Chapter 9

Paper C
Estimating Deadline Miss Probabilities of Continuous-Emission Markov Chain Tasks.

Anna Friebe, Filip Marković, Alessandro V. Papadopoulos, and Thomas Nolte.
Under review.

Abstract

Estimating the response times of real-time tasks and applications is important for the analysis and implementation of real-time systems. Probabilistic approaches have gained attention over the past decade, as they provide a modeling framework that allows for less pessimism for the analysis of real-time systems. Among the different proposed approaches, Markov chains have been shown to be effective for the analysis of real-time systems, in particular, in the estimate of the pending workload probability distribution and of the deadline miss probability. However, this has been analyzed only for discrete emission distributions, but not for continuous ones. In this paper, we propose a method for analyzing the workload probability distribution and bounding the deadline miss probability for a task executing in a Constant Bandwidth Server, where execution times are described by a Markov model with Gaussian emission distributions. In the evaluation, deadline miss probability bounds and estimates are derived with a workload accumulation scheme. The results are compared to simulation and measured deadline miss ratios from tasks under the Linux Constant Bandwidth Server implementation SCHED_DEADLINE.

9.1 Introduction

Real-time systems are commonly characterized as hard or soft real-time systems. In a hard real-time system deadlines must always be met, but in a soft real-time system deadline misses can be tolerated to some extent. Deadline misses in a soft real-time system lead to a deterioration of the Quality of Service (QoS) [10] or Quality of Control (QoC) [28]. The number of deadline misses must be sufficiently low so that the QoS or QoC is retained at an acceptable level [8].

Hidden Markov Models (HMMs) have been utilized to model execution times in systems with dependencies, and where there is regularity in the variation of the execution times. In [5, 16], the authors have modeled execution times as Markov models with discrete emission distributions, including estimating the deadline miss probability under a Constant Bandwidth Server (CBS). Emission distributions have also been modeled as continuous Gaussian distributions [18, 17], with the advantage of potentially providing more robust estimates from a lower number of samples. Gaussian distributions also allow for representation with only two parameters, as opposed to the case where individual probabilities of each discrete execution time value are stored. The application of HMMs with continuous emission distributions has been limited to the estimation of the sole execution time [18].

This paper focuses on the problem of bounding and estimating the deadline miss probability of a real-time application, exploiting HMMs. In the literature, two concepts related to probabilistic deadlines are commonly used. The Deadline Miss Probability (DMP) is interpreted as the ratio of missed deadlines to the number of jobs in a long (tending to infinite) time interval. The Worst-Case Deadline Failure Probability (WCDFP) is interpreted as an upper bound on the probability of a deadline miss for any single job [12]. In this paper, we focus on the DMP as the long-run frequency interpretation, for the overall HMM and for each state separately.

More specifically, in this paper, we address the problem of upper bounding the workload distribution and deadline miss probability under CBS of a periodic task where execution times are modeled by a Markov chain with Gaussian emission distributions. We propose an iterative workload accumulation scheme, where workload distributions are accumulated sequentially over task periods. The scheme starts from a point of workload depletion, that is a task period with zero carry-in workload. The method provides an upper bound on the deadline miss probability in each state and overall under certain assumptions.

The method is evaluated by comparing the obtained results with the dead-

line miss ratio of tasks running under the Linux kernel implementation of CBS, `SCHED_DEADLINE` [22], and with results from simulation.

The paper is structured as follows. In Section 9.2, related work is discussed. The notation used in the paper and the system model is outlined in Section 9.3. The analysis for upper bounding the deadline miss probability is described in Section 9.4, and in Section 9.5 the parts are combined in an overall workload accumulation process. In Section 9.6 the evaluation is presented and results are provided. Conclusions and future work are discussed in Section 9.7.

9.2 Related Work

Davis and Cucu-Grosjean provide a comprehensive survey on probabilistic schedulability analysis techniques [12], along with a survey on probabilistic timing analysis [13].

Díaz et al. [14] presented a response time analysis for periodic tasks where execution times are independent random variables and showed that the backlog is a Markov chain.

Maxim and Cucu-Grosjean [29] showed that in systems where execution times, deadlines, and interarrival times are independent random variables, the Worst-Case Response Time (WCRT) can be found by synchronous release if deadlines are constrained and jobs are aborted when their deadline is missed.

Ivers and Ernst [19] addressed the case where execution times are dependent and proposed the use of Fréchet bounds and probability boxes.

Extreme Value Theory (EVT) has been applied in measurement-based statistical analysis of response times to find the probabilistic WCRT (pWCRT). This is an upper bound on the probability of exceeding a response time for every valid sequence of program executions and is based on finding the distribution of the extreme values, the distribution's tail. Most of the work in this regard has been done by Lu et al. [26, 25, 24]. Maxim et al. [30] have shown that the methods based on EVT provide sound results. EVT is applicable in cases of dependence, as long as there is stationarity [20] or extremal independence [33].

Real-time queuing theory [21] provides methods for analyzing the response time distribution specifically in the case of heavy traffic when utilization is close to 1.

Bozhko et al. [7] proposed a response time analysis with Monte Carlo simulation for fixed-priority preemptive scheduling with execution times as independent random variables.

Von der Brüggen et al. [35] provided a method for over-approximating the WCDFP under EDF for tasks with different execution modes. This includes derivation for acyclic task chain dependencies among a bounded number of subsequent jobs. The number of intervals considered is substantially reduced due to the observation that the probability of a deadline miss in an interval is bounded by the probability that the processor does not idle in the same interval.

Mills and Anderson [31] provide response time and tardiness bounds for soft real-time tasks with stochastic execution times, in a server-based scheduler. In this work, execution time dependence is considered within but not across time windows. A larger window leads to greater tardiness bounds. Liu, Mills, and Anderson [23] further proposed the use of independence thresholds, where independence is assumed for execution times exceeding a determined threshold value.

The CBS is described in Section 9.3. It was introduced by Abeni and Buttazzo [2], and used to obtain probabilistic deadlines for QoS guarantees [3]. Analysis under CBS has been performed with execution times [4, 32] and interarrival times [6, 27] modeled with probability distributions.

Tasks with dependent execution times have been modeled as Markov chains and been analyzed under CBS by Frías et al. [16, 5]. The steady-state response time distribution was calculated. The results were compared to running the task under Linux `SCHED_DEADLINE`. The time required for the analysis depends on the range of computation times, the number of states, and the resampling factor [34].

Execution times have been modeled as continuous Gaussian distributions in the context of emission distributions in a Markov chain [18, 17]. We are not aware of any work that analyzes this model in terms of response times or deadline miss probabilities. In this paper, we aim to bridge this gap and enable the use of an HMM with Gaussian emission distributions for schedulability analysis. Similarly as in the work of Frías et al. [16, 5], dependencies are explicit in the HMM, and the task is running in a CBS. The CBS provides isolation from other tasks on the system, so that the pending workload considered is carry-in workload from previous jobs of the same task, instead of workload from other tasks as in most work concerning response times. The choice of Gaussian distribution is partly based on simplicity and tractability. In [18] a HMM with Gaussian emission distributions was shown to be a valid model in a video decompression case. Modeling the execution times of each state as a Gaussian distribution may seem simplistic. However, several states with Gaussian distributions can be combined to form a more general distribution shape. In addition, if the means of the states' distributions differ significantly, the Markov Model transition probabilities may affect the response times to a

greater extent than the state distribution shapes. As an example, a high likelihood of several consecutive jobs in the state with the longest execution times will lead to much longer response times, compared to a case where there is high likelihood that a job in this state is followed by a job in the state with the shortest execution times. Nevertheless, the use of the Gaussian distribution is a limitation of this work, and therefore the method is also evaluated for non-Gaussian distributions. Here, an exponential distribution is chosen. Exponential-tail distributions have been used in pWCET analysis[9, 1, 11], as the tail beyond a certain point is a safe upper bound of light-tailed distributions.

The iterative approach that we propose in this paper provides a bound/ estimate already after a few accumulation periods, while the method proposed by Frías et al. requires the calculation of the full steady state response time distribution.

9.3 System Model and Notation

The notation used in the paper is outlined in Table 9.1. We use the notation $\hat{x}$ to indicate the estimate of a variable X, and we use the superscripts *, $^\uparrow$, and $^\downarrow$ for the true values, upper, and lower bounds, respectively.

We will use the concept or upper bounding random variables, as defined in Definition 9.3.1.

Definition 9.3.1 (cf. [15, 13])**.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be two random variables. We say that $\mathcal{X}$ is greater than or equal to $\mathcal{Y}$ (i.e., $\mathcal{X}$ upper bounds $\mathcal{Y}$), if the Cumulative Distribution Function (CDF) of $\mathcal{X}$ is never above that of $\mathcal{Y}$, and we denote this relation by $\mathcal{X} \geq \mathcal{Y}$.*

To upper bound workload distributions, we will use the partial Gaussian distribution, as defined in Definition 9.3.2. Let us consider a Gaussian $\mathcal{N}(\mu, \sigma^2)$ with probability density function $f(x|\mu, \sigma^2)$. $\Phi(x)$ is the cumulative density function of the standard normal distribution.

Definition 9.3.2. *We define a partial Gaussian distribution $\mathcal{N}^{tail}(\mu, \sigma^2, \alpha)$, originated from a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, as:*

$$f^{tail}(x|\mu, \sigma^2, \alpha) = \begin{cases} 0, & x \leq \alpha \\ \frac{1}{\Phi(\frac{\mu-\alpha}{\sigma})} f(x|\mu, \sigma^2) & x > \alpha \end{cases} \tag{9.1}$$

In a partial Gaussian distribution, the probability for the elements of the Gaussian distribution lower than α are set to zero and the remaining is normalized so that the distribution integrates into one.

