example, the number $x = 0.0000000000000007$ can be written as 7×10^{-16}, so, one only needs to store the mantissa, which in this case is 7, and the exponent, which is 16.

Using this form of representation, performing multiplication is as simple as multiplying the two mantissas and adding the two exponents.

The addition of two such numbers can be more complicated if they are of different sizes, for example adding 7×10^{-16} and 2×10^{-4} results in $2.00000000007 \times 10^{-4}$ which takes us back to the problem of requiring a large amount of precision. This can be solved by truncating the probability value to the largest exponent, keeping a few values in the mantissa, and moving the probability mass that is cut off to the smallest larger value of the probability distribution, computing it and potentially decreasing its exponent. The truncation is done to all values of the distribution, starting from the smallest one to the largest one.

8. CONCLUSIONS

In this paper, we addressed the problem of re-sampling complex distributions of worst-case execution times, with a large number of values. We compared three re-sampling techniques, uniform spacing which is commonly used, and two techniques which we derived; reduced pessimism re-sampling and domain quantisation.

We showed via a series of experiments that all three techniques are effective in addressing the complexity problem of exact analysis, which exhibits an exponential increase in computation times with the number of convolutions.

Our experiments showed that reduced pessimism re-sampling is highly effective in terms of obtaining a close, but safe approximation to the distribution resulting from exact analysis at high levels of probability, even when the re-sampling threshold is set as low as 100 values. However, selecting points in this way can compromise the shape of the tail of the distribution leading to significant and undesirable pessimism in that region, which is an issue for probabilistic real-time analysis. The domain quantisation and uniform spacing techniques did not suffer from this problem to the same extent.

In future we intend to investigate the performance of re-sampling techniques aimed at preserving the shape of the tail of the distribution after a variable number of convolutions. We also intend to investigate the use of the complementary technique referred to as range quantisation (described in section 7). This technique is orthogonal to the re-sampling techniques described in this paper, and so can be applied in conjunction with any of them. Range quantisation removes the requirement to use arbitrary precision arithmetic and so can significantly increase the speed of the basic multiplication and addition operations used in convolution. Initial experiments in this area are promising, with the pessimism introduced by range quantisation found to be insignificant.

9. AKNOWLEDGEMENTS

The research leading to these results has received partial funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under the PROARTIS Project grant agreement no. 249100 (www.proartis-project.eu) and the UK EPSRC funded Tempo project (EP/G055548/1).

10. REFERENCES

[1] A. Burns, G. Bernat, and I. Broster, "A probabilistic framework for schedulability analysis," in *Third International Embedded Software Conference (EMSOFT 2003)*, 2003, pp. 1–15.

[2] M. Gardner and J. Lui, "Analyzing stochastic fixed-priority real-time systems," in *5thInternational Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 1999.

[3] G. A. Kaczynski, L. L. Bello, and T. Nolte, "Towards stochastic response-time of hierarchically scheduled real-time tasks," in *Proceedings of 11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2006)*, 2006, pp. 453–456.

[4] 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 *Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS 2002)*, 2002, p. 289.

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

[6] A. Atlas and A. Bestavros, "Statistical rate monotonic scheduling," in *19th IEEE Real-Time Systems Symposium (RTSS 1998)*, 1998.

[7] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzeti, Q. E., and F. Cazorla, "Measurement-based probabilistic timing analysis for multi-path programs," in *the 24th Euromicro Conference on Real-Time Systems (ECRTS12)*, 2012.

[8] D. Khan, N. Navet, B. Bavoux, and J. Migge, "Aperiodic traffic in response time analyses with adjustable safety level," in *14th IEEE International Conference on Emerging Techonologies and Factory Automation (ETFA2009)*, 2009.

[9] K. Kim, J. L. Diaz, L. Lo Bello, J. M. Lopez, C.-G. Lee, and S. L. Min, "An exact stochastic analysis of priority-driven periodic real-time systems and its approximations," *IEEE Trans. Comput.*, vol. 54, no. 11, pp. 1460–1466, 2005.

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

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

[12] J. Beirlant, J. Teugels, Y. Goegebeur, J. Segers, D. D. Waal, and C. Ferro, *Statistics Of Extremes: Theory And Applications*, Wiley, Ed. Wiley, 2004.

[13] I. B. Y. Lu, T. Nolte and L. Cucu, "A new way about using statistical analysis of worst-case execution times," in *in the*

WiP sesion of the Euromicro Conference on Real-Time Systems, 2011.

[14] J. Diaz, J. Lopez, G. M., A. Campos, K. Kim, and L. Lo Bello, "Pessimism in the stochastic analysis of real-time systems: Concept and applications," in *25th IEEE International Real-Time Systems Symposium (RTSS 2004)*, 2004, pp. 197–207.

[15] W. A. Fuller, *Sampling Statistics*, Wiley, Ed. Wiley, 2009.

[16] K. S. Refaat and P.-E. Hladik, "Efficient stochastic analysis of real-time systems via random sampling," in *IEEE Euromicro Conference on Real-Time Systems (ECRTS 2010)*, 2010, pp. 175–183.

[17] F. Cazorla, E. Quinones, T. Vardanega, L. Cucu-Grosjean, B. Triquet, G. Bernat, E. Berger, J. Abella, F. Wartel, M. Houston, L. Santinelli, D. Maxim, L. Kosmidis, and C. Lo, "Proartis: Probabilistically analyzable real-time system," *ACM Transactions on Embedded Computing Systems*, 2012.

[18] J. M. López, J. L. Díaz, J. Entrialgo, and D. García, "Stochastic analysis of real-time systems under preemptive priority-driven scheduling," *Real-Time Syst.*, pp. 180–207, 2008.

[19] L. Santinelli, P. M. Yomsi, D. Maxim, and L. Cucu-Grosjean, "A component-based framework for modeling and analyzing probabilistic real-time systems," *16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'11)*, 2007.

[20] D. Maxim, L. Santinelli, and L. Cucu-Grosjean, "Improved sampling for statistical timing analysis of real-time systems," in *4th Junior Researcher Workshop on Real-Time Computing*, 2010.

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

[22] "http://www.rapitasystems.com/."

[23] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Knowledge and Information Systems*, vol. 3, pp. 263–286, 2001, 10.1007/PL00011669. [Online]. Available: http://dx.doi.org/10.1007/PL00011669

[24] D. Maxim, O. Buffet, L. Santinelli, L. Cucu-Grosjean, and R. Davis, "Optimal priority assignments for probabilistic real-time systems," in *the 19th International Conference on Real-Time and Network Systems (RTNS2011)*, 2011.