Response Times Parametric Estimation of Real-Time Systems

Kevin Zagalo, Olena Verbytska, Liliana Cucu-Grosjean, Avner Bar-Hen

Response Times Parametric Estimation of Real-Time Systems

Kevin Zagalo[1], Olena Verbytska[1], Liliana Cucu-Grosjean[1], and
Avner Bar-Hen[1,2]

[1]Inria Paris
[2]Cnam Paris

Abstract

Real-time systems are a set of programs, a scheduling policy and a system architecture, constrained by timing requirements. Most of daily embedded devices are real-time systems, *e.g.* airplanes, cars, trains, spatial probes, *etc.*. The time required by a program for its end-to-end execution is called its *response time*. Usually, upper-bounds of response times are computed in order to provide safe deadline miss probabilities. In this paper, we propose a suited re-parametrization of the inverse Gaussian mixture distribution adapted to response times of real-time systems and the estimation of deadline miss probabilities. The parameters and their associated deadline miss probabilities are estimated with an adapted Expectation-Maximization algorithm.

1 Introduction

The increasing demand for new functionalities in embedded systems like automotive, avionics and space industries is driving an increase in the performance required in embedded processors. For embedded systems with small energy and computing resources, real-time systems have a specific design with a micro-controller architecture and a set of programs (a.k.a. *task set*) running on it. An important part of this design is to associate a processing unit to a given task set. The time taken by the system to respond to an input and provide the output or display the updated information is known as the *response time*. To ensure that every task is executed within their specified timing constraints, the computation resources are allocated to different tasks according to their priority. During the run-time, each instance of the tasks compete for processing time on the basis of there priority. These sets must be *schedulable* by construction, meaning that there must exist an order (or a *schedule*) such that programs respect their timing requirements, that are called *deadlines*, in the given processing time. Timing correctness of real-time systems is traditionally guaranteed

1

by a separate schedulability analysis and a worst-case execution time analysis. Classical techniques for worst-case execution time analysis aim at finding the absolute upper bound on the execution time. After knowing this worst-case, the worst-case response time (WCRT) is computed by summing the worst-case execution times of the appropriate tasks in the worst-case scenario, *i.e.* the scenario producing the biggest response times. However, as this method is efficient to make schedulable task sets, it forces designers to over-estimate the quantity of processing unit necessary to run a task set. A way to soften resources requirements is to allow a failure rate for each task, such that the probability that a deadline is missed is bounded by this failure rate. Concentration inequalities have been widely studied these last years to bound deadline miss probabilities [6, 35, 24, 7]. Currently, the most efficient bound is the Hoeffding bound (HB) [34]. These bounds compute deadline miss probabilities only from the parameters of the studied task set. The method built in this paper uses knowledge on the task set and infers response time data to compute the maximum likelihood estimate (MLE).

Furthermore, execution times may correspond to various state of a real-time system (see [25, 28] among others). Statistical approaches have been also used to detect mode changes within the functioning of time critical embedded systems. Such detection serves a higher-level objective: characterizing a functional mode that may be a normal, exceptional, functional or degraded, in order to increase the reactivity of these systems and to predict mode transitions. Indeed, by adapting the reaction of the system with respect to a given mode, an optimized utilization of resources is possible, which becomes another commercial trend within the time critical embedded systems industry. Sometime the state is explicit, such as a drone in a take-off mode for example, but tasks often depend of unobserved latent variables such as environmental variables. Under smooth hypotheses, we justify in Section 2 that response times are first-passage times of Brownian motions, and hence the inverse Gaussian (IG) family is the natural family for response time approximations. The IG family is a natural choice for a statistical modelling of positive and right-skewed distributions, see [12, 33]. It is used in many fields, such as industrial degradation modelling [36], psychology [31, 23], and many others like hydrology, market research, biology, ecology, and so on *c.f.* [32].

In our case, we propose a suited parametrization of the IG distribution in Section 3 after introducing real-time systems in Section 2. Using an adapted Expectation-Maximization (EM) algorithm, we estimate the parameters of a mixture of IG distributions. This allows to estimate response times without using the extreme value approach and provide parametric scheduling knowledge. Finally in Section 4.1 we show the rate of convergence of the algorithm and compare it to the Hoeffding bound (see Lemma 2) with simulations, compare it to the classic EM algorithm in terms of computation time, and apply to real data.

2 Real-time systems

In this paper we consider *periodic* tasks, meaning that an instance of each task is periodically released at a given rate, and a single core system, *i.e.* only one task is processed at a time.

2.1 Model

Let us consider a single core real-time system composed of a finite task set $\Gamma = \{\tau_1, \ldots, \tau_\gamma\}$. A task τ_i is characterized by:

- its *execution time* $C_i > 0$,

- its *inter-arrival time* $t_i > 0$,

- its *permitted failure rate* $\alpha_i \in [0, 1)$.

The studied scheduling policy in this paper is the *Rate Monotonic* (RM) policy. RM assigns higher priorities to smaller periods, *i.e.* the task τ_i will stop (*i.e.* preempt) any running task τ_j with $j > i$ to execute itself if needed, if and only if $t_i < t_j$. We consider Γ ordered by decreasing priority order and scheduled with the RM policy. RM is optimal for real-time systems using fixed priorities and implicit deadlines [2].

The j-th instance of the task τ_i is called a *job* and we denote it $\tau_{i,j}$. Its execution time is denoted $C_{i,j}$. We assume that execution times of a given task τ_i are *i.i.d.* with a probability function f_i of positive and discrete support $\{c_i^{min}, \ldots, c_i^{max}\}$, and with mean m_i and standard deviation s_i.

Let the *mean utilization of level i* be $u_i = \sum_{j=1}^{i} m_j/t_j$ and the *deviation of level i* be $v_i = (\sum_{j=1}^{i} s_j^2/t_j)^{1/2}$.

2.2 Response times

Formally, the *response time* $R_{i,j}$ of a job $\tau_{i,j}$ is the elapsed time between its arrival time $a_{i,j} = (j-1)t_i$ and the end of its execution with the RM policy, see Figure 1.

