A Framework for the Response Time Analysis of Fixed-Priority Tasks with Stochastic Inter-arrival Times

Liliana Cucu
IPP-Hurray Research Group
Polytechnic Institute of Porto
ISEP, Rua Dr. A. Bernardino de Almeida, 431
4200-072 Porto Portugal
+351228340529
lcucu@dei.isep.ipp.pt

Eduardo Tovar
IPP-Hurray Research Group
Polytechnic Institute of Porto
ISEP, Rua Dr. A. Bernardino de Almeida, 431
4200-072 Porto Portugal
+351228340502
emt@dei.isep.ipp.pt

ABSTRACT

Real-time scheduling usually considers worst-case values for the parameters of task (or message stream) sets, in order to provide safe schedulability tests for hard real-time systems. However, worst-case conditions introduce a level of pessimism that is often inadequate for a certain class of (soft) real-time systems. In this paper we provide an approach for computing the stochastic response time of tasks where tasks have inter-arrival times described by discrete probabilistic distribution functions, instead of minimum inter-arrival (MIT) values.

Categories and Subject Descriptors

D.4.7 [**Operating Systems**]: Organization and design – *real-time systems and embedded systems*; C.3 [**Special-purpose and Application-based Systems**]: *real-time and embedded systems*

General Terms

Algorithms, Design

Keywords

Real-time systems, Schedulability analysis, Stochastic analysis, Fixed-priority scheduling

1. INTRODUCTION

A real-time system is usually characterized by deadlines, and any schedulability analysis for these systems tries to answer the question of whether deadlines are ever met or not. According to the consequences of missing deadlines, a real-time system is often categorized as hard or soft. In a hard real-time system no deadlines can ever be missed, while in a soft real-time system deadline misses can be tolerated from time to time. In this case, the system designer may like to have a measure of the minimum likelihood to miss deadlines.

Worst-case approaches taking minimum inter-arrival times (MIT) as tasks' periods introduce an unacceptable level of pessimism in the context of soft real-time systems. This is particularly true in those systems where inter-arrival times of jobs (or messages) have a large and known variability. Deadlines may be guaranteed, but actual (allowed by the schedulability test) system utilization turns out to be dramatically low.

In this paper, we are particularly interested in systems in which worst-case computation times are constant, and inter-arrival times are defined by random variables rather than by MIT values. The results provided are of particular interest for distributed real-time systems, where message streams are queued and distributed stochastically. Typically, in these systems, message durations (equivalent to jobs' computation time) have a small variability.

There has been a significant number of research works devoted to this problem. However, and to our best knowledge, there are no comparable results so far. In fact, there are a few related works which consider special scheduling models providing isolation between tasks [1], or assuming a known (*a priori*) maximum number of arrivals, thus introducing an unnecessary level of pessimism in the analysis [2]. The results provided in [2] are improved in [3], albeit considering only one task described by random inter-arrival times. In [4], the author addresses the problem in a context where both computation and inter-arrival times are described by random variables. The approach is based on a modification of the queuing theory for real-time systems, but results are only valid for particular cases of random variables. In [5] the same approach is applied for the case of general random variables, but all inter-arrival times of a task are defined by the same distribution function. A more general approach is proposed in [6], which only addresses the case of variability in computation times, being their approach not extendible to the case of random inter-arrival times.

In this paper we tackle the problem of random inter-arrival times given by generic discrete probabilistic distribution functions when fixed-priority systems are considered. Throughout this paper the computation times are constant and we consider worst-case values. The distribution functions are considered given and known, either analytically or obtained from experiments. Their determination is not the purpose of this paper.

The rest of the paper is organized as follows. Section 2 defines the task model. In Section 3 we propose an approach for a stochastic response time analysis of tasks when higher priority random tasks exist. Section 4 illustrates the application of the approach to an example task set, and ongoing work is drawn in Section 5.

2. TASK SET MODEL

We consider a system of n independent tasks:

$$S = \{\tau_1, ..., \tau_n\}$$

The tasks have fixed and unique priorities. Without loss of generality, tasks are ordered according to the non-decreasing order of their priorities. Each task τ_i is described as follows:

$$\tau_i = (\boldsymbol{T}_i, C_i, D_i), \quad \forall i \in \{1, ..., n\}$$

In the definition above, $\boldsymbol{T}_i$ corresponds to a non-negative discrete random variable. Throughout the rest of this paper, variables in bold will denote discrete random variables. In the above task definition, C_i corresponds to the usual worst-case execution time (WCET) of τ_i and D_i to its relative deadline.