Table 9.1: Overview of notation used in this paper.

Symbol	Description
Basic notation	
T	Task period
J_i	Job at task period i
a_i	Arrival time of J_i
d_i	Absolute deadline of J_i
D	Relative deadline
P	Server period
Q	Server budget
n	Number of server periods in a task period
k	Number of server periods in a relative deadline
S	Number of Markov states
$\boldsymbol{M}$	State transition matrix
N	Number of task periods in workload accumulation
Values of random variables	
c_i	Execution time of J_i
f_i	Finishing time of J_i
v_i	Workload at task period i
h	Accumulation sequence of state visits in Markov chain since workload depletion
$\tilde{h}$	Accumulation vector of the number of visits in each Markov state since workload depletion
Probability distributions and probabilities	
C	Execution time distribution
$\mathcal{V}_h, \mathcal{V}_{\tilde{h}}$	Workload distribution associated with an accumulation sequence or vector
$m_{i,j}$	Transition probability from state i to state j
$\xi(s)$	Stationary probability of being in s
$p_{in}(s, \tilde{h})$	Probability of entering s with $\tilde{h}$
$p_{co}(s, \tilde{h})$	Probability that $\tilde{h}$ in s carries workload to the next task period
$p_{wd}(s)$	Probability of workload depletion in s
p_{dm}	Deadline miss probability
$\beta(s)_N$	Probability of being in state s with h longer than N.

In the derivation of workload distributions, we use convolutions as defined in Definition 9.3.3.

Definition 9.3.3. *The convolution of f and g, denoted with the $*$ operator is:*

$$[f * g](z) = \int_{-\infty}^{\infty} f(z - x)g(x)\,\mathrm{d}x$$

9.3.1 Task Model

A real-time task τ consists of a sequence of jobs J_i, $i \in \mathbb{N}$. The arrival time of J_i is a_i. The tasks are periodic, with no jitter, i.e., $a_{i+1} = a_i + T$, with a_0 being

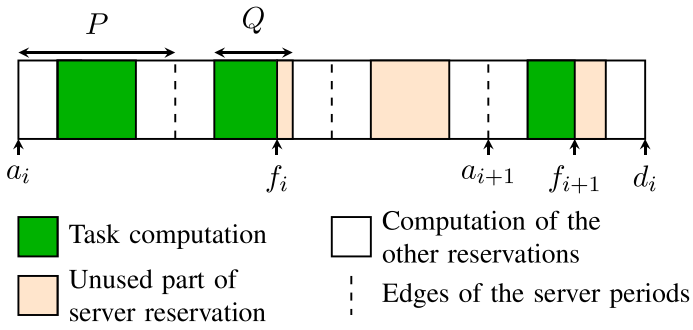


Figure 9.1: An illustration of the task model and the CBS.

the arrival time of the first job. The execution time of J_i is c_i and its finishing time is f_i. The jobs may be preempted, and $f_i \geq a_i + c_i$. The execution time c_i is modeled as a random variable. The random variable $\mathcal{R}$ models the time from activation time to finish time of a job.

The deadline of a job J_i is $d_i = a_i + D$, where D is the relative deadline. Jobs are executed until completion, even when a deadline is missed. The relative deadline can be longer than the task period. We consider the probability of a deadline miss p_{dm}, that is the overall probability that a job finishes after the deadline, $p_{dm} = p(\mathcal{R} > D)$.

9.3.2 Scheduling Algorithm

The considered scheduling algorithm is reservation-based, namely the Constant Bandwidth Server (CBS). Each task has its own server. Within each server period P, the task is guaranteed to receive Q units of processing time. The fraction of the processing resource dedicated to this task, the bandwidth, is $B = Q/P$. We choose the server period so that it divides the task period evenly, i.e., $T = nP$, where n is a positive integer. We also define k, a positive integer that is the number of server periods in the relative deadline D, $D = kP$.

An illustration of the task model and CBS is shown in Figure 9.1. In this illustration, the task period is divided into three server periods, and the bandwidth is 0.5. As illustrated, the deadline of a job does not need to be within a task period from the arrival; the relative deadline may be longer than the period.

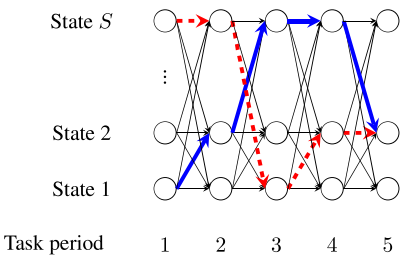


Figure 9.2: Illustration of the period by period workload accumulation sequence.

9.4 Execution Time Model and Analysis

9.4.1 Markov Chain Execution Times

In this section, we consider a task, where the execution time distribution is described by a Markov model characterized by the triplet $\langle \mathbb{S}, \boldsymbol{M}, \mathbb{C} \rangle$. $\mathbb{S} = \{1, 2, \ldots, S\}$ is the set of S states, $S \in \mathbb{N}$. $\boldsymbol{M}$ is the $S \times S$ state transition matrix, where the element $m_{a,b}$ represents the conditional probability of being in state b at task period $i + 1$, given that at task period i the state is a. $\mathbb{C} = \{C_1, C_2, \ldots, C_S\}$ is the set of execution time distributions, or *emission distributions* related to the respective state. These are modeled as Gaussian distributions with mean μ_s, and variance σ_s^2, i.e., $C_s \sim \mathcal{N}(\mu_s, \sigma_s^2)$.

9.4.2 Overview of the Proposed Approach

To upper bound the deadline miss probability of the task running under CBS, we propose a method based on a workload accumulation scheme. The main idea is outlined below, followed by the details in the remaining subsections.

In each task period, task τ is guaranteed nQ units of processing time. The pending workload at the i-th task period is denoted as v_i and defined as in [3]:

$$v_i = \underbrace{\max(0, v_{i-1} - nQ)}_{\text{carry-in workload}} + c_i \tag{9.2}$$

where the first term accounts for the previous workload, initially set to 0, and the first period is $v_1 = c_1$. An example of how the workload evolves according to the Markov chain model is shown in Figure 9.2. We start with zero initial pending workload and add one task period at a time of workload accumulation. In the figure, the dashed red and solid blue lines depict two possible workload accumulation sequences that are in state 2 at task period 5 from workload depletion. The *accumulation sequence* is modeled as a random variable $\mathcal{H}$ that

can take the values of any possible workload accumulation path. In the example from Figure 9.2, in the dashed red path it takes the value $h = (S, S, 1, 2, 2)$.

Davis and Cucu-Grosjean [12] define the deadline miss probability for a task as the average deadline miss probability of task jobs during a hyperperiod. In this work, using the CBS allows us to discard pending workload from other tasks in the task set. The notion of hyperperiod is therefore irrelevant, and we define the Deadline Miss Probability *DMP* as the average deadline miss probability of the task's jobs. We do this by considering accumulation sequences. More specifically, we define the probability of a job arrival leading to the accumulation sequence h as $p_{in}(h)$. Since each job arrival leads to one specific accumulation sequence, the sum of $p_{in}(h)$ over all h equals 1. We define the conditional deadline miss probability for a job with accumulation sequence h as $p_{dm}(h)$. Then, the *DMP* is defined as the sum of the deadline miss probabilities for each accumulation sequence weighted with their respective probabilities:

$$DMP = \sum_{\forall h} p_{in}(h) p_{dm}(h) \tag{9.3}$$

Problem: *The sum of Equation* (9.3) *has a countably infinite number of terms. This paper investigates how to find a bound for DMP with a finite number of terms.*

In the remainder of this section, we will provide an upper bound on DMP by finding the upper bounds on p_{in} and p_{dm}. The process is divided into two steps. First, we compute the upper bounds on p_{in} and p_{dm} of accumulation sequences up to length N, thus approaching the true deadline miss probability. To make a safe bound, we then sum the p_{in} values in the remaining accumulation sequences of length $N + 1$ to infinity, assuming that p_{dm} for these periods is 1. This sum is referred to as β. This finally leads to the safe over-approximation of *DMP*.

The steps for deriving a bound on *DMP* are presented in this paper as follows:

Section 9.4.3: To determine upper bounds on p_{dm} and p_{in} in Equation (9.3) we need to find upper bounds on the pending workload distributions associated with each state and accumulation sequence. This is done in Equations (9.20) and (9.24).

Section 9.4.4: Bounds on p_{in} depend on the probability of carry-over workload p_{co} from the previous step and the transition probabilities m. In the first step of the accumulation process, p_{in} depends on the probability of workload depletion p_{wd} for each state. With p_{wd} propagating along the accumulation, each p_{in} is a linear combination of p_{wd} for the different states.

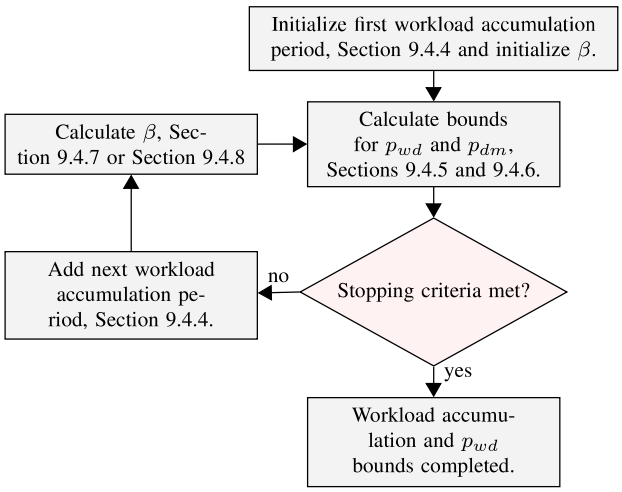


Figure 9.3: The workload accumulation process.