In [38, 14], authors prove that when the utilization u_i is smaller than 1, there exists a time t^* such that the response times of jobs of level i released after t^* are stationary, *i.e.* the sequence $\{R_{i,j} : a_{i,j} > t^*\}$ is identically distributed. Before t^* the system is said transient. We treat in this paper the transient and stationary response time with a mixture model and estimate its deadline miss probability. In order to find the probability density function of this response time, the arrival of jobs is modeled with a $D/G/1$ queue [38], and an approximation of response times is determined by using the *heavy-traffic assumption* [17, 11] *c.f.* Figure 2, which permits to approximate the size of a queue with a Brownian motion. $R_{i,j}$ is modeled as the *first-passage time* of a Brownian motion of drift $u_i - 1$ and deviation v_i, which is known to follow an IG distribution [3, 29]. The IG
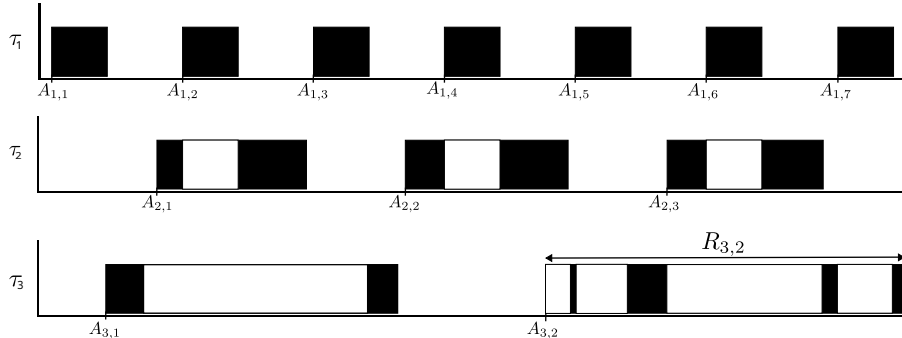
Figure 1: Example of a fixed-priority schedule. The higher priority tasks stop the execution of tasks if needed. Thus, the response time is time between the release and the end of a job, taking the jobs of higher priority tasks into account.

probability density function ψ is defined by

$$\psi(x; \xi, \delta) = \sqrt{\frac{\delta}{2\pi x^3}} \exp\left(\frac{\delta(x-\xi)^2}{2x\xi^2}\right), x \geq 0 \tag{1}$$

where $\xi > 0$ corresponds to the mean and and $\delta > 0$ is called the shape.

In order for a task $\tau_i \in \Gamma$ to be *schedulable*, its *deadline miss probability* $\Delta_i = \sup_j \mathbf{P}\left(R_{i,j} > t_i\right)$ [6] should be lower than its permitted failure rate $\alpha_i \in [0, 1)$, *i.e.*

$$\Delta_i \leq \alpha_i \tag{2}$$

thus finding an analytical expression of the probability density function of response times permits to determine if a task is schedulable or not. Furthermore, the estimation must provide in the worst-case bigger quantiles than the measurements, so that the inequality (2) is satisfied.

In [38], authors prove that if $u_i < 1$, there exists a distribution $P_{i,j}$ that depends on f_i and the arrival time $a_{i,j}$ such that the probability density function of $R_{i,j}$ is

$$h_{i,j}(x) = \int_0^\infty \psi_{i-1}(x; \theta) dP_{i,j}(\theta), x \geq 0 \tag{3}$$

where $\psi_i(x; \theta)$ is the probability density function of the IG distribution of mean $\theta/(1 - u_i)$ and shape $(\theta/v_i)^2$. The parameter θ is linked to the backlog of the associated $D/G/1$ queue used to model the arrival of jobs.

This representation of response times is accurate, but it is a very expensive task to compute $dP_{i,j}(\theta)$ for each $\theta > 0$ and each $a_{i,j}$ at each decision of the scheduling algorithm. Moreover, the distribution $P_{i,j}$ depends on the fact that f_i is known, which is not always the case. Therefore, in practice we estimate $P_{i,j}$. We find in the next section the appropriate parametric estimation of the probability density function h_i, using only the utilization u_i and the deviation v_i, which only use the mean and variance of C_i.
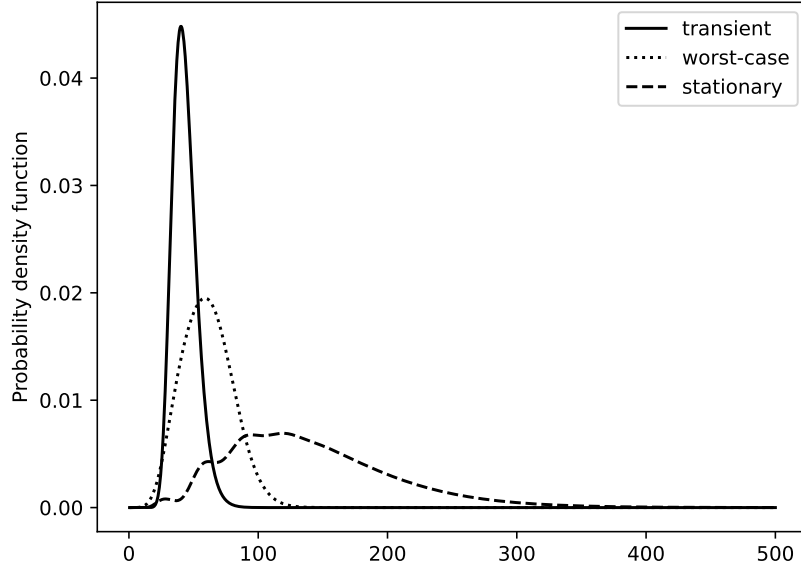
4

Figure 2: Example of the MLE of the transient response time, and the stationary and worst-case response time distribution.

3 IG mixture model for response times

Let R_i be the response time of τ_i. Its distribution function is the mixture of the distribution functions of the response times $R_{i,j}$. The distribution of R_i is composed of k_i components. Formally, this means that we approximate the probability density function of the response time R_i with a the variable R_i of probability density function

$$h_i(x; \boldsymbol{\pi}_i, \boldsymbol{\theta}_i) = \sum_{k=1}^{k_i} \pi_{i,k} \psi_{i-1}(x; \theta_{i,k}) \tag{4}$$

In real-time systems, the interest of the analytical approach is to measure the deadline miss probability Δ_i with a closed expression. For example, a task τ_i should not miss its deadline with a permitted failure rate α_i, and the inequality in (2) is approximated with the mixture (4).

3.1 Re-parameterized IG distribution for response times

The purpose of this section is to provide the efficient distribution family for an approximation of response times and an adapted EM algorithm to estimate the

parameters of this approximation. This adapted re-parametrization of the IG distribution reduces the number of parameters of the mixture model. Furthermore, as underlined in [26], the log-likelihood of the IG distribution has flat regions thus the EM algorithm has very tiny variations. Reducing the number of parameters addresses a part of this problem. A second reduction of this problem is the use of the Aitken acceleration procedure [1].