Each new release j of a task τ_i may be defined by different random variables. We denote by $\boldsymbol{T}_{i,j}$ the random variable giving the inter-arrival time between the $(j-1)^{th}$ and the j^{th} job of a task τ_i. It is assumed that these $\boldsymbol{T}_{i,j}$ random variables are independent of other jobs of task τ_i and of jobs of other tasks τ_k (with $k \neq i$).

We consider $\boldsymbol{T}_{i,j}$ to be discrete, and thus its probability mass function is given by:

$$\boldsymbol{T}_{i,j} = \begin{pmatrix} t_k \\ P(T = t_k) \end{pmatrix}_{k \in \{1, ..., k_{i,j}\}} \tag{1}$$

The possible values of t_k in the definition of $\boldsymbol{T}_{i,j}$ are as follows:

$$t_k \in \left[t_{i,j}^{\min}, t_{i,j}^{\max}\right]$$

We denote the set of possible values of t_k by $\nu(\boldsymbol{T}_{i,j})$. Its cumulative distribution function is defined as follows:

$$F_{\boldsymbol{T}_{i,j}}(t) = P(\boldsymbol{T}_{i,j} \leq t), \text{ with } t \in \left[t_{i,j}^{\min}, t_{i,j}^{\max}\right] \tag{2}$$

For the sake of simplicity, from now on we denote as periodic tasks those tasks τ_i for which the random variable concerning inter-arrival information is a constant, while tasks with at least two possible values for the inter-arrival information are denoted as random tasks. Thus, the periodic tasks are those with $k_{i,j} = 1$, $\forall_{j \in \aleph}$ in Equation (1). We denote by P_{hpi} the set of indexes of periodic tasks which have a priority higher or equal to the priority of the task under analysis. Similarly, we denote by R_{hpi} the set of indexes of random tasks which have a priority higher or equal to the priority of that task. Obviously $P_{hpi}, R_{hpi} \subseteq \{1, \ldots, n\}$.

3. STOCHASTIC ANALYSIS - OVERVIEW

In this section we will briefly outline our approach for computing the stochastic response time of fixed-priority tasks having stochastic inter-arrival times. In Section 4, some further details will be provided.

3.1 Assumptions

We suppose, without loss of generality, that task τ_i is a periodic or sporadic task for which response time needs to be evaluated. We denote this response time by $\boldsymbol{R}_i$, which is a random variable that gives a set of values for the response time, and for each of these values, its probability.

Also without loss of generality, we assume a pre-emptive context. A non pre-emptive formulation (usually required for message stream models) can then easily be derived.

For the sake of simplicity we consider that all tasks, including random tasks, have the first release time equal to zero. Nonetheless, this does not imply a loss of generality of our results. It can be easily proved that the worst-case response time of a task is obtained in this case even if there are random tasks with higher priority. Also for simplicity, and without loss of generality, we consider that all inter-arrival times of each random task are defined by the same random variable. This last assumption would mean that Equation (1) is simplified as follows:

$$\boldsymbol{T}_{i,j} = \boldsymbol{T}_i = \begin{pmatrix} t_k \\ P(T = t_k) \end{pmatrix}, \quad \forall_{\tau_i} \text{ and with } k \in \{1, ..., k_i\} \tag{3}$$

3.2 Overview of the Approach

If the task set is strictly composed of periodic tasks, then the following well-know formulation [7] allows computing the worst-case response time of a task τ_i:

$$R_i = C_i + \sum_{j \in hpi} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j \tag{4}$$

where *hpi* denotes the set of tasks with a priority higher or equal to that of τ_i. Equation (4) is solved iteratively, by forming a recurrence relationship. However, the above formulation does not hold if random tasks with higher or equal priority exist in the task set, unless MIT values are taken into the formulation. Therefore, while trying to adhere to the above formulation, we introduce an alternative approach:

$$\boldsymbol{R}_i = \boldsymbol{C}_i \otimes \left(\otimes_{k \in P_{hpi}} \left\lceil \frac{\boldsymbol{R}_i}{T_k} \right\rceil \times C_k \right) \otimes (\otimes_{k \in R_{hpi}} \boldsymbol{N}_k \times C_k) \tag{5}$$

In Equation (4), $\boldsymbol{C}_i$ is a constant random variable with a unique value C_i, which corresponds to the worst-case execution time (WCET) of τ_i. $\boldsymbol{N}_k$ is the random variable giving the number of arrivals of τ_k together with the probabilities of those numbers. With the symbol $\otimes$ we denote the convolution of two random variables. Note that the sum of two independent discrete random variables is given by the convolution of the two.