Section 9.4.5: Bounds on p_{wd} are derived, these rely on the sum of p_{in} in accumulation periods after N, denoted as β.

Section 9.4.6: In this section bounds on p_{dm} are presented, using the bounds on $\mathcal{V}$, p_{in} and β. The upper bound of p_{dm} for a state is defined in Equation (9.30).

Section 9.4.7: In this section, we derive a bound on β. β is the minimum of Equations (9.31) and (9.32), and is utilized for computing the lower bounds on p_{in}, p_{co} and finally $\mathcal{V}$.

Section 9.4.8: In this section, we derive an estimate of β as an alternative to the bound.

The parts are tied together in the iterative workload accumulation algorithm presented along with an example in Section 9.5. In Figure 9.3 the process is illustrated with reference to the different sections. Section 9.4.3 is not referenced in the figure as it is a basis for all the remaining sections.

9.4.3 Bounding the Conditional Pending Workload Distribution Associated with a Workload Accumulation Sequence

We seek upper and lower bounds of the conditional pending workload distribution conditioned on having a given accumulation sequence since the most recent point of workload depletion. As an example from Figure 9.2, we want to define the pending workload distribution in state 2 at task period 5, provided that the transitions since workload depletion have been along the path marked as dashed red, $h = (S, S, 1, 2, 2)$.

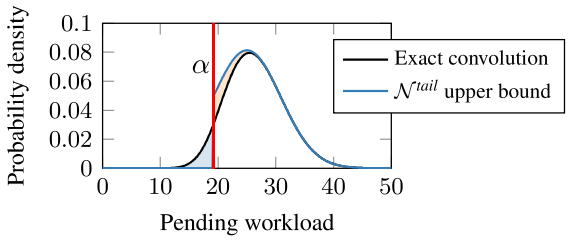


Figure 9.4: Illustration of a convolution result with an upper bounding partial Gaussian distribution.

We denote the conditional pending workload distribution, conditioned on a given accumulation sequence h as $\mathcal{V}_h$ with a probability density function $p(v|\mathcal{H} = h)$.

The lower and upper bounds for this conditional pending workload distribution depend only on the number of visits in each state in the accumulation sequence and are independent of their order. We model the accumulation vector as a random variable $\tilde{\mathcal{H}}$ that takes values as S-dimensional vectors of non-negative integer values, where each value represents the number of visits in a state. This means that the dashed red and the solid blue accumulation sequence lines in Figure 9.2 will contribute to the same accumulation vector at task period 5 since they both have the same number of visits in each state, that is $\tilde{h} = [1, 2, \ldots, 2]$. We define the operation $\tilde{h}[s]$ as taking the s-th element of $\tilde{h}$. We also define $\tilde{h}_{+s}$ as $\tilde{h}$ with the s-th element incremented by one, to simplify the notation of the accumulation vector in s with carry-in workload from $\tilde{h}$.

The number of possible bounded pending workload distributions of length N in a system with S states is $\binom{N+S-1}{N} = \frac{(N+S-1)!}{N!(S-1)!}$, as opposed to S^N which would be needed if ordering were taken into account. For a fixed number of states S, the number of distributions to consider increases with the number of periods considered as $\mathcal{O}(N^{S-1})$.

Recalling Definition 9.3.1, we derive an upper bound conditional pending workload distribution $\mathcal{V}^{\uparrow}_{\tilde{h}} \geq \mathcal{V}_h$.

In the following, we show that a *partial Gaussian distribution* (see Definition 9.3.2) upper bounds the conditional pending workload distribution. An illustration is in Figure 9.4, where the blue curve and red line upper bounds the black workload distribution, lower probability values (blue area) are moved to higher (orange area).

Theorem 9.1. *The conditional pending workload distribution associated with each state s and accumulation vector $\tilde{h}$ is upper bounded by*

$\mathcal{N}^{tail}(\mu(\tilde{h}), \sigma^2(\tilde{h}), \alpha(\tilde{h}, s))$.

We prove this by induction. For clarity we state a Lemma 9.2 for the base case, and Lemma 9.3 for the inductive step.

Lemma 9.2. *The partial Gaussian distribution $\mathcal{N}^{tail}(\mu_s, \sigma_s^2, 0)$ upper bounds the conditional pending workload distribution $\mathcal{V}_{\tilde{h}}$ in state s immediately after a point of workload depletion.*

Proof. In the first step after workload depletion, the conditional pending workload distribution $\mathcal{V}_h$ equals the execution time distribution of the entered state s. Excluding negative values and normalizing gives an upper bounding distribution, as probabilities are moved from lower workload values to higher. Thus, $\mathcal{N}^{tail}(\mu_s, \sigma_s^2, 0)$ is an upper bound. □

With non-zero carry-over workload in a transition from state s_p with accumulation vector $\tilde{h}$, and an upper bound on the workload distribution $\mathcal{N}^{tail}(\mu(\tilde{h}), \sigma^2(\tilde{h}), \alpha(\tilde{h}, s_p))$, into state s, we will show that the conditional pending workload distribution is upper bounded by the partial Gaussian distribution $\mathcal{N}^{tail}(\mu(\tilde{h}_{+s}), \sigma^2(\tilde{h}_{+s}), \alpha(\tilde{h}_{+s}, s))$. Below, in Equations (9.4) and (9.5) we define $\mu(\tilde{h}_{+s})$ and $\sigma^2(\tilde{h}_{+s})$. Equations (9.6) and (9.7) are used to simplify the expression of the starting value $\alpha(\tilde{h}_{+s}, s)$ of the resulting upper bounding distribution, defined in Equation (9.8). Here $sf^{-1}(q, \mu, \sigma^2)$ denotes the inverse survival function at q of a Gaussian distribution with mean μ, and variance σ^2. Equation (9.7) defines $K(\tilde{h}, s_p)$, the normalization factor needed for the conditional probability calculation. We perform a convolution with the upper bounding workload distribution in s_p with $\tilde{h}$ extending past the task period. $K(\tilde{h}, s_p)^{-1}$ is the integral of this part, to get a probability distribution integrating to one.

$$\mu(\tilde{h}_{+s}) = \mu_s + \sum_{i=1}^{S} \tilde{h}[i](\mu_i - nQ) \tag{9.4}$$

$$\sigma^2(\tilde{h}_{+s}) = \sigma_s^2 + \sum_{i=1}^{S} \tilde{h}[i]\sigma_i^2 \tag{9.5}$$

$$\alpha_\Delta(\tilde{h}, s_p) = \max(0, \alpha(\tilde{h}, s_p) - nQ) \tag{9.6}$$

$$K(\tilde{h}, s_p) = \left[\Phi\left(\frac{\mu(\tilde{h}) - nQ - \alpha_\Delta(\tilde{h}, s_p)}{\sigma(\tilde{h})}\right)\right]^{-1} \tag{9.7}$$

$$\alpha(\tilde{h}_{+s}, s) = \begin{cases} 0 & \tilde{h} = \mathbf{0} \\ sf^{-1}(\frac{1}{K(\tilde{h}, s_p)}, \mu(\tilde{h}_{+s}), \sigma^2(\tilde{h}_{+s})) & \tilde{h} \neq \mathbf{0} \end{cases} \tag{9.8}$$

Lemma 9.3. *When transitioning with non-zero carry-over workload from state s_p with accumulation vector $\tilde{h}$ into state s, and with an upper bound on the workload distribution in the previous task period $\mathcal{V}^{\uparrow}$ as $\mathcal{N}^{tail}(\mu(\tilde{h}), \sigma^2(\tilde{h}), \alpha(\tilde{h}, s_p))$, the conditional pending workload distribution is upper bounded by $\mathcal{N}^{tail}(\mu(\tilde{h}_{+s}), \sigma^2(\tilde{h}_{+s}), \alpha(\tilde{h}_{+s}, s))$.*

Proof. The strictly positive carry-over workload distribution is the normalized workload tail beyond the task period processing time, which can be written as $\mathcal{N}^{tail}(\mu(\tilde{h}) - nQ, \sigma^2(\tilde{h}), \max(0, \alpha(\tilde{h}, s_p) - nQ))$.

The execution time distribution in state s is described by $\mathcal{N}(\mu_s, \sigma_s^2)$. The resulting upper bound on the conditional workload distribution $\mathcal{V}^{\uparrow}_{\tilde{h}_{+s}}$ in state s with accumulation vector $\tilde{h}_{+s}$ is the result of the convolution, Definition 9.3.3, of the probability density functions of the execution time and the upper bound on the positive carry-over workload. This holds because execution times are independent random variables and the dependence of the Markov model is restricted to the transition probabilities.