In [26], the author introduces a modified version of the IG distribution of parameters (ξ, δ), using its mode $\mu = (\xi^2 + (3\xi^2/2\delta)^2)^{1/2} - 3\xi^2/2\delta$ and its *variability coefficient* $\lambda = \xi^2/\delta$ instead of the mean and shape. This re-parameterized IG distribution (rIG) of parameters (μ, λ) is defined by the probability density function

$$\tilde{\psi}(x; \mu, \lambda) = \sqrt{\frac{\mu(3\lambda + \mu)}{2\pi\lambda x^3}} \exp\left\{-\frac{\left(x - \sqrt{\mu(3\lambda + \mu)}\right)^2}{2\lambda x}\right\} \tag{5}$$

With the rIG distribution applied to the form that take the parameters of the distributions of response times, one can see that only the mode is sensitive to the mixture provided in (3). The variability coefficient of an IG distribution of mean $\theta/(1 - u_i)$ and shape $(\theta/v_i)^2$ is

$$\lambda_i = \frac{v_i^2}{(1 - u_i)^2} \tag{6}$$

and its mode is

$$\mu_i(\theta) = \sqrt{\left(\frac{\theta}{1 - u_i}\right)^2 + \frac{9\lambda_i^2}{4}} - \frac{3\lambda_i}{2} \tag{7}$$

so that $\psi_i(x; \theta)$ is the probability density function of an rIG distribution of mode $\mu_i(\theta)$ and variability λ_i.

3.2 Maximum likelihood estimation of response time distributions

In this section we present an adaptation of the maximum likelihood estimator (MLE) proposed by [26] for real-time systems $(\boldsymbol{\pi}_i, \boldsymbol{\theta}_i)$. Both are implemented in the Python language in the library rInverseGaussian [39].

When $k_i = 1$, $\pi_1 = 1$ and we have the MLE

$$\hat{\theta}_i = (1 - u_{i-1})\sqrt{\bar{R}_i(\bar{R}_i + 3\lambda_{i-1})} \tag{8}$$

where $\bar{R}_i = \frac{1}{n}\sum_{j=1}^{n} R_{i,j}$, see [12]. The complete-likelihood of mixture models [19, 5] can be written as

$$L_c(\boldsymbol{Z}_i, \boldsymbol{\pi}_i, \boldsymbol{\theta}_i) = \prod_{j=1}^{n}\prod_{k=1}^{k_i} [\pi_{i,k}\psi_{i-1}(r_j; \theta_{i,k})]^{Z_{i,j,k}} \tag{9}$$

6

and the complete log-likelihood $\ell_c = \log L_c$ is

$$\ell_c(\boldsymbol{Z}_i, \boldsymbol{\pi}_i, \boldsymbol{\theta}_i) = \ell_{c_1}(\boldsymbol{Z}_i, \boldsymbol{\pi}_i) + \ell_{c_2}(\boldsymbol{Z}_i, \boldsymbol{\theta}_i) \tag{10}$$

where

$$\ell_{c_1}(\boldsymbol{Z}_i, \boldsymbol{\pi}_i) = \sum_{j=1}^{n} \sum_{k=1}^{k_i} Z_{i,j,k} \log \pi_{i,k} \tag{11}$$

and

$$\ell_{c_2}(\boldsymbol{Z}_i, \boldsymbol{\theta}_i) = \sum_{j=1}^{n} \sum_{k=1}^{k_i} Z_{i,j,k} \log \psi_{i-1}(r_j; \theta_{i,k}) \tag{12}$$

which leads to the following EM algorithm:

E-step   For the $(s + 1)$th step of the EM algorithm, $\boldsymbol{z}_i^{(s)}$ the conditional expectation of $\boldsymbol{Z}_i$ given $(\boldsymbol{\pi}_i, \boldsymbol{\theta}_i) = \left(\boldsymbol{\pi}_i^{(s)}, \boldsymbol{\theta}_i^{(s)} \right)$ is given by

$$z_{i,j,k}^{(s)} = \frac{\pi_{i,k}^{(s)} \psi_{i-1}\left(r_j; \theta_{i,k}^{(s)} \right)}{h_i \left(r_j; \boldsymbol{\pi}_i^{(s)}, \boldsymbol{\theta}_i^{(s)} \right)} \tag{13}$$

M-step   For the $(s + 1)$th step of the EM algorithm, $\ell_{c_1}(\boldsymbol{z}_i^{(s)}, \cdot)$ is maximized by

$$\pi_{i,k}^{(s+1)} = \frac{1}{n} \sum_{j=1}^{n} z_{i,j,k}^{(s)}, k = 1, \ldots, k_i \tag{14}$$

and maximizing ℓ_{c_2} with respect to $\boldsymbol{\theta}$ is maximizing each of the k_i expressions

$$\sum_{j=1}^{n} z_{i,j,k}^{(s)} \log \psi_{i-1}(r_j; \theta_{i,k}), k = 1, \ldots, k_i \tag{15}$$

using Newton-like algorithms to solve

$$\nabla \ell_c = 0 \tag{16}$$

Then with $\frac{\partial \log \psi_i}{\partial \theta}(x; \theta) = \frac{\partial \mu_i}{\partial \theta}(\theta) \frac{\partial \log \tilde{\psi}}{\partial \mu}(x; \mu_i(\theta), \lambda_i)$ and the derivatives

$$
\begin{aligned}
\frac{\partial \log \tilde{\psi}}{\partial \mu}(x; \mu, \lambda) &= -\frac{3}{2x} - \frac{\mu}{x\lambda} + \frac{1}{3\lambda + \mu} + \frac{3\lambda}{2\mu(3\lambda + \mu)} + \frac{\sqrt{\mu}}{2\lambda\sqrt{3\lambda + \mu}} + \frac{\sqrt{3\lambda + \mu}}{2\lambda\sqrt{\mu}} \\
\frac{\partial \mu_i}{\partial \theta}(\theta) &= \frac{\theta}{(1 - u_i)^2} \left(\left(\frac{\theta}{1 - u_i} \right)^2 + \frac{9\lambda_i^2}{4} \right)^{-1/2}
\end{aligned}
\tag{17}
$$

(16) is equivalently solved by (14) and the solutions of

$$\sum_{j=1}^{n} z_{i,j,k}^{(s)} \frac{\partial \log \psi_{i-1}}{\partial \theta}(r_j; \theta_k) = 0, \forall k = 1, \ldots, k_i \tag{18}$$

7

In order to stop the algorithm, the author in [26] proposes the *Aitken acceleration* to stop the algorithm. The Aitken acceleration at iteration $s + 1$ is given by

$$a^{(s+1)} = \frac{\ell^{(s+2)} - \ell^{(s+1)}}{\ell^{(s+1)} - \ell^{(s)}} \tag{19}$$

where $\ell^{(s)}$ is the observed-data log-likelihood from iteration s. The limit ℓ_∞ of the sequence of values of the log-likelihood is

$$\ell_\infty^{(s+2)} = \ell^{(s+1)} + \frac{\ell^{(s+2)} - \ell^{(s+1)}}{1 - a^{(s+1)}} \tag{20}$$

The EM algorithm is considered to have converged if

$$|\ell_\infty^{(s+2)} - \ell_\infty^{(s+1)}| < \varepsilon \tag{21}$$

with a tolerance $\varepsilon > 0$.