We will now briefly outline an algorithm able to efficiently find a solution for Equation (5). The solution is found, similarly to what happens with Equation (4), by forming a recurrence relationship, albeit with additional details to properly handle the stochastic nature of the formulation. Let m denote each of the iterations performed to find the solution of Equation (5). In each iteration m, we perform the two steps described next.

3.2.1 Step 1

Firstly, we determine an intermediate pseudo-random variable, denoted as $\mathbf{L}_i^m$, which has $|\nu(\mathbf{R}_i^{m-1})|$ values. By $|\nu(\mathbf{R}_i^{m-1})|$ we understand the set of values of random variable $\mathbf{R}_i^{m-1}$. Therefore, each value L_i^m of $\mathbf{L}_i^m$ corresponds to a value $r_v^{m-1} \in \nu(\mathbf{R}_i^{m-1})$, thus computed in iteration $m-1$. Basically in this first step of iteration m, we compute the level-i busy period taking into account new arrivals of tasks belonging to the set P_{hpi}, while using a fixed number of instances of tasks belonging to R_{hpi} as computed in iteration $m-1$.

Each value $L_i^m(r_v^{m-1})$ of the $|\nu(\mathbf{R}_i^{m-1})|$ values of $\mathbf{L}_i^m$ is computed by the following equation, which does not contain any random variable:

$$L_i^m\left(r_v^{m-1}\right) = C_i + \sum_{k \in P_{hpi}} \left\lceil \frac{L_i^m\left(r_v^{m-1}\right)}{T_k} \right\rceil \times C_k + \sum_{k \in R_{hpi}} N_k\left(r_v^{m-1}\right) \times C_k \qquad (6)$$

In Equation (6), $N_k(r_v^{m-1})$ is the number of arrivals of τ_k as computed in iteration $m-1$. Equation (6) is solved iteratively by forming a recurrence relationship. For the first of the m iterations ($m = 1$) the relevant values for initiating the iterative procedure to solve the only Equation (6) are as follows:

$$\begin{cases} \mathbf{R}_i^0 = R_i^0 = \begin{pmatrix} r_1^0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ \text{thus, the relevant value } r_1^0 = 0 \\ N_k\left(r_1^0\right) = n_k^0 = 0, \quad \forall k \in R_{hpi} \end{cases}$$

In summary, an intermediate pseudo-random variable $\mathbf{L}_i^m$ is build upon the $|\nu(\mathbf{R}_i^{m-1})|$ results obtained for $L_i^m(r)$ solutions such that each value of the $\mathbf{L}_i^m$ is equal to a solution of one equation and their probabilities are those of r_v^{m-1} in $\mathbf{R}_i^{m-1}$.

As an example, in iteration $m = 1$, $\mathbf{L}_i^1$ is going to have only one value with probability 1, since no interference of tasks belonging to set of R_{hpi} are considered in the first iteration:

$$\mathbf{L}_i^1 = \begin{pmatrix} L_{i,1}^1 = C_i + \sum_{k \in P_{hpi}} \left\lceil \frac{L_{i,1}^1}{T_k} \right\rceil \times C_k + \sum_{k \in R_{hpi}} 0 \times C_k \\ 1 \end{pmatrix}, \ \forall i \in \{1,\ldots,n\}$$

In the formulation above, and the in $L^1_{i,v}$, v denotes the index of each of the different values that the pseudo-random variable may assume.

3.2.2 Step 2

In the second step of each iteration m, a new random variable $\mathbf{R}_i^m$ is obtained by convoluting $\mathbf{L}_i^m$ with a random variable corresponding to the interference caused by new arrivals of tasks belonging to R_{hpi} given the new level-i busy period as computed in step 1. Therefore,

$$\mathbf{R}_i^m = \mathbf{L}_i^m \otimes \left(\otimes_{k \in R_{hpi}} \mathbf{\Delta}_k^m \times C_k \right) \qquad (7)$$

$\mathbf{\Delta}_k^m$ is a random variable which is computed for each of the values obtained for $\mathbf{L}_i^m$ in step 1 for each of the random tasks belonging to R_{hpi}. Further details on $\mathbf{\Delta}_k^m$ will be provided in Section 3.3.