To simplify the notation in the convolution expansion, we introduce the following:

$$\mu_R(z) = \frac{(z - \mu_s)\sigma^2(\tilde{h}) + (\mu(\tilde{h}) - nQ)\sigma_s^2}{\sigma_s^2 + \sigma^2(\tilde{h})} \tag{9.9}$$

$$\sigma_R^2 = \frac{\sigma_s^2 \sigma^2(\tilde{h})}{\sigma_s^2 + \sigma^2(\tilde{h})} \tag{9.10}$$

$$\mu_{\Sigma\Delta} = \mu_s + \mu(\tilde{h}) - nQ \tag{9.11}$$

$$\sigma_\Sigma^2 = \sigma_s^2 + \sigma^2(\tilde{h}) \tag{9.12}$$

Expanding the convolution for $\mathcal{V}^{\uparrow}_{\tilde{h}_{+s}}$:

$$\begin{aligned}
&\int_{-\infty}^{\infty} f(z - x|\mu_s, \sigma_s^2) f^{tail}(x|\mu(\tilde{h}) - nQ, \sigma^2(\tilde{h}), \alpha_\Delta)\, \mathrm{d}x \\
&= K(\tilde{h}, s_p) \int_{\alpha_\Delta}^{\infty} f(z - x|\mu_s, \sigma_s^2) f(x|\mu(\tilde{h}) - nQ, \sigma^2(\tilde{h}))\, \mathrm{d}x \\
&= K(\tilde{h}, s_p) f(z|\mu_{\Sigma\Delta}, \sigma_\Sigma^2) \int_{\alpha_\Delta}^{\infty} f(x|\mu_R(z), \sigma_R^2)\, \mathrm{d}x
\end{aligned} \tag{9.13}$$

where the last step isolated the part of the expression that is independent of x. We recognize the integral in the second factor of Equation (9.13) as the survival function or 1-CDF at α_Δ of $\mathcal{N}(\mu_R(z), \sigma_R^2)$. This is monotonically increasing and goes to 0 as z goes to $-\infty$ and to 1 as z goes to ∞. Thus,

we can find a value $\alpha(\tilde{h}_{+s}, s)$ where the area under the curve of the exact convolution of the pending workload distribution up to $\alpha(\tilde{h}_{+s}, s)$ equals the area between the curves of the exact pending workload distribution and the partial Gaussian distribution, $\mathcal{N}^{tail}(\mu_{\Sigma\Delta}, \sigma^2_\Sigma, \alpha(\tilde{h}_{+s}, s))$ from $\alpha(\tilde{h}_{+s}, s)$. An illustration is provided in Figure 9.4. Using K for normalization of the partial Gaussian distribution ensures that the tail of the upper bound approaches the tail of the full convolution asymptotically. We find the $\alpha(\tilde{h}_{+s}, s)$ which gives:

$$K(\tilde{h}, s_p) \int_{\alpha(\tilde{h}_{+s}, s)}^{\infty} f(x|\mu_{\Sigma\Delta}, \sigma^2_\Sigma)\, \mathrm{d}x = 1 \tag{9.14}$$

As we know that the result of the convolution integrates to one, this shows that the two regions described and illustrated in Figure 9.4 have the same area. Replacing the exact convolution with the partial Gaussian is equivalent to moving probability weight from lower pending workload values to higher, leading to an overestimate. We have:

$$\mu(\tilde{h}_{+s}) = \mu_{\Sigma\Delta} \tag{9.15}$$

$$\sigma^2(\tilde{h}_{+s}) = \sigma^2_\Sigma \tag{9.16}$$

$$\alpha(\tilde{h}_{+s}, s) = sf^{-1}\left(\frac{1}{K(\tilde{h}, s)}, \mu_{\Sigma\Delta}, \sigma^2_\Sigma\right) \tag{9.17}$$

This concludes our proof. □

Considering all states s_p containing the accumulation vector $\tilde{h}$, we define:

$$\alpha_\Delta(\tilde{h}) = \max(0, \max_{\forall s_p} \alpha(\tilde{h}, s_p) - nQ) \tag{9.18}$$

We use this instead of Equation (9.6) in Equations (9.7) and (9.8). With these lemmas we are ready to prove Theorem 9.1.

Proof. We prove this by induction.
Base case: For the task period after workload depletion, this follows by Lemma 9.2.
Inductive hypothesis: If we have such a workload distribution upper bound for all states and accumulation vectors in one task period, it also holds for the next period.
Inductive step: This follows from Lemma 9.3 and taking the maximum α in Equation (9.18). □

With similar reasoning, we can use a Gaussian distribution as a lower bound of the pending workload distribution $\mathcal{V}^{\downarrow}_{\tilde{h}} \leq \mathcal{V}_h$. This is illustrated in

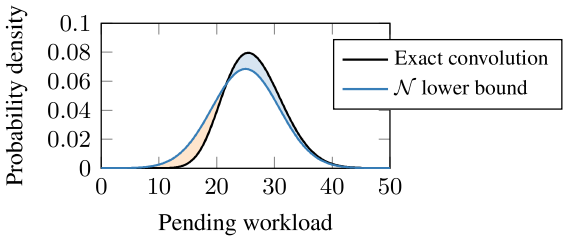


Figure 9.5: An illustration of a convolution result and the Gaussian distribution that forms a lower bound.

Figure 9.5. As $K > 1$, and the area under the curve equals one for both the Gaussian distribution with mean $\mu_{\Sigma\Delta}$ and variance σ^2_{Σ} and the result of the convolution, replacing the workload distribution with the Gaussian implies moving probability weight from higher workload values to lower, thus providing a lower bound.

9.4.4 Bounds on the Probability of Entering a State with an Accumulation Vector

Each state s in each task period is associated with one or more accumulation vectors, $\tilde{h}$. Each accumulation vector in a state is associated with lower and upper bounds on the probability of entering this state with the associated accumulation vector $p^{\downarrow}_{in}(s, \tilde{h})$ and $p^{\uparrow}_{in}(s, \tilde{h})$. Each accumulation vector in a state is also associated with lower and upper bounds on the probability of the workload contributing to carry-over into the next period, $p^{\downarrow}_{co}(s, \tilde{h})$ and $p^{\uparrow}_{co}(s, \tilde{h})$.

In the first period, with no carry-in workload, each state is associated with a single accumulation vector containing zeros except for the current state that is set to 1. The probability of entering a state in the first period after workload depletion depends on the stationary probabilities $\xi(s)$ of being in each state, the probability of workload depletion $p_{wd}(s)$ in each state, and the transition matrix. The stationary probabilities and the transition matrix are known, but the probability of workload depletion in each state is unknown at this stage. In Section 9.4.5 we will describe how to retrieve this. Assuming that we have lower and upper bounds on the probabilities of workload depletion, $p^{\downarrow}_{wd}(s)$ and $p^{\uparrow}_{wd}(s)$, we can calculate lower and upper bounds on the probability of entering

the states in the first workload accumulation period as

$$p_{in}^{\downarrow}(s,\tilde{h}) = \sum_{s_p=1}^{S} \xi(s_p) p_{wd}^{\downarrow}(s_p) m_{s_p,s} \tag{9.19}$$

$$p_{in}^{\uparrow}(s,\tilde{h}) = \sum_{s_p=1}^{S} \xi(s_p) p_{wd}^{\uparrow}(s_p) m_{s_p,s} \tag{9.20}$$

Since there is only one accumulation vector in each state in the first accumulation period, there is no dependency on $\tilde{h}$.

For the following periods, accumulation vectors are created by copying each accumulation vector from the states in the previous task period and incrementing the current state element by 1. We denote this vector as $\tilde{h}_{+s}$ Note that paths from different states in the previous period can lead to the same accumulation vector. The probability of entering s with $\tilde{h}_{+s}$ depends on the probability of $\tilde{h}$ contributing to carry-over into the next period in all states, and transition probabilities.

The probability that the workload contributes to carry-over into the next period is the probability of entering the state with this accumulation vector times the probability that the conditional pending workload exceeds the available processor time in a task period. This probability is bounded by $p_{co}^{\downarrow}(s,\tilde{h})$ and $p_{co}^{\uparrow}(s,\tilde{h})$, then calculated as:

$$\begin{aligned} p_{co}^{\downarrow}(s,\tilde{h}) &= p_{in}^{\downarrow}(s,\tilde{h}) p(\mathcal{V}_{\tilde{h}}^{\downarrow} > nQ) \\ &= p_{in}^{\downarrow}(s,\tilde{h}) p(\mathcal{N}(\mu(\tilde{h}),\sigma^2(\tilde{h})) > nQ) \end{aligned} \tag{9.21}$$

$$\begin{aligned} p_{co}^{\uparrow}(s,\tilde{h}) &= p_{in}^{\uparrow}(s,\tilde{h}) p(\mathcal{V}_{\tilde{h}}^{\uparrow} > nQ) \\ &= p_{in}^{\uparrow}(s,\tilde{h}) p(\mathcal{N}^{tail}(\mu(\tilde{h}),\sigma^2(\tilde{h}),\alpha(\tilde{h})) > nQ) \end{aligned} \tag{9.22}$$

The probability of entering state s with the accumulation vector $\tilde{h}$ is lower and upper bounded by:

$$p_{in}^{\downarrow}(s,\tilde{h}_{+s}) = \sum_{s_p=1}^{S} p_{co}^{\downarrow}(s_p,\tilde{h}) m_{s_p,s} \tag{9.23}$$

$$p_{in}^{\uparrow}(s,\tilde{h}_{+s}) = \sum_{s_p=1}^{S} p_{co}^{\uparrow}(s_p,\tilde{h}) m_{s_p,s}. \tag{9.24}$$

9.4.5 Bounds on the Probability of Workload Depletion

Bounds on the probability of workload depletion for each state are used to calculate p_{in} in the first step after workload depletion in Equations (9.19)