Finally, we initialize the algorithm with a k-means clustering.

3.3   Bayesian information criteria

The Bayesian information criteria (BIC, [30]) is used to chose the number of components of the mixture, which has been proven consistent for mixture models [27, 13, 10]. The number of parameters of a mixture of k components being $2k - 1$, the number of components chosen is equal to

$$k_i = \underset{k}{\operatorname{argmax}} \, 2\ell_n(\boldsymbol{\pi}_i, \boldsymbol{\theta}_i) - (2k - 1) \log n \tag{22}$$

where ℓ_n is the observed-data log-likelihood. The number of parameters being reduced from $3k_i - 1$ to $2k_i - 1$, the computation time of this EM algorithm is also reduced.

3.4   Deadline miss probability

We use the link of IG distributions with the χ_1^2 distribution to check the quality of the MLE. Indeed, if X is an IG variable of mean ξ and shape δ, then $\delta(X - \xi)^2 / \xi^2 X$ is distributed as a Chi-squared distribution of one degree of freedom [33]. Let $\mathbf{P}_k^{\mathrm{IG}}$ be the probability conditionally that the response time R_i is in the k-th component in the IG estimation, and

$$g_i(x; \theta) = \frac{\left(x - \frac{\theta}{1 - u_{i-1}}\right)^2}{\lambda_{i-1} x}, \, x > 0 \tag{23}$$

In our case, for each component $k = 1, \ldots, k_i$ of the mixture (4), after classification we should have that

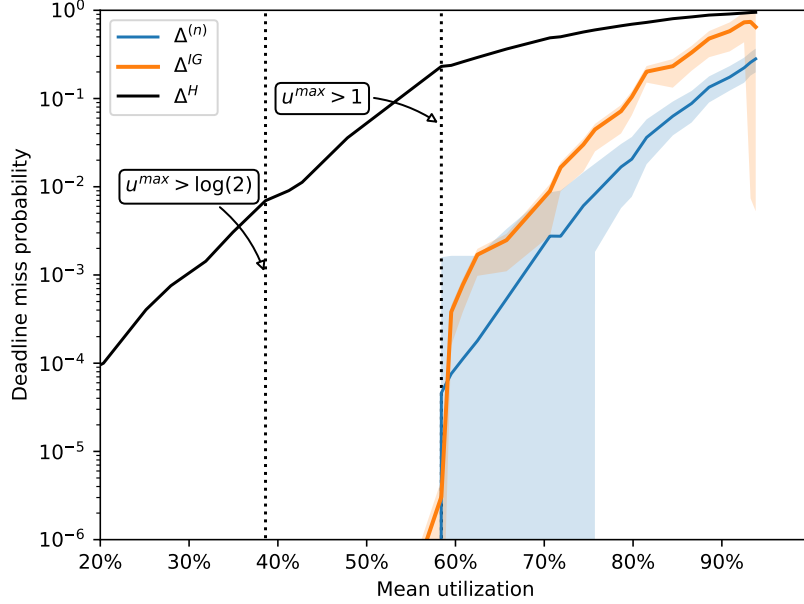$$g_i\left(R_i; \hat{\theta}_{i,k}\right) \sim \chi_1^2 \tag{24}$$

8

Figure 3: Utilizations u_i against the deadline miss probabilities of the IG estimation Δ_i^{IG}, the Hoeffding bound Δ_i^{H} and the empirical deadline miss probabilities $\Delta_i^{(n)}$ over a sample of $n = 10\,000$ response times per task simulated on SimSo.

under the probability $\mathbf{P}_k^{\mathrm{IG}}$. Therefore, we use (24) to validate the MLE, and provide the deadline miss probabilities we are looking for. Note that the larger quantiles values are the ones that real-time designers are interested in to determine whether a task is schedulable or not, see (2). We use (24) to determine whether a task is schedulable in its transient state or not.

Proposition 1. *The deadline miss probability of the IG estimation is*

$$\Delta_i^{\mathrm{IG}} = \sum_{k=1}^{k_i} \pi_{i,k} |\mathbf{1}_{\{t_i > \frac{\theta_{i,k}}{1-u_{i-1}}\}} - \chi_1^2 \left(g_i(t_i; \theta_{i,k})\right)| \tag{25}$$

where χ_1^2 is the cumulative distribution function of the Chi-squared distribution of one degree of freedom.

Proof. By unconditioning we have $\Delta_i^{\mathrm{IG}} = \sum_{k=1}^{k_i} \pi_{i,k} \mathbf{P}_k^{\mathrm{IG}}(R_i > t_i)$ and since $g_i(\cdot; \theta)$ is positive and, decreasing for $x \le \frac{\theta}{1-u_{i-1}}$ and increasing for $x > \frac{\theta}{1-u_{i-1}}$, we get the result. $\square$

See in Figure 3 a comparison between the empirical deadline miss probabilities, the IG method in (25) and the Hoeffding bound.

Proposition 2. *Let Γ be a task set as defined in Section 2.1 scheduled with the RM policy. Suppose $u_i < 1$, and let $v_i^{max} = \sum_{j=1}^{i}(c_j^{max} - c_j^{min})^2/t_j$ and*

$$\Delta_i^{\mathrm{H}} = \exp\left(-t_i\frac{(1-u_i)^2}{v_i^{max}}\right) \tag{26}$$

be the Hoeffding bound. If $t_i > \frac{\sum_{j=1}^{i} m_j}{1-u_i}$, then $\Delta_i \leq \Delta_i^{\mathrm{H}}$.

Proof. According to [34, Theorem 6], the Hoeffding inequality applied to a fixed-priority policiy gives us

$$\Delta_i \leq \inf_{\substack{t\in(0,t_i)\\ t>\mathbf{E}[W_i(t)]}} \exp\left(-2\frac{(t-\mathbf{E}[W_i(t)])^2}{\sum_{j=1}^{i}(c_j^{max}-c_j^{min})^2 n_j(t)}\right) \tag{27}$$

where $n_j(t) = \lceil t/t_j \rceil$ is the number of jobs of the task τ_j released before $t > 0$, and $W_i(t) = C_i + \sum_{j=1}^{i-1}\sum_{k=1}^{n_j(t)} C_{j,k}$.