At this point, the variable $\mathbf{R}_i^m$ will correspond to the random variable $\mathbf{R}_i^{m-1}$ to which new arrivals of higher priority periodic and random tasks are added.

Actually, Equations (6) and (7) are particular analytical formulations of a more general equation which would allow us to integrate both steps of each iteration m into only one step. The obvious reason for separating it into two steps is that the integration would increase considerably the computational cost of the approach.

3.2.3 Improvement of the Approach

The iterative procedure finishes when at the end of an iteration m, either $\mathbf{R}_i^m$ contains only unchanged values or all new values are larger than the deadline. Obviously the response time is a non-decreasing function and the probability of the values is decreasing, which implies the convergence of the algorithm.

In order to detect when the response time, which is a random variable, contains only unchanged values we need to define a proper comparison function between two random variables.

Additionally, we could continue the iteration also with the unchanged values, but any further iteration will leave them unchanged. The presence of these values would unnecessarily increase the complexity of the approach, this without producing any new values for the response time in the next iteration.

We start an iteration m with a known random variable $\mathbf{R}_i^{m-1}$. Then we obtain an intermediate pseudo-random variable ($\mathbf{L}_i^m$) and finally we compute $\mathbf{R}_i^m$. In order to "eliminate" the unchanged values, we use the `Comp` function applied to the random variables $\mathbf{L}_i^m$ and $\mathbf{R}_i^m$ in order to obtain the random variable that will actually be used as initial value in iteration $m+1$.

As an example, assume the following scenario at a given m iteration:

$$\begin{cases} \mathbf{L}_i^m = \begin{pmatrix} 1 & 3 & 4 \\ 0.5 & 0.2 & 0.3 \end{pmatrix} \\ \mathbf{R}_i^m = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 0.1 & 0.4 & 0.3 & 0.1 & 0.1 \end{pmatrix} \end{cases}$$

The result of applying the `Comp` function will be:

$$\mathbf{R}_i^m = Comp\left(\mathbf{L}_i^m, \mathbf{R}_i^m\right) = \begin{pmatrix} 2 & 5 \\ 0.4 & 0.1 \end{pmatrix}$$

Therefore, we actually apply this "transformation" in $\mathbf{R}_i^m$ before starting a new iteration m. This is illustrated in Figure 1, which describes a simplified version of our approach.

3.3 Further Details

In this section we will further detail the step 2 briefly described in Section 3.2.2. Specifically, we start by introducing the proposed analytical formulation for the random variable Δ_k^m introduced in Section 3.2.2 in Equation (7). The proposed refinement is as follows:

$$\mathbf{R}_i^m = \mathbf{L}_i^m \otimes \left(\otimes_{k \in R_{hpi}} Shift\left(n_0^k, \mathbf{F}_k^*\left(v\left(\mathbf{L}_i^m\right)\right)\right) \times C_k \right) \qquad (8)$$

In Equation (8), the function $\mathbf{F}_k^*$ gives a random variable with the number of arrivals of random task τ_k from the time instant 0 to the time instant corresponding to the value to $v(\mathbf{L}_i^m)$, together with the probabilities of each of those number of arrivals to occur.

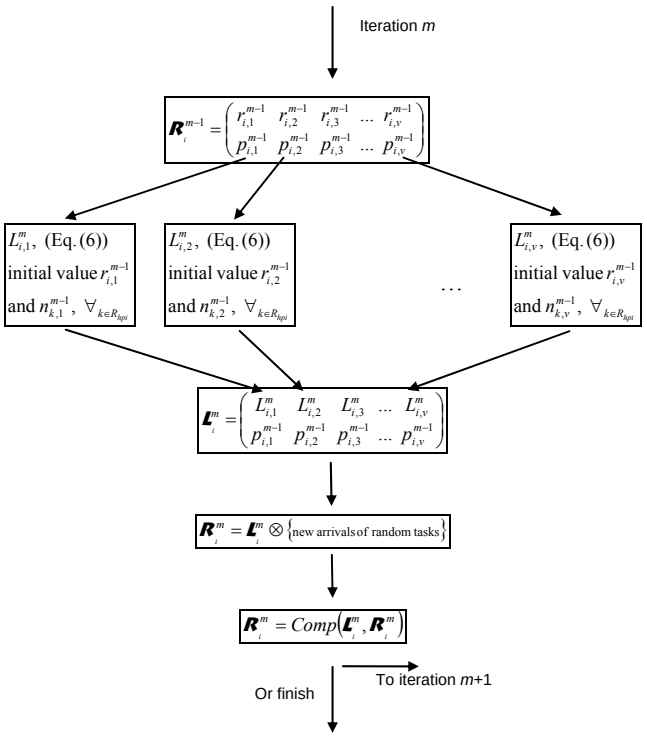

Figure 1. Simplified version of the approach.