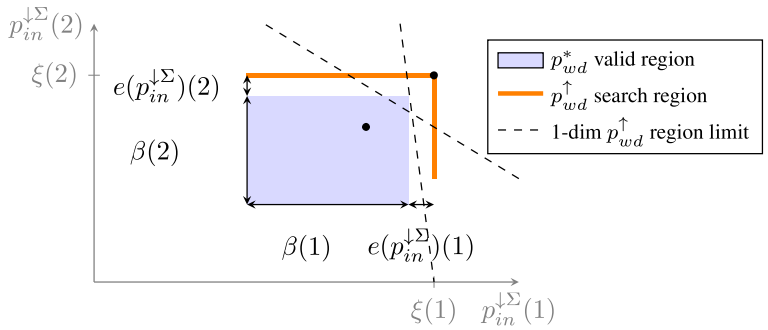


Figure 9.6: An illustration of the possible valid region of $p_{in}^{\downarrow\Sigma}$ for two states, if the true probabilities of workload depletion would be used as $p_{wd}^{\downarrow}$ in Equation (9.19).

and (9.20), and are further propagated to all p_{in}. The true workload depletion probability p_{wd}^{*} is unknown, and in this section we will derive bounds for it. Had p_{wd}^{*} been known, and input as $p_{wd}^{\downarrow}$ in Equation (9.19), the sum of the lower bounds on the probabilities $p_{in}^{\downarrow\Sigma}$ associated with all accumulation vectors accounted for would be lower than the stationary probabilities for all states. Using $\tilde{h} \in (s,i)$ to denote the set of accumulation vectors associated with state s in task period i, we formulate:

$$p_{in}^{\downarrow\Sigma}(s,p_{wd}) = \sum_{i=1}^{N} \sum_{\tilde{h}\in(s,i)} p_{in}^{\downarrow}(s,\tilde{h}) \leq \xi(s), \quad \forall s \tag{9.25}$$

We define $\beta(s)_N$ as the probability of being in s with workload accumulation past N. We also define $e(p_{in}^{\downarrow\Sigma})$ as the error introduced by using the lower bounding Gaussian distribution in place of the true convolution result. Had we known the true p_{wd}^{*} and input it as $p_{wd}^{\downarrow}$ in Equation (9.19) that would give values of $p_{in}^{\downarrow\Sigma}$ in the blue area of Figure 9.6.

Had the true workload depletion probability p_{wd}^{*} been known and input as $p_{wd}^{\uparrow}$ in Equation (9.20), the sum of the upper bounds of the probabilities $p_{in}^{\uparrow\Sigma}$ associated with all accumulation vectors would be greater than the stationary probabilities minus the probability of being in the state with longer accumulation vectors $\beta(s)_N$, for all states. This is outlined in Equation (9.26):

$$p_{in}^{\uparrow\Sigma}(s,p_{wd}) = \sum_{i=1}^{N} \sum_{\tilde{h}\in(s,i)} p_{in}^{\uparrow}(s,\tilde{h}) \geq \xi(s) - \beta(s)_N, \forall s \tag{9.26}$$

We define $e(p_{in}^{\uparrow\Sigma})$ the error introduced by using the upper bounding partial Gaussian distribution in place of the true convolution result. If we input true

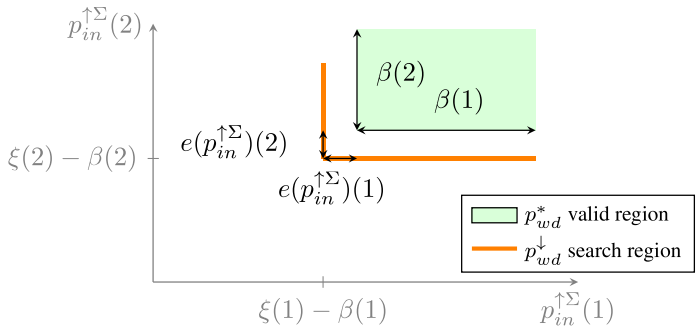


Figure 9.7: An illustration of the possible valid region of $p_{in}^{\uparrow\Sigma}$ for two states, if the true probabilities of workload depletion would be used as $p_{wd}^{\uparrow}$ in Equation (9.20).

workload depletion probabilities as $p_{wd}^{\uparrow}$ in Equation (9.20) the resulting $p_{in}^{\uparrow\Sigma}$ would be in the range depicted as green in Figure 9.7. This allows us to bound the true workload depletion probabilities to values mapping within both the blue region of Figure 9.6 and the green region of Figure 9.7.

An upper bound of the workload depletion probability p_{wd} is found for each state as the maximum of the values that lead to $p_{in}^{\downarrow\Sigma}$ along the orange lines of Figure 9.6.

Theorem 9.4. *The state-wise maximum of p_{wd} taken within the region of p_{wd} leading to $p_{in}^{\downarrow\Sigma}(s) \leq \xi(s)$ for all states, and where equality holds for all but at most one s is an upper bound of p_{wd}.*

Proof. Each $p_{in}^{\downarrow}(s, \tilde{h})$ is a linear combination of p_{wd} for all states, this follows from Equations (9.19), (9.21) and (9.23). Combined with Equation (9.25) it follows that $p_{in}^{\downarrow\Sigma}(s)$ is also a linear combination of p_{wd} for all states, which for some positive factors $A_{i,s}$ we can write:

$$p_{in}^{\downarrow\Sigma}(s, p_{wd}) = \sum_{i=1}^{S} A_{i,s} p_{wd}(i) \tag{9.27}$$

Assume that we have the true workload depletion probability p_{wd}^{*}. For an arbitrary state dimension j in p_{wd}, we can increase $p_{wd}(j)$ with an amount $\delta_{s,j}$ so that we reach a plane defined by Equation (9.28). For the lowest $\delta_{s,j}$, the first plane we encounter along the line, we have $p_{in}^{\downarrow\Sigma}(i) \leq \xi(i), \forall i \neq s$.

$$p_{in}^{\downarrow\Sigma}(s, p_{wd}) = A_{j,s}(p_{wd}^{*}(j) + \delta_{s,j}) + \sum_{i=1, i\neq j}^{S} A_{i,s} p_{wd}^{*}(i) = \xi(s) \tag{9.28}$$

Because the true p^*_{wd} gives $p^{\downarrow\Sigma}_{in}(s) \leq \xi(s)$ and due to the linear combination, it follows that at least one dimension of p_{wd} is an upper bound at every point in the planes defined by $p^{\downarrow\Sigma}_{in}(s, p_{wd}) = \xi(s)$, including the point with equality for all s - the upper right corner on the orange lines in Figure 9.6. If a particular dimension does not have an upper bound at this point, we have an upper bound on one of the planes, as the black dot in the illustration in Figure 9.6. The plane separating the region of the plane with upper bounds on this dimension from the region with underestimates will cross at least one of the orange lines, which ensures that an upper bound will be found in the region. Illustrations of possible separating planes are dashed lines in Figure 9.6. This concludes the proof. □

Similarly, a lower bound on the workload depletion probability p_{wd} is found for each state as the minimum of the values that lead to $p^{\uparrow\Sigma}_{in}$ along the orange lines of Figure 9.7. By using the lower bound from Figure 9.7 to determine the endpoints of the orange sections in Figure 9.6 and the upper bound from Figure 9.6 to determine the endpoints of the orange sections in Figure 9.7 $e(p^{\downarrow\Sigma}_{in})$ and $e(p^{\uparrow\Sigma}_{in})$ can be ignored. The endpoints are adjusted if they are outside the valid range for p_{wd}, that is if the probabilities are lower than 0 or higher than 1. As all $p^{\downarrow\Sigma}_{in}(s)$ and $p^{\uparrow\Sigma}_{in}(s)$ depend linearly on all $p_{wd}(s)$, we only need to consider the endpoints of the orange sections.

9.4.6 Upper Bounding the Deadline Miss Probability

We can then calculate an upper bound on the deadline miss probability as defined in Equation (9.3). The upper bound on the deadline miss probability $p^{\uparrow}_{dm}$ conditioned on an accumulation vector $\tilde{h}$ and a state s is:

$$\begin{aligned} p^{\uparrow}_{dm}(s, \tilde{h}) &= p(\mathcal{V}^{\uparrow}_{\tilde{h}} > kQ) \\ &= p(\mathcal{N}(\mu(\tilde{h}), \sigma^2(\tilde{h})) > kQ). \end{aligned} \tag{9.29}$$

The upper bound on the deadline miss probability in a state is

$$p^{\uparrow}_{dm}(s) = \frac{\beta(s)^{\uparrow}_{N}}{\xi(s)} + \frac{\sum_{i=1}^{N} \sum_{\tilde{h} \in (s,i)} p^{\uparrow}_{in}(s, \tilde{h}) p^{\uparrow}_{dm}(s, \tilde{h})}{\xi(s)}. \tag{9.30}$$

9.4.7 Bounding the Probability of Longer Workload Accumulation

The sum of p_{in} in task periods beyond N, β is still unknown, and in this section a bound is derived. β decreases monotonically with each accumulated period,

as all probabilities are non-negative. For each period, $\beta(s)$ decreases with at least the lower bound on the probability of being in the state in the same period, i.e.

$$\beta(s)_N \leq \beta(s)_{N-1} - \sum_{\tilde{h} \in (s,N)} p_{in}^{\downarrow}(s, \tilde{h}) \forall s \tag{9.31}$$

We also know that β is at most the stationary probability minus the lower bound on the probabilities accounted for, i.e.

$$\beta(s)_N \leq \xi(s) - \sum_{i=1}^{N} \sum_{\tilde{h} \in (s,i)} p_{in}^{\downarrow}(s, \tilde{h}) \tag{9.32}$$

Thus, given a safe bound for the probability of accumulation vectors not accounted for, β, in one accumulation period, we can obtain safe bounds for subsequent periods as the minimum of Equations (9.31) and (9.32).