Let $t \in (0, t_i)$. Since only one job of τ_i is released before t_i, we have $W_i(t) = \sum_{j=1}^{i}\sum_{k=1}^{n_j(t)} C_{j,k}$ and $\mathbf{E}[W_i(t)] = \sum_{j=1}^{i} n_j(t)m_j$. Since

$$t/t_j \leq n_j(t) \leq t/t_j + 1 \tag{28}$$

and $u_i < 1$, we have the relation $u_i t + \sum_{j=1}^{i} m_j \geq \mathbf{E}[W_i(t)] \geq u_i t$. Hence, $t > \mathbf{E}[W_i(t)]$ if $t > \frac{\sum_{j=1}^{i} m_j}{1-u_i}$.

Suppose $t_i > \frac{\sum_{j=1}^{i} m_j}{1-u_i}$ and $t \in (\frac{\sum_{j=1}^{i} m_j}{1-u_i}, t_i)$. With (28) we get

$$\frac{(t-\mathbf{E}[W_i(t)])^2}{\sum_{j=1}^{i}(c_j^{max}-c_j^{min})^2 n_j(t)} \geq \frac{t(1-u_i)^2}{v_i^{max} + t^{-1}\sum_{j=1}^{i}(c_j^{max}-c_j^{min})^2} \tag{29}$$

Finally the infinimum in (27) is reached for $t = t_i$, and we are using the RM policy, thus we have $t_j \leq t_i$ for $j \leq i$, hence we get

$$v_i^{max} + t_i^{-1}\sum_{j=1}^{i}(c_j^{max}-c_j^{min})^2 \leq 2v_i^{max} \tag{30}$$

which gives us the result with (29).

$\square$

In the following, we test with simulation if Δ_i^{IG} is a good estimation of Δ_i and if the Hoeffding bound is a safe bound of the IG estimation, *i.e.* $\Delta_i^{\mathrm{IG}} \leq \Delta_i^{\mathrm{H}}$.

4 Experimental results

The seminal work of Liu and Layland [18] provides a sufficient condition for the schedulability of any system with finite supports of execution times using the

maximal utilization of the priority level i $u_i^{max} = \sum_{j=1}^{i} c_j^{max}/t_j$, where c_i^{max} is the worst-case execution time, *i.e.* the maximum value of the support of the execution time C_i. Whenever

$$u_\gamma^{max} < \gamma(2^{1/\gamma} - 1) \tag{31}$$

the task set Γ is proven schedulable for $\alpha_1 = \cdots = \alpha_\gamma = 0$ [18, Theorem 5]. The bound usually used is $\lim_{\gamma \to \infty}^{\uparrow} \gamma(2^{1/\gamma} - 1) = \log(2)$. Moreover, while $u_i^{max} < 1$ there exists a scheduling policy, with dynamic priorities that can satisfy the schedulability of the task τ_i [18]. Hence there are two phase transitions, one at $u_i^{max} > \log(2)$ where deadline misses *can* happen, and one at $u_i^{max} > 1$ where deadline misses *must* happen. As proven in [38], the necessary condition for the schedulability of a task τ_i (2) is that the mean utilization u_i is less than one. Hence, there is a gap to fill in the theory between the necessary condition $u_i < 1$ and the sufficient condition $u_i^{max} < \log(2)$. In particular in the case where $u_i < 1$ and $u_i^{max} > 1$ as we see in Figure 3.

4.1 Simulations

In this section, we apply our results with simulated data. The simulated data is generated using SimSo [9], a Python framework used to generate arrivals of jobs and scheduling policies. A modified version of SimSo [8] allows to generate random inter-arrival times and random execution times [37][1]. We study the quality of the estimation as a function of utilization level. We show that the utilization, the better the estimation. We also check how much the IG bound is larger than the empirical deadline miss probabilities,

$$\Delta_i^{(n)} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}_{R_{i,j} > t_i} \tag{32}$$

Consider a task set where the probability density functions $(f_i)_i$ of the execution times $(C_i)_i$ are known. From SimSo we generate the response times of tasks with the RM scheduling policy from the probability functions $(f_i, i = 1, \ldots, 28)$. Their parameters are given in Table 1. The distributions of execution times used in the simulations are generated with UUnifast [4], to emphasize the fact that f_i can be any distribution ($D/G/1$ queue). Two methods are used: one with a finite support where the maximal utilization u_i^{max} is finite, and another one with an infinite support with exponential distributions where the maximal utilization is not defined. This schedule is instantiated 100 times, thus in Figure 4(a), the box-plots of each task are based on 100 estimators. Also note in Figure 4(a) that the variability of the estimates decreases with the priority level. Because of the fixed-priority structure of RM, we can see in Figure 4 that the error of the estimation decreases with the priority level. The first task is never pre-empted, so its response time is always equal to its execution time. Therefore the estimation of its response time cannot be good in general.

---

[1]`https://github.com/kevinzagalo/simso/blob/main/generator/task_generator.py`
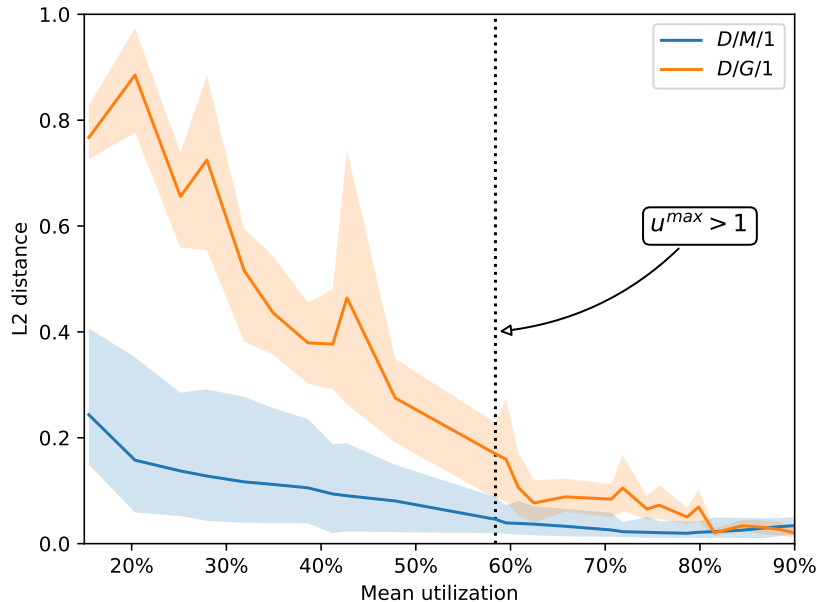
Figure 4: L2 distance between the empirical distribution of the simulations and the MLE distribution, of 100 instances of the schedule, for the task set shown in Table 1.