We will now reason on how to compute these values. Let $\mathbf{X}_{k,j}$ be the random variable giving the values of release times of the j^{th} job of a task τ_k. This random variable is obtained as follows:

$$\mathbf{X}_{k,j} = \mathbf{X}_{k,j-1} \otimes \mathbf{T}_k, \text{ with } \mathbf{X}_{k,0} = \mathbf{T}_k$$

This formulation is correct because $\mathbf{X}_{k,j-1}$ and $\mathbf{T}_k$ are independent. Therefore, the random variables $\mathbf{X}_{k,j}$ allow us to calculate the number of arrivals of random tasks from time instant 0 to time instant t. We define first the function

$$F_k(.,t): \aleph \to [0,1], \text{ where } k \neq i$$

as follows:

$$F_k(n,t) = P\left(X_{k,n-1} \le t \wedge X_{k,n} > t\right)$$

where $P(X_{k,n-1} \le t \wedge X_{k,n} > t)$ is the probability that $\mathbf{X}_{k,n} > t$ knowing that $\mathbf{X}_{k,n-1} \le t$ is true. We obtain this probability by considering, while calculating $\mathbf{X}_{k,n}$ only the values of $\mathbf{X}_{k,n-1}$ which are smaller or equal to t. Thus we consider the following restriction

$$\mathbf{X}_{k,n-1}^{x \le t}$$

of $\mathbf{X}_{k,n-1}$. Given this, we formulate $F_k(n,t)$ as follows:

$$F_k(n,t) = P\left(X_{k,n}^{x \le t} > t\right) \tag{9}$$

The random variable giving the number of arrivals at a time t and their respective probabilities is obtained using function $F_k(.,t)$ as follows:

$$\mathbf{F}_k^*(t) = \begin{pmatrix} n \\ F_k(n,t) \end{pmatrix} \tag{10}$$

To give an instantiating example, assume a task τ_k with the inter-arrival times defined by the following random variable:

$$\mathbf{T}_{k,j} = \begin{pmatrix} 8 & 12 & 16 \\ 0.2 & 0.3 & 0.5 \end{pmatrix}, \ \forall j$$

Thus, we have the following:

$$\mathbf{X}_{k,0} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}; \ \mathbf{X}_{k,1} = \begin{pmatrix} 8 & 12 & 16 \\ 0.2 & 0.3 & 0.5 \end{pmatrix}; \ \mathbf{X}_{k,2} = \begin{pmatrix} 16 & 20 & 24 & 28 & 32 \\ 0.04 & 0.12 & 0.29 & 0.3 & 0.25 \end{pmatrix}$$

According to Equation (9), and for $n = 1$, we have:

$$\begin{cases} F_k(1,12) = P\left(X_{1,1}^{x \le 12} > 12\right) = 0.5, \ \text{where} \\ \mathbf{X}_{1,1}^{x \le 12} = \begin{pmatrix} 8 & 12 & 16 \\ 0.2 & 0.3 & 0.5 \end{pmatrix} = \mathbf{T}_{k,j} \otimes \mathbf{X}_{1,0}^{x \le 12} = \mathbf{T}_{k,j} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{cases}$$

For $n = 2$, we have:

$$F_k(2,12) = P\left(X_{1,2}^{x \le 12} > 12\right) = 1 - 0.5$$

For $n > 2$, $F_k(n,2) = 0$. Thus, we get for the example the following result:

$$\mathbf{F}_k^*(12) = \begin{pmatrix} 1 & 2 \\ 0.5 & 0.5 \end{pmatrix}$$

Finally, and before concluding Section 3.