9.4.8 Estimating the Probability of Longer Workload Accumulation

As an alternative or complement to the bound of β presented in Section 9.4.7, β can be estimated. First, the probability of workload depletion is estimated as the mean of the upper and lower bounds.

$$\widehat{p_{wd}} = \frac{p_{wd}^{\downarrow} + p_{wd}^{\uparrow}}{2} \tag{9.33}$$

Then we estimate β according to

$$\hat{\beta}(s)_N = \xi(s) - \sum_{i=1}^{N-1} \sum_{\tilde{h} \in (s,i)} p_{in}^{\downarrow}(s, \tilde{h}), \tag{9.34}$$

where $\widehat{p_{wd}}$ is used instead of $p_{wd}^{\downarrow}$ in Equation (9.19) for the first accumulation period. A new estimate is retrieved for each accumulation period. β of the first period is estimated as:

1. Let the probabilities of the first task period be ξ.
2. Calculate the probability of carry over into s in the second period from all C_i and M.
3. Set $\hat{\beta}(s)$ to the probability of being in the second period relative to the sum of both periods, scaled with $\xi(s)$.

9.5 Iterative Workload Accumulation

We propose an iterative approach where workload periods are successively accumulated. The process is illustrated in Figure 9.3 and ends when one of the following stopping criteria is met:

1. The upper bounds of the workload depletion probability of all states have turned from decreasing to increasing, or the lower bounds have turned from increasing to decreasing.

2. A maximum number of task periods is reached.

The first condition is met if the workload depletion probability bounds converge, or if the region within the bounds starts to grow. With each accumulation period, a convolution is performed, potentially increasing the error introduced by using the upper and lower bounding distributions in place of the true convolution result. This is illustrated by the white space between the blue area and the orange lines in Figure 9.6, and by the white space between the green area and the orange lines in Figure 9.7. If the increase in this error is not compensated by a sufficiently low probability of the associated accumulation vectors, the bounding region of the workload depletion probability can start to increase, and we stop at the period with the tightest bound.

The second condition is needed in the case where the bounds on the workload depletion probabilities or deadline miss probabilities diverge from the beginning. This may be due to insufficient bandwidth provided to the task in the CBS, or because the errors introduced are too large. The second condition is also activated when we have a slow convergence of the workload depletion probability bounds.

As an example we take a Markov model defined by:

$$S = 2, \qquad M = \begin{pmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{pmatrix}, \qquad \mathcal{C} = \{\mathcal{N}(20, 9), \mathcal{N}(40, 16)\}.$$

The stationary probabilities are 0.875 for state 1 and 0.125 for state 2. In our example, the CBS is defined such that there are $n = 4$ server periods within each task period, and the budget in each server period is $Q = 8$. The deadline is defined by $k = 8$.

First, we use the bound on the probability of longer workload accumulation as described in Section 9.4.7. We initialize the accumulation with one period after workload depletion, and β to $(0.1238, 0.0397)$, the probability of being in states 1 and 2 respectively with workload carried over from at least one task period. These probabilities are obtained from the simulation. In Figure 9.8 the

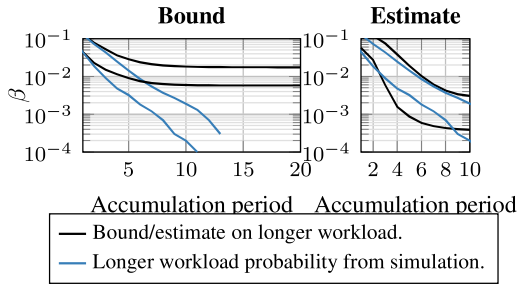


Figure 9.8: Bounds and estimates on β for the two states in black, along with probability estimates of longer accumulation histories obtained from simulation in blue. (Log scale.)

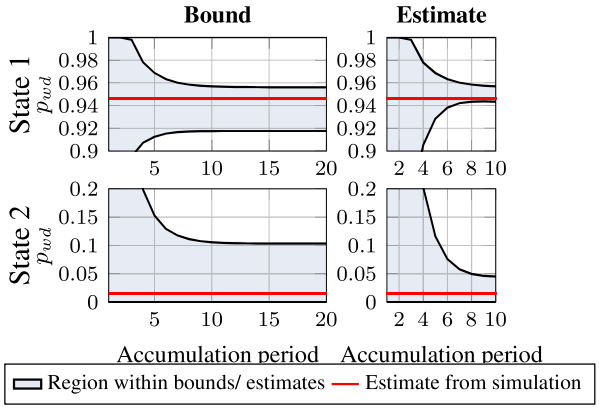


Figure 9.9: The region between the upper and lower bounds/ estimates on the per-state probability of workload depletion in the example, along with the estimates obtained from simulation in red.

obtained bounds for beta for the two states as we add accumulation periods are displayed in black. Estimated probabilities of longer accumulation histories obtained from simulation are displayed in blue.

The upper and lower bounds of the probabilities of workload depletion obtained with these values for β are shown in black in Figure 9.9, along with estimates obtained by simulation shown as red lines. The workload accumulation stops at the maximum number of task periods, 20.

In Figure 9.10 the bounds on the deadline miss probabilities during the workload accumulation process of our example are displayed. The parts of the second terms resulting from the sum over the accumulation vectors are shown as dashed. In the example this sum approaches the p_{dm} from simulation, and the pessimism comes from the pessimism in β. These bounds are compared to

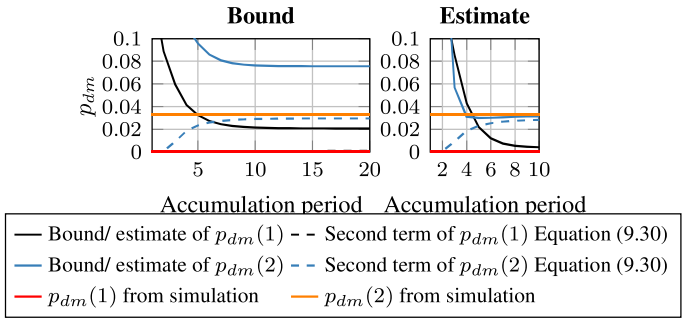


Figure 9.10: The bounds and estimates on the deadline miss probabilities during the workload accumulation process of the example, along with results from simulation.

the results from the simulation.

Using estimates on β as outlined in Section 9.4.8 in our example gives the values displayed in Figure 9.8. Here the initial values of beta are taken as an estimate of the probability of being in the second accumulation period for each state.

Using these estimates of β to find upper and lower estimates of the probabilities of workload depletion gives the results shown in Figure 9.9. The workload accumulation process stops after 10 accumulation periods.

The estimates of the deadline miss probabilities for each state during the workload accumulation process, along with the deadline miss probabilities from the simulation, are shown in Figure 9.10. The parts stemming from the second term of Equation (9.30) are dashed.

Comparing the two approaches, the bound on β is a safe overestimate, but relies on having a bound or close estimate for the first accumulation period. The estimate on β, however, is not a safe bound but can be initiated with a rough estimate in the first accumulation period. Using the estimate of β results in a lower first term in the deadline miss probability $p_{dm}^{\uparrow}$ calculation of Equation (9.30). In the example, this estimate is about 2.5 times higher than the deadline miss ratio obtained in simulation, while the bound is about 5 times higher.

9.6 Evaluation

9.6.1 Goal of the Evaluation

We aim to evaluate the proposed method of bounding and over-estimating the deadline miss probability p_{dm} for a task implementing a Markov Model, where

the execution times of the jobs vary depending on the task's state. In addition, we aim to investigate the method's sensitivity to the shape of the execution time distribution.

For this purpose, we use a test program with a known Markov Chain structure. The deadline miss probability bounds and estimates calculated with the proposed method are compared to deadline miss probabilities from simulations, and to the deadline miss ratio obtained when running a task under the Linux `SCHED_DEADLINE` scheduling policy that implements CBS based on EDF. To investigate the sensitivity to the distribution shape, test programs with two shapes of execution time distributions are implemented, one with Gaussian and one with shifted/ translated exponential distributions. The Gaussian distribution is used to evaluate the method with the conditions fulfilled. We choose exponential distributions for comparison. With a lower bound on the computation times, at which the probability density is highest and a wider tail compared to the Gaussian distribution, it is chosen as a challenge to the proposed method.

A test program with a three-state Markov chain structure is implemented that activates jobs periodically. In each job, a state transition may be performed, and different computations are performed depending on the current state. Execution time traces are obtained from running the program under FIFO scheduling. These traces are used to estimate the means and standard deviations for the three states of the Markov model. They are also used to estimate the rate and translation parameter for the exponential distribution used in the simulation for comparison with the exponential test program. The transition matrix is known from the test program implementation.

The Markov models obtained in this way are used with the methods described in Section 9.4 to calculate the deadline miss probability bounds and estimates. The maximum number of accumulation periods is set to 20. Three different configurations of server budget and period ratios are used. For each of these configurations, two relative deadlines are evaluated. The configurations are listed in Table 9.2.

The test program is run under `SCHED_DEADLINE` with the different con-

Table 9.2: Server parameters.

Q (ms)	n	k_1	k_2
100	4	7	8
120	3	7	8
90	4	9	10

figurations of server budget, period ratio, and deadline. The deadline miss ratios are calculated and compared to the estimates.