In a second step, a task set with exponentially distributed execution times is simulated for comparison ($D/M/1$ queue), as it is a special case widely studied in queueing theory [22]. This is a baseline for determining the rate of convergence of the response times estimation as a function of the priority levels. This baseline confirms that the rate of convergence depends on the type of distributions used for execution times, but that there is a phase transition at $u_i^{max} > 1$, independent from the type of distribution used for execution times. The parameters of the task set are given in Table 1.

In Figure 3, we have the mean utilizations $(u_i)_i$ on the x-axis and $(\Delta_i^{(n)}, \Delta_i^{\mathrm{H}}, \Delta_i^{\mathrm{IG}})_i$ on the y-axis. We can see that when $u_i^{max} < \log(2)$, it is useless to compare the methods because they are already proven schedulable in (31). Moreover, all methods start increasing when $u_i^{max} > 1$.

Table 1: Parameters of the task set used for the simulations in Section 4.1.

Task τ_i	0	1	2	3	4	5	6	7	8	9
Mean m_i (ms)	15.481	5.556	5.708	3.38	5.198	4.057	4.998	3.786	2.167	7.453
Std s_i (ms)	17.1957	5.6996	5.9963	3.323	5.5314	4.0299	5.1184	3.9005	1.8259	8.3373
Periods t_i (ms)	100	114	119	121	132	133	136	144	145	146
Deadlines d_i (ms)	100	114	119	121	132	133	136	144	145	146
Mean utilisation u_i	0.1548	0.2035	0.2515	0.2794	0.3188	0.3493	0.3861	0.4124	0.4273	0.4784
Maximum utilization u_i^{max}	0.28	0.3589	0.443	0.4926	0.5683	0.6285	0.702	0.7506	0.7713	0.874

Task τ_i	10	11	12	13	14	15	16	17	18	19
Mean m_i (ms)	16.812	1.833	2.167	2.7	5.448	8.46	2.167	4.665	2.4	5.604
Std s_i (ms)	19.1248	1.5271	1.8259	2.4495	5.4483	9.2277	1.8259	4.7411	2.2361	5.7741
Periods t_i (ms)	159	165	165	165	166	173	181	182	183	191
Deadlines d_i (ms)	159	165	165	165	166	173	181	182	183	191
Mean utilisation u_i	0.5841	0.5952	0.6083	0.6247	0.6575	0.7064	0.7184	0.744	0.7571	0.7865
Maximum utilization u_i^{max}	1.0879	1.1061	1.1242	1.1485	1.2027	1.301	1.3175	1.3615	1.3834	1.4357

Task τ_i	20	21	22	23	24	25	26	27	28
Mean m_i (ms)	2.333	3.334	5.927	4.535	4.225	6.246	4.779	2.167	1.834
Std s_i (ms)	1.9147	3.0555	6.0701	4.4073	4.1931	6.758	4.8654	1.8259	1.4149
Periods t_i (ms)	193	200	201	214	215	268	296	298	315
Deadlines d_i (ms)	193	200	201	214	215	268	296	298	315
Mean utilisation u_i	0.7986	0.8152	0.8447	0.8659	0.8856	0.9089	0.925	0.9323	0.9381
Maximum utilization u_i^{max}	1.4513	1.4763	1.531	1.5637	1.6009	1.6494	1.6764	1.6865	1.696

4.2 Data

In this section we use the IG method on real data. We use a real case of 9 programs of an autopilot of a drone, PX4-RT [21], a modified version of PX4 [20] with a real-time behavior, and a clock measuring preempting the operating system itself. PX4-RT is run on an ARM Cortex M4 CPU clocked at 180 MHz with 256 KB of RAM using a simulated environment from Gazebo [16]. PX4-RT allows to measure execution times (Figure 5 and Table 2) and response times during the flight of a drone. It runs on top of NuttX, a Unix-like operating system. It provides an infrastructure for internal communications between all programs and off-board applications. Each task is a NuttX task launched at the beginning of the PX4 program. The tasks read data from sensors (snsr), estimate positions and attitudes using a Kalman filter (ekf2), control the position (pctl) and the attitude (actl) of the drone, the flight manager (fmgr), the hover thrust estimator (hte), handle the navigation (navr), command the state of the drone (cmdr), and the rate controller (rctl), which is the inner-most loop to control the body rates. These tasks are in constant interference with the operating system NuttX. Because the operating system has the highest priority, the nine tasks studied are constantly preempted by NuttX. Unfortunately, it is not possible to have information about the interfering operating system programs. Unlike the simulation in Section 4.1, PX4-RT runs concurrently with other tasks which do not have timing requirements, making it a complex system with many unknown variables. We test in this section whether and when the proposed parametric estimation is suitable for such complex system.

In this case, the distribution functions of execution times cannot be provided. Therefore, we use the empirical distributions shown in Figure 5. Thus, the mean utilizations $\hat{u}_i$ are computed with the empirical means of execution times, and the maximal utilizations $\hat{u}_i^{max}$ with the empirical maximum of execution times. See Table 2 for a full description of the parameters. As shown in the previous section, the response times of the highest priority task *snsr* are not estimated.

These programs generate response times shown in Figure 6, on which we use the mixture model proposed in (4) with the EM algorithm provided in Section 3.2. See [15] for a full description of the data. The QQplots in Figure 6 show that the estimation is good for the large quantiles, which is what is important to determine the schedulability of a system, *c.f.* (2). We can identify in Figure 6 that for the *cmdr* and *fmgr* tasks the estimation is not good enough, which means that we have too little information about the programs interfere with them (operating system etc.), and that a schedulability test on this task would not be suitable with the method built in this paper. Nevertheless, for the other tasks the approximation is good and can therefore be used for a schedulability analysis.
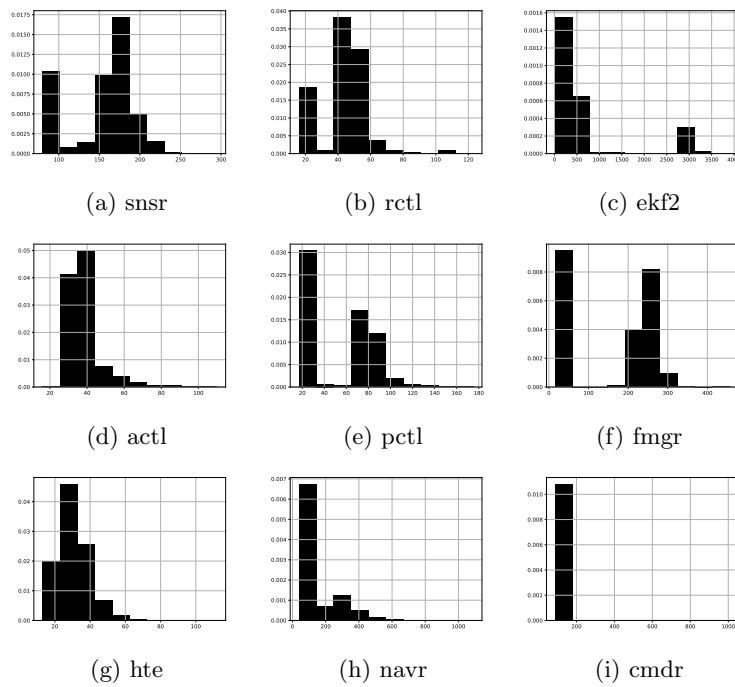
14

(a) snsr (b) rctl (c) ekf2

(d) actl (e) pctl (f) fmgr

(g) hte (h) navr (i) cmdr

Figure 5: Execution time empirical probability functions of the 9 studied tasks of the drone autopilot.