Because the function $\mathbf{F}^*_k{}^m$ ($v(\mathbf{L}_i^m)$) provide results corresponding to the time span [0, $v(\mathbf{L}_i^m)$] there is a need to "eliminate" the arrivals already taken into account at iteration $m-1$; that is, those arrivals in the time span [0, $v(\mathbf{L}_i^{m-1})$]. Therefore, we apply to $\mathbf{F}^*_k{}^m$ ($v(\mathbf{L}_i^m)$) a function `Shift` which returns a random variable having the same probabilities, but with values decreased by $n_0{}^k$:

$$\begin{cases} Shift(n_0^k, \mathbf{X}) = \begin{pmatrix} x_k - n_0^k \\ P(X = x_k) \end{pmatrix} \\ \text{where } \mathbf{X} = \begin{pmatrix} x_k \\ P(X = x_k) \end{pmatrix} \end{cases} \tag{11}$$

with $n_0{}^k$ given by:

$$\begin{cases} 0 & ;if\ m = 1 \\ \max\left\{\mathbf{F}_k^*\left(v\left(\mathbf{L}_i^m\right)\right)\right\} - \max\left\{\mathbf{F}_k^*\left(v\left(\mathbf{L}_i^{m-1}\right)\right)\right\} & ;if\ \text{these max are} \neq \\ \max\left\{\mathbf{F}_k^*\left(v\left(\mathbf{L}_i^m\right)\right)\right\} & ;if\ \text{these max are} = \end{cases} \tag{12}$$

Thus at iteration m, we consider only the random arrivals from time interval [$v(\mathbf{L}_i^{m-1})$, $v(\mathbf{L}_i^m)$].

3.4 Pseudo-code Algorithm

In this section we summarize the approach described in Sections 3.2 and 3.3 through a pseudo-code algorithm (Figure 2).

```
1: let m = 1
2: let R^θ = N_k^θ = 0
2: while R changed values and deadline not missed do
3:   for index = 1 to |ν(R_1^{m-1})| do
4:     compute each of the L_1^m with Eq. (6) with ν(R_1^{m-1})
5:     as initial value and using N_k(ν(R_1^{m-1}))
6:   end for
7:   L_1^m from the L_1^m values with probabilities as in R_1^{m-1}
8:   for all k ∈ R_hp1 do
9:     for index = 1 to |ν(L_1^m)| do
10:       compute F*_1^m using Eqs. (10) and (9)
11:       /* the F*_1^m values will be used in */
12:       /* the next iteration              */
13:       compute n_θ^k using Eqs. (12)
14:       apply the shift function as Eq. (11)
15:    end for
16:  end for
17:  compute R_1^m using Eq. (8)
18:  if R_1^m and L_1^m have common values then
19:    R_1^m = R_1^m \common values
20:  end if
21:  let m = m + 1
22: end while
```

Figure 2. Pseudo-code Algorithm of the approach.

4. NUMERICAL EXAMPLE

We will now instantiate the approach proposed in Section 3 to a task set example. The purpose is also to provide further intuition to the reader concerning the application of the analytical formulations provided.

We consider the system S of five tasks. The tasks' parameters are as shown in Table 1. For the sake of simplicity we consider that all the periodic tasks have the first release time equal to zero. We also consider that all the inter-arrival times of each random task are defined by the same random variable.

We will now elaborate the example concerning the response time analysis of task τ_4. For this particular case, we will have $P_{hpi} = \{2\}$ and $R_{hpi} = \{1,3\}$ as the sets of the periodic and random tasks with higher priority than τ_4, respectively.

In the first iteration ($m = 1$):

$$\boldsymbol{L}_4^1 = \begin{pmatrix} L_{4,1}^1 = C_4 + \left\lceil \frac{L_{4,1}^1}{T_2} \right\rceil \times C_2 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 1 \end{pmatrix}$$

Table 1. Task set example.

Task	Inter-arrival	WCET
τ_1	$\boldsymbol{T}_1 = \begin{pmatrix} 8 & 10 & 15 \\ 0.1 & 0.3 & 0.6 \end{pmatrix}$	3
τ_2	$\boldsymbol{T}_2 = \begin{pmatrix} 10 \\ 1 \end{pmatrix}$	3
τ_3	$\boldsymbol{T}_3 = \begin{pmatrix} 15 & 20 \\ 0.6 & 0.4 \end{pmatrix}$	2
τ_4	$\boldsymbol{T}_4 = \begin{pmatrix} 15 \\ 1 \end{pmatrix}$	2
τ_5	$\boldsymbol{T}_1 = \begin{pmatrix} 14 & 22 \\ 0.4 & 0.6 \end{pmatrix}$	2

We need now to compute the number of arrivals of each random task belonging to R_{hpi} in the time span $[0, L_4^1]$:

$$\begin{cases} \boldsymbol{F}_1^*\left(L_{4,1}^1\right) = \boldsymbol{F}_1^*(5) = \begin{pmatrix} n_1^1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ \boldsymbol{F}_3^*\left(L_{4,1}^1\right) = \boldsymbol{F}_3^*(5) = \begin{pmatrix} n_3^1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \end{cases}$$