A simulation is performed, where the Markov Model is used to generate execution time samples for 10^6 task periods. Gaussian distributions with the parameters from Table 9.3 are used in the simulation for comparison with the Gaussian test case. Translated exponential distributions with translation and rate parameters from Table 9.4 are used in the simulation for comparison with the exponential test program. The workload in simulation is tracked according to Equation (9.2). Deadline miss ratios for each state and the overall deadline miss ratio are recorded.

9.6.2 Test Setup

A test program with three states has been implemented. The program executes periodically, and jobs perform a state transition according to a transition matrix, followed by a state-dependent computation with a pseudo-random variation. We evaluate two versions of the task, one where the execution times of each state are distributed according to a Gaussian and one where they are distributed according to an exponential distribution. That is, a number is generated from a Gaussian or exponential distribution with parameters depending on the current state. An iteration of additions, modulo operations, and swaps are performed in a small (100 integers) memory area, and the number of iterations is proportional to the generated number. The test program versions are implemented so that the means and standard deviations of the states' distributions are similar. The exponential distributions are shifted to accommodate this. The test program contains a deadline miss counter, and at the end of each job, a check for deadline miss is performed. The program activates 500 jobs before termination. The tests are performed on a Raspberry Pi 3B+ single-board computer with Arch Linux ARM kernel 4.14.87 patched with `PREEMPT_RT` 4.14.87-49, configured with a fully preemptible kernel and timer frequency of 100Hz. The test program is pinned to a core set up as an exclusive `cpuset`, and the scaling governor is set to performance.

9.6.3 Timing Traces and Markov models

Timing information is collected with the ftrace framework, `trace-cmd` is run, recording `sched_switch` events. The execution time is calculated as the time from the process is switched in until the time when it is switched out. Executions of the program for collecting timing information are performed under FIFO scheduling with maximum user-space priority. The traces are used

for estimating means and standard deviations of the states. The first 50 execution time measurements are excluded because in some cases there have been outliers in this region. The execution times from the traces are classified into states by cutoff points at 250 ms and 400 ms. The means and standard deviations need to be estimated. Although we know the parameters of the distributions to generate the random numbers, we do not know how these translate into execution times. The means and standard deviations are calculated incrementally. The first estimate uses the first execution time trace, then traces are added until the addition of a trace changes all means and standard deviations by less than 1%. This results in 7 traces being included in the estimate for the Gaussian distributions and 9 traces for the exponential distribution. Histograms of the traces can be seen in Figure 9.11.

The transition matrix of the test program is

$$M = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.5 & 0.3 & 0.2 \\ 0.5 & 0.4 & 0.1 \end{pmatrix}, \tag{9.35}$$

which gives stationary probabilities of 0.625, 0.25, and 0.125 for the respective states. The means and standard deviations for the states are shown in Tables 9.3 and 9.4. In Table 9.4 the rate and translation parameters of the exponential distribution are also shown. The rate parameter is σ^{-1}, and the translation parameter is $\mu - \sigma$.

Histograms of all execution times from the traces are shown in Figure 9.11. Gaussian distributions with the means and standard deviations are overlaid and scaled with the stationary probabilities.

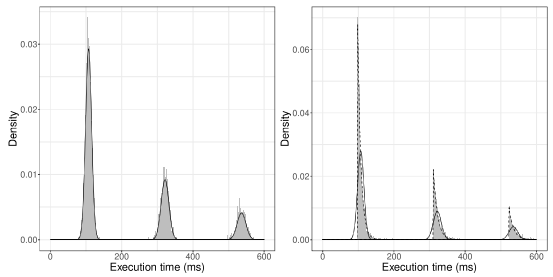
9.6.4 Evaluated Methods for Deriving a Deadline Miss Probability

In the evaluation, we compared four different methods for deriving the deadline-miss probability. Those are:

- **Linux-CBS** : A deadline-miss ratio using a Linux CBS evaluation with a `SCHED_DEADLINE`. For each evaluated combination of server budget Q,

Table 9.3: Characterization of the states of the Gaussian version traces.

State	mean (ms)	standard deviation (ms)
1	107.111	8.513
2	321.611	10.853
3	536.221	12.174



(a) **(b)**

Figure 9.11: Histograms of the execution times from the Gaussian (a) and exponential (b) version FIFO traces with the Gaussian distributions used in the method and the exponential distributions used in the simulation for (b) overlaid.

task to server period ratio n and the relative deadline to server period ratio k, 60 runs of the program under `SCHED_DEADLINE` are performed. Deadline misses after the first 50 periods of each run are recorded, as there appears to be an increased number of deadline misses in a run-in period. The bandwidth is 50%.

- **SIM**: A deadline-miss probability derived with Markov chain simulation. The obtained Markov models are used to simulate a sequence of 10^6 samples. For the Gaussian test program we use the Gaussian parameters in Table 9.3, and for the exponential test program we use the rate and translation parameters from Table 9.4. The output execution time sequence is analyzed with the different configurations of server reservation, period ratio, and deadline as listed in Table 9.2. The workload depletion ratio and the deadline miss ratio for each state are recorded.

- **Bound**: A safe bound on the deadline-miss probability, using the accumulation process defined in Section 9.5 with the upper bound on β as defined in Section 9.4.7.

- **Estimate**: An estimate of the deadline-miss probability, using the accumu-

Table 9.4: Characterization of the states of the exponential version traces.

State	mean (ms)	stddev (ms)	rate (ms^{-1})	translation (ms)
1	106.960	8.891	0.11248	98.0696
2	321.761	11.143	0.089742	310.6178
3	535.293	12.242	0.081688	523.0508

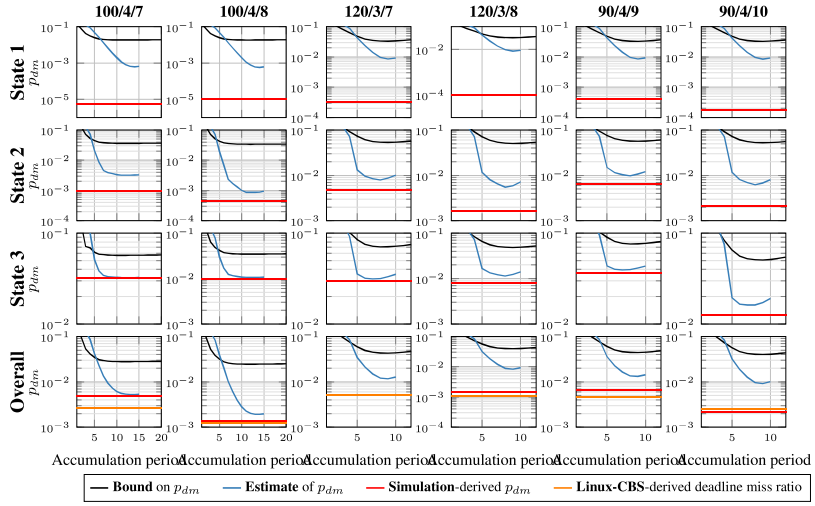


Figure 9.12: Result from test program with Gaussian emission distributions.

lation process defined in Section 9.5 with the estimate on β as defined in Section 9.4.8.

9.6.5 Results & Discussion

The deadline miss probability bounds and estimates p_{dm} obtained during the workload accumulation process are shown in Figures 9.12 and 9.13. Here, we also show deadline miss ratios of simulation with the Markov Model and the mean deadline miss ratios of the executions under `SCHED_DEADLINE`.

From the results shown in Figure 9.12, it is clear that the calculated bound adds significant pessimism compared to the estimates. The pessimism increases with 5 to 70 times when using the bounds on β compared to the estimates. Estimates and bounds are tightest for the state with the highest deadline miss probability. The pessimism in the overall case is 5 to 20 times higher compared to state 3. We also see lower pessimism for the cases with lower utilization and for shorter deadlines. In the case with β estimates and $Q/n/k$ = 100/4/7 the pessimism in state 3 is 1%, but with parameters 120/3/8 it increases to 40%.

When compared to the test with exponential execution time distributions as shown in Figure 9.13, we see that as expected the shape of the execution time distribution affects the deadline miss probability. When the assumption of Gaussian distributions does not hold, in one case (state 3 with server/ deadline parameters 100/4/7), the resulting deadline miss probability estimate, 3.11%,

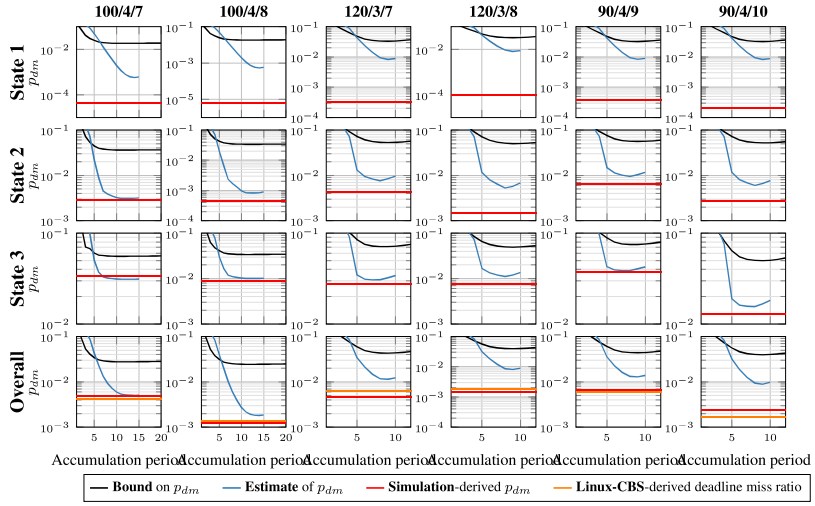


Figure 9.13: Result from test program with exponential emission distributions.

is lower than the deadline miss ratio obtained from simulation, 3.38%.