15

(a) rctl

(b) ekf2

(c) actl

(d) pctl
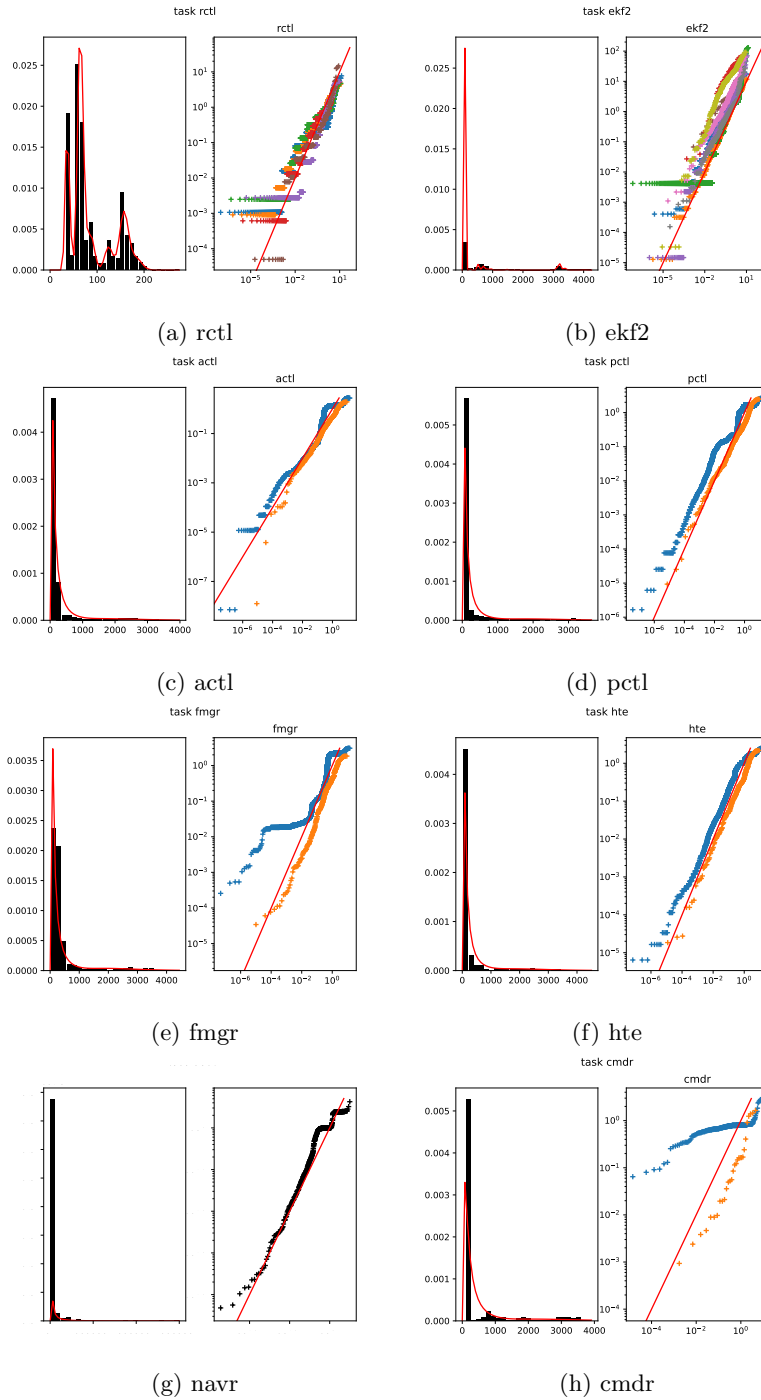
(e) fmgr

(f) hte

(g) navr

(h) cmdr

Figure 6: Response times empirical distributions of the PX4-RT autopilot and QQplots with the χ_1^2 quantiles from (24) for each component of the estimated mixtures.

16

Table 2: Empirical parameters, periods and deadlines used in the autopilot PX4-RT described in Section 4.2.

Task	snsr	rctl	ekf2	actl	pctl	fmgr	hte	navr	cmdr
Priority i	1	2	3	4	5	6	7	8	9
Number of components k_i	–	6	9	2	1	2	2	2	1
Empirical mean $\hat{m}_i$ (μs)	152.5	44.2	610.9	38.7	52.5	151.0	30.61	159.8	114.4
Empirical std $\hat{s}_i$ (μs)	40.5	16.4	982.8	9.5	32.6	114.4	10.9	133.9	61.6
Period t_i (ms)	3.0	4.0	4.1	5.0	5.2	6.0	7.0	50.0	100.0
Deadline d_i (ms)	3.0	4.0	4.1	5.0	5.2	6.0	7.0	50.0	100.0
Empirical mean utilization $\hat{u}_i$	0.05	0.06	0.21	0.22	0.23	0.25	0.26	0.26	0.26
Empirical maximum utilization $\hat{u}_i^{max}$	0.10	0.13	1.08	1.10	1.14	1.21	1.23	1.25	1.26
Empirical deadline miss probability $\Delta_i^{(n)}$	0.0	0.0	0.001	0.0	0.0	0.0	0.0	0.0378	0.0
IG deadline miss probability Δ_i^{IG}	-	0.0	0.0006	0.0043	0.028	0.0022	0.0009	0.5487	0.0
Hoeffding bound Δ_i^{H}	-	10^{-168}	0.2512	0.1919	0.1881	0.1673	0.1274	10^{-7}	10^{-13}

References

[1] AC Aitken. Iii.—a series formula for the roots of algebraic and transcendental equations. *Proceedings of the Royal Society of Edinburgh*, 45(1):14–22, 1926.

[2] Neil C Audsley, Alan Burns, Mike F Richardson, and Andy J Wellings. Hard real-time scheduling: The deadline-monotonic approach. *IFAC Proceedings Volumes*, 24(2):127–132, 1991.