Therefore,

$$\begin{aligned} \boldsymbol{R}_4^1 &= \boldsymbol{L}_4^1 \otimes \left(\otimes_{k \in R_{hpi}} Shift\left(0, \boldsymbol{F}_k^*(5)\right) \times C_k \right) = \\ &= \begin{pmatrix} 5 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} \left(n_1^1 - 0\right) \times C_1 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} \left(n_3^1 - 0\right) \times C_3 \\ 1 \end{pmatrix} = \\ &= \begin{pmatrix} 5 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \times 3 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \times 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 5+3+2 \\ 1 \end{pmatrix} = \begin{pmatrix} 10 \\ 1 \end{pmatrix} \end{aligned}$$

Before proceeding to iteration $m = 2$, we perform the `Comp` function as follows:

$$\boldsymbol{R}_4^1 = Comp\left(\boldsymbol{L}_4^1, \boldsymbol{R}_4^1\right) = \boldsymbol{R}_4^1$$

In the second iteration ($m = 2$):

$$\boldsymbol{L}_4^2 = \begin{pmatrix} L_{4,1}^2 = C_4 + \left\lceil \frac{L_{4,1}^2}{T_2} \right\rceil \times C_2 + n_1^1 \times C_1 + n_3^1 \times C_3 \\ 1 \end{pmatrix} = \begin{pmatrix} 10 \\ 1 \end{pmatrix}$$

We need now to compute the number of arrivals of each random task belonging to R_{hpi} in the time span $[0, L_4^2]$:

$$\begin{cases} \boldsymbol{F}_1^*\left(L_4^2\right) = \boldsymbol{F}_1^*(10) = \begin{pmatrix} n_{1,1}^2 & n_{1,2}^2 \\ 0.6 & 0.4 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 0.6 & 0.4 \end{pmatrix} \\ \boldsymbol{F}_3^*\left(L_4^2\right) = \boldsymbol{F}_3^*(10) = \begin{pmatrix} n_3^2 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \end{cases}$$

Using Equation (12), we obtain:

$$n_0^1 = 1;\ n_0^3 = 1$$

These values are used in the first term of the `Shift` function as follows:

$$\boldsymbol{R}_4^2 = \boldsymbol{L}_4^2 \otimes \left[Shift\left(1, \boldsymbol{F}_1^*(10)\right) \times C1\right] \otimes \left[Shift\left(1, \boldsymbol{F}_3^*(10)\right) \times C_3\right] =$$
$$= \begin{pmatrix} 10 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} \left(n_{1,1}^2 - 1\right) \times C_1 & \left(n_{1,2}^2 - 1\right) \times C_1 \\ 0.6 & 0.4 \end{pmatrix} \otimes \begin{pmatrix} \left(n_3^2 - 1\right) \times C_3 \\ 1 \end{pmatrix} =$$
$$= \begin{pmatrix} 10 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} (1-1) \times 3 & (2-1) \times 3 \\ 0.6 & 0.4 \end{pmatrix} \otimes \begin{pmatrix} (1-1) \times 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 10 & 13 \\ 0.6 & 0.4 \end{pmatrix}$$

Before proceeding to iteration m = 3, we perform the `Comp` function as follows:

$$\boldsymbol{R}_4^2 = Comp\left(\boldsymbol{L}_4^2, \boldsymbol{R}_4^2\right) = \begin{pmatrix} 13 \\ 0.4 \end{pmatrix}$$

In the third iteration (m = 3):

$$\boldsymbol{L}_4^3 = \begin{pmatrix} L_{4,1}^3 = C_4 + \left\lceil \frac{L_{4,1}^3}{T_2} \right\rceil \times C_2 + n_{1,2}^2 \times C_1 + n_3^2 \times C_3 \\ 0.4 \end{pmatrix} = \begin{pmatrix} 15 \\ 0.4 \end{pmatrix}$$

We need now to compute the number of arrivals of each random task belonging to R_{hpi} in the time span [0, L_4^3]:

$$\begin{cases} \boldsymbol{F}_1^*\left(L_4^3\right) = \boldsymbol{F}_1^*(15) = \begin{pmatrix} n_1^3 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \\ \boldsymbol{F}_3^*\left(L_4^3\right) = \boldsymbol{F}_3^*(15) = \begin{pmatrix} n_{3,1}^3 & n_{3,2}^3 \\ 0.4 & 0.6 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 0.4 & 0.6 \end{pmatrix} \end{cases}$$