9.7 Conclusions and Future Work

In this paper, we proposed a workload accumulation scheme for upper bounding or estimating the deadline miss probability of a task executing in a Constant Bandwidth Server (CBS), having execution times modeled by an Hidden Markov Model (HMM) with Gaussian emission distributions. The deadline miss probability bounds and estimates obtained with the method are compared with deadline miss ratios of tasks running under the Linux kernel implementation of CBS. The bounds and estimates are also compared with the results from the simulation for each state separately and for the overall case. Tasks with Gaussian and exponential execution time distributions are evaluated. The comparison of the analytical and empirical results shows that the proposed methods result in a safe upper bound, except in one experiment instance. With Gaussian distributions all bounds and estimates are overestimates. The estimate for the state with the highest DMP is optimistic in one experiment instance performed on the exponential distribution. The estimate over all states is still safe in this case.

The performed evaluation has focused on assessing the pessimism introduced for a case where assumptions hold, and getting an initial estimate of the feasibility of the approach when the shape of the emission distributions differs from the Gaussian assumption. In future work, we intend to perform further

evaluation. First, we will evaluate the method with a more realistic workload. Second, we will evaluate the scalability of the approach, in comparison to the method proposed by Frías et al. [16, 5], and investigate the usefulness of the proposed method in adaptive settings. The estimates could potentially be used for monitoring changes in the deadline miss probability and adapting the Quality-of-Service (QoS) level.

Bibliography

[1] Jaume Abella, Maria Padilla, Joan Del Castillo, and Francisco J Cazorla. Measurement-based worst-case execution time estimation using the coefficient of variation. *ACM Trans. Des. Aut. of Elect. Syst. (TODAES)*, 22(4):1–29, 2017.

[2] Luca Abeni and Giorgio Buttazzo. Integrating multimedia applications in hard real-time systems. In *IEEE Real-Time Systems Symp. (RTSS)*, pages 4–13, 1998.

[3] Luca Abeni and Giorgio Buttazzo. Qos guarantee using probabilistic deadlines. In *Euromicro Conf. on Real-Time Systems (ECRTS)*, pages 242–249, 1999.

[4] Luca Abeni and Giorgio Buttazzo. Stochastic analysis of a reservation based system. In *Int. Workshop on Parallel and Distributed Real-Time Systems*, volume 1, 2001.

[5] Luca Abeni, Daniele Fontanelli, Luigi Palopoli, and Bernardo Villalba Frías. A markovian model for the computation time of real-time applications. In *IEEE international instrumentation and measurement technology conference (I2MTC)*, pages 1–6, 2017.

[6] Luca Abeni, Nicola Manica, and Luigi Palopoli. Efficient and robust probabilistic guarantees for real-time tasks. *Journal of Systems and Software*, 85(5):1147–1156, 2012.

[7] Sergey Bozhko, Georg von der Brüggen, and Björn Brandenburg. Monte carlo response-time analysis. In *IEEE Real-Time Syst. Symp. (RTSS)*, pages 342–355, 2021.

[8] Giorgio C Buttazzo, Giuseppe Lipari, Luca Abeni, and Marco Caccamo. *Soft Real-Time Systems: Predictability vs Efficiency*. Springer, 2005.

[9] Xavier Civit, Joan del Castillo, and Jaume Abella. A reliable statistical analysis of the best-fit distribution for high execution times. In *Euromicro Conf. Dig. Syst. Des. (DSD)*, pages 727–734, 2018.

[10] David D. Clark, Scott Shenker, and Lixia Zhang. Supporting real-time applications in an integrated services packet network: Architecture and mechanism. *SIGCOMM Comput. Commun. Rev.*, 22(4):14–26, 1992.

[11] Liliana Cucu-Grosjean, Luca Santinelli, Michael Houston, Code Lo, Tullio Vardanega, Leonidas Kosmidis, Jaume Abella, Enrico Mezzetti, Eduardo Quiñones, and Francisco J Cazorla. Measurement-Based probabilistic timing analysis for multi-path programs. In *Euromicro Conf. on Real-Time Systems (ECRTS)*, pages 91–101, 2012.

[12] Robert Ian Davis and Liliana Cucu-Grosjean. A survey of probabilistic schedulability analysis techniques for Real-Time systems. *LITES: Leibniz Transactions on Embedded Systems*, pages 1–53, 2019.

[13] Robert Ian Davis and Liliana Cucu-Grosjean. A survey of probabilistic timing analysis techniques for Real-Time systems. *LITES: Leibniz Transactions on Embedded Systems*, 6(1):03–1–03:60, 2019.

[14] José Luis Díaz, Daniel F García, Kanghee Kim, Chang-Gun Lee, L Lo Bello, José María López, Sang Lyul Min, and Orazio Mirabella. Stochastic analysis of periodic real-time systems. In *IEEE Real-Time Systems Symp. (RTSS)*, pages 289–300, 2002.

[15] Jose Luis Diaz, Jose Maria Lopez, Manuel Garcia, Antonio M Campos, Kanghee Kim, and Lucia Lo Bello. Pessimism in the stochastic analysis of real-time systems: Concept and applications. In *IEEE Real-Time Systems Symp. (RTSS)*, pages 197–207, 2004.

[16] Bernardo Villalba Frias, Luigi Palopoli, Luca Abeni, and Daniele Fontanelli. Probabilistic real-time guarantees: There is life beyond the i.i.d. assumption. In *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 175–186, 2017.

[17] Anna Friebe, Filip Marković, Alessandro V. Papadopoulos, and Thomas Nolte. Adaptive runtime estimate of task execution times using bayesian modeling. In *IEEE Int. Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–10, 2021.

[18] Anna Friebe, Alessandro V. Papadopoulos, and Thomas Nolte. Identification and validation of markov models with continuous emission distributions for execution times. In *IEEE Int. Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–10, 2020.

[19] Matthias Ivers and Rolf Ernst. Probabilistic network loads with dependencies and the effect on queue sojourn times. In *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, pages 280–296. Springer, 2009.

[20] M Ross Leadbetter, Georg Lindgren, and Holger Rootzén. Conditions for the convergence in distribution of maxima of stationary normal processes. *Stochastic Processes and their Applications*, 8(2):131–139, 1978.

[21] John P Lehoczky. Real-time queueing theory. In *IEEE Real-Time Systems Symp. (RTSS)*, pages 186–195, 1996.

[22] Juri Lelli, Claudio Scordino, Luca Abeni, and Dario Faggioli. Deadline scheduling in the linux kernel. *Software: Practice and Experience*, 46(6):821–839, 2016.

[23] Rui Liu, Alex F Mills, and James H Anderson. Independence thresholds: Balancing tractability and practicality in soft real-time stochastic analysis. In *IEEE Real-Time Syst. Symp. (RTSS)*, pages 314–323, 2014.

[24] Yue Lu, Thomas Nolte, Iain Bate, and Liliana Cucu-Grosjean. A statistical response-time analysis of real-time embedded systems. In *IEEE Real-Time Systems Symp. (RTSS)*, pages 351–362, 2012.

[25] Yue Lu, Thomas Nolte, Johan Kraft, and Christer Norstrom. A statistical approach to response-time analysis of complex embedded real-time systems. In *IEEE Int. Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 153–160, 2010.

[26] Yue Lu, Thomas Nolte, Johan Kraft, and Christer Norstrom. Statistical-based response-time analysis of systems with execution dependencies between tasks. In *IEEE Int. Conf. on Engineering of Complex Computer Systems*, pages 169–179, 2010.

[27] Nicola Manica, Luigi Palopoli, and Luca Abeni. Numerically efficient probabilistic guarantees for resource reservations. In *IEEE Int. Conf. on Emerging Technologies & Factory Automation (ETFA)*, pages 1–8, 2012.

[28] Pau Martí, Josep M Fuertes, Gerhard Fohler, and Krithi Ramamritham. Improving quality-of-control using flexible timing constraints: metric and scheduling. In *IEEE Real-Time Systems Symp. (RTSS)*, pages 91–100, 2002.

[29] Dorin Maxim and Liliana Cucu-Grosjean. Response time analysis for fixed-priority tasks with multiple probabilistic parameters. In *IEEE Real-Time Systems Symp. (RTSS)*, pages 224–235. IEEE, 2013.

[30] Dorin Maxim, Frank Soboczenski, Iain Bate, and Eduardo Tovar. Study of the reliability of statistical timing analysis for real-time systems. In *International Conference on Real Time and Networks Systems (RTNS)*, pages 55–64, 2015.

[31] Alex F. Mills and James H. Anderson. A multiprocessor server-based scheduler for soft real-time tasks with stochastic execution demand. In *IEEE Int. Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 207–217, 2011.

[32] Luigi Palopoli, Daniele Fontanelli, Luca Abeni, and Bernardo Villalba Frias. An analytical solution for probabilistic guarantees of reservation based soft real-time systems. *IEEE Trans. Parallel and Distributed Systems*, 27(3):640–653, 2015.

[33] Luca Santinelli, Jérôme Morio, Guillaume Dufour, and Damien Jacquemart. On the sustainability of the extreme value theory for wcet estimation. In *Int. Workshop on Worst-Case Execution Time Analysis (WCET)*, 2014.

[34] Bernardo Villalba Frias. *Bringing Probabilistic Real-Time Guarantees to the Real World*. PhD thesis, University of Trento, 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 *IEEE Real-Time Syst. Symp. (RTSS)*, pages 214–226, 2021.