[3] Ole Barndorff-Nielsen, Preben Blæsild, and Christian Halgreen. First hitting time models for the generalized inverse gaussian distribution. *Stochastic Processes and their Applications*, 7(1):49–54, 1978.

[4] Enrico Bini and Giorgio C Buttazzo. Measuring the performance of schedulability tests. *Real-Time Systems*, 30(1):129–154, 2005.

[5] Charles Bouveyron, Gilles Celeux, T Brendan Murphy, and Adrian E Raftery. *Model-Based Clustering and Classification for Data Science: With Applications in R*, volume 50. Cambridge University Press, , 2019.

[6] Kuan-Hsun Chen and Jian-Jia Chen. Probabilistic schedulability tests for uniprocessor fixed-priority scheduling under soft errors. In *2017 12th IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 1–8, 2017.

[7] Kuan-Hsun Chen, Georg Von Der Brüggen, and Jian-Jia Chen. Analysis of deadline miss rates for uniprocessor fixed-priority scheduling. In *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 168–178. IEEE, 2018.

[8] Maxime Cheramy. Simso.

[9] Maxime Chéramy, Pierre-Emmanuel Hladik, and Anne-Marie Déplanche. Simso: A simulation tool to evaluate real-time multiprocessor scheduling algorithms. In *Proc. of the 5th International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems*, WATERS, 2014.

[10] Abhijit Dasgupta and Adrian E Raftery. Detecting features in spatial point processes with clutter via model-based clustering. *Journal of the American statistical Association*, 93(441):294–302, 1998.

[11] Bogdan Doytchinov, John Lehoczky, and Steven Shreve. Real-time queues in heavy traffic with earliest-deadline-first queue discipline. *Annals of Applied Probability*, pages 332–378, 2001.

[12] J Leroy Folks and Raj S Chhikara. The inverse gaussian distribution and its statistical application—a review. *Journal of the Royal Statistical Society: Series B (Methodological)*, 40(3):263–275, 1978.

[13] Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association*, 97(458):611–631, 2002.

[14] Augustus Josephus Elizabeth Maria Janssen and JSH Van Leeuwaarden. Relaxation time for the discrete d/g/1 queue. *Queueing Systems*, 50(1):53–80, 2005.

[15] Marwan Wehaiba El Khazen, Kevin Zagalo, Hadrien Clarke, Mehdi Mezouak, Yasmina Abdeddaïm, Avner Bar-Hen, Slim Ben Amor, Rihab Bennour, Adriana Gogonel, Kossivi Kougblenou, Yves Sorel, and Liliana Cucu-Grosjean. Work in progress: Kdbench - towards open source benchmarks for measurement-based multicore wcet estimators. In *2022 IEEE 28th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 309–312, 2022.

[16] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2149–2154. IEEE, 2004.

[17] John P Lehoczky. Real-time queueing theory. In *17th IEEE Real-Time Systems Symposium*, pages 186–195. IEEE, 1996.

[18] Chung Laung Liu and James W Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)*, 20(1):46–61, 1973.

[19] Geoffrey J McLachlan, Sharon X Lee, and Suren I Rathnayake. Finite mixture models. *Annual review of statistics and its application*, 6:355–378, 2019.

[20] Lorenz Meier, Dominik Honegger, and Marc Pollefeys. Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6235–6240, 2015.

[21] Evariste Ntaryamira. *A generalized asynchronous method preserving the data quality of real-time embedded systems : Case of the PX4-RT autopilot.* Theses, Sorbonne Université, May 2021.

[22] Charles D Pack. The output of a d/m/1 queue. *SIAM Journal on Applied Mathematics*, 32(3):571–587, 1977.

[23] Evan M Palmer, Todd S Horowitz, Antonio Torralba, and Jeremy M Wolfe. What are the shapes of response time distributions in visual search? *Journal of experimental psychology: human perception and performance*, 37(1):58, 2011.

[24] Luigi Palopoli, Daniele Fontanelli, Nicola Manica, and Luca Abeni. An analytical bound for probabilistic deadlines. In *2012 24th Euromicro Conference on Real-Time Systems*, pages 179–188. IEEE, 2012.

[25] Paulo Pedro and Alan Burns. Schedulability analysis for mode changes in flexible real-time systems. In *Proceeding. 10th EUROMICRO Workshop on Real-Time Systems (Cat. No. 98EX168)*, pages 172–179. IEEE, 1998.

[26] Antonio Punzo. A new look at the inverse gaussian distribution with applications to insurance and economic data. *Journal of Applied Statistics*, 46(7):1260–1287, 2019.

[27] Adrian E Raftery. Bayesian model selection in social research. *Sociological methodology*, pages 111–163, 1995.

[28] Jorge Real and Alfons Crespo. Mode change protocols for real-time systems: A survey and a new proposal. *Real-time systems*, 26(2):161–197, 2004.

[29] Sidney Redner. *A guide to first-passage processes*. Cambridge university press, , 2001.

[30] Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, pages 461–464, 1978.

[31] Wolfgang Schwarz. The ex-wald distribution as a descriptive model of response times. *Behavior Research Methods, Instruments, & Computers*, 33(4):457–469, 2001.

[32] Venkata Seshadri. *The inverse Gaussian distribution: statistical theory and applications*, volume 137. Springer Science & Business Media, , 2012.

[33] Maurice CK Tweedie. Statistical properties of inverse gaussian distributions. i. *The Annals of Mathematical Statistics*, 28(2):362–377, 1957.

[34] Georg von der Brüggen, Nico Piatkowski, Kuan-Hsun Chen, Jian-Jia Chen, and Katharina Morik. Efficiently approximating the probability of deadline misses in real-time systems. In *30th Euromicro Conference on Real-Time Systems (ECRTS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[35] Georg von der Brüggen, Nico Piatkowski, Kuan-Hsun Chen, Jian-Jia Chen, Katharina Morik, and Björn B Brandenburg. Efficiently approximating the worst-case deadline failure probability under edf. In *2021 IEEE Real-Time Systems Symposium (RTSS)*, pages 214–226. IEEE, 2021.

[36] Zhi-Sheng Ye and Nan Chen. The inverse gaussian process as a degradation model. *Technometrics*, 56(3):302–311, 2014.

[37] Kevin Zagalo. Simso with probabilistic execution times, 2022.

[38] Kevin Zagalo, Yasmina Abdeddaïm, Avner Bar-Hen, and Liliana Cucu-Grosjean. Response time stochastic analysis for fixed-priority stable real-time systems. *IEEE Transactions on Computers*, pages 1–12, 2022.

[39] Kevin Zagalo and Olena Verbytska. rinversegaussian, 2022.