Using Equation (12), we obtain:

$$n_0^1 = 2;\ n_0^3 = 1$$

These values are used in the first term of the `Shift` function as follows:

$$\boldsymbol{R}_4^3 = \boldsymbol{L}_4^3 \otimes \left[Shift\left(2, \boldsymbol{F}_1^*(15)\right) \times C1\right] \otimes \left[Shift\left(1, \boldsymbol{F}_3^*(15)\right) \times C_3\right] =$$
$$= \begin{pmatrix} 15 \\ 0.4 \end{pmatrix} \otimes \begin{pmatrix} \left(n_1^3 - 2\right) \times C_1 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} \left(n_{3,1}^3 - 1\right) \times C_3 & \left(n_{3,2}^3 - 1\right) \times C_3 \\ 0.4 & 0.6 \end{pmatrix} =$$
$$= \begin{pmatrix} 15 \\ 0.4 \end{pmatrix} \otimes \begin{pmatrix} (2-2) \times 3 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} (1-1) \times 2 & (2-1) \times 2 \\ 0.4 & 0.6 \end{pmatrix} =$$
$$= \begin{pmatrix} 15 \\ 0.4 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 2 \\ 0.4 & 0.6 \end{pmatrix} = \begin{pmatrix} 15 & 17 \\ 0.16 & 0.24 \end{pmatrix}$$

We might continue with the fourth iteration if we want to calculate more values of the response time of task τ_4. All the values after 17 are larger than the relative deadline (if we assume it to be equal to the period = 15). So, we already have the probability of meeting its deadline:

$$\begin{pmatrix} yes & no \\ 0.76 & 0.24 \end{pmatrix}$$

Given that we know:

$$\boldsymbol{R}_4 = \begin{pmatrix} 10 & 15 & \geq 17 \\ 0.6 & 0.16 & 0.24 \end{pmatrix}$$

5. ONGOING WORK

In this paper we proposed a framework for computing the response time of a fixed-priority scheduled task within a task set where the inter-arrival times of tasks may be described by discrete random variables. We are presently working on the extension of our algorithm to the case of continuous variables.

The random variables considered are assumed independent, which is eventually not a realistic assumption for some real-time systems. Research is on-going concerning addressing this issue in the context of the framework proposed in this paper.

The number of iterations of the algorithm is bounded by the value of the deadline of τ_i. Thus the algorithm presented here is polynomial in the value of the deadline. The complexity of each of the iterations depends on the complexity of calculating a convolution, and in order to decrease its complexity approaches like those presented in [8] are being investigated.

6. ACKNOWLEDGMENTS

This work was partially funded by Fundação para a Ciência e Tecnologia (FCT), under the CISTER research unit (UI608), and by the Network of Excellence on Embedded Systems Design under the IST-004527 ARTIST2 project.

7. REFERENCES

[1] L. Abeni and G. Buttazzo. QoS Guarantee Using Probabilistic Deadlines. In Proceedings of the 11th Euromicro Conference on Real-Time Systems (ECRTS99), pp. 242-249, June 1999.

[2] A. Burns, G. Bernat and I. Broster. A Probabilistic Framework for Schedulability Analysis. In the Proceedings of the 3rd International Embedded Software Conference (EMSOFT03), pp. 1-15, 2003.

[3] I. Broster and A. Burns. Applying Random Arrival Models to Fixed Priority Analysis. In the Proceedings of the Work-In-Progress of the 25th IEEE Real-Time Systems Symposium (RTSS04), 2004.

[4] J.P. Lehoczky. Real-Time Queueing Theory. In the Proceedings of the 10th IEEE Real-Time Systems Symposium (RTSS96), pp. 186-195, 1996.

[5] H. Zhu, J.P. Hansen, J.P. Lehoczky and R. Rajkumar. Optimal partitioning for Quantized EDF Scheduling. In the Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS02), pp. 202-213, 2002.

[6] J.L. Díaz, D. F. García, K. Kim, C.G. Lee, L.L. Bello, J.M. López and O. Mirabella. Stochastic Analysis of Periodic Real-Time Systems. In the Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS02), pp. 289-300, 2002.

[7] M. Joseph and P. Pandya. Finding Response Times in a Real-Time System. In The Computer Journal, Vol. 29, No. 5, pp. 390-395, 1986.

[8] M.Werman. Fast Convolution. In the Proceedings of the 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG03), 2003.