Continuous-Emission Markov Models for Real-Time Applications: Bounding Deadline Miss Probabilities

Anna Friebe
anna.friebe@mdu.se
Mälardalen University (MDU), Sweden

Filip Marković
fmarkovic@mpi-sws.org
Max Planck Institute for Software Systems (MPI-SWS), Germany
Mälardalen University (MDU), Sweden

Alessandro Vittorio Papadopoulos
alessandro.papadopoulos@mdu.se
Mälardalen University (MDU), Sweden

Thomas Nolte
thomas.nolte@mdu.se
Mälardalen University (MDU), Sweden

March 20, 2023

Abstract

Probabilistic approaches have gained attention over the past decade, providing a modeling framework that enables less pessimistic analysis of real-time systems. Among the different proposed approaches, Markov chains have been shown effective for analyzing real-time systems, particularly in estimating the pending workload distribution and deadline miss probability. However, the state-of-the-art mainly considered discrete emission distributions without investigating the benefits of 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 reservation-based server, where execution times are described by a Markov model with Gaussian emission distributions. The evaluation is performed for the timing behavior of a Kalman filter for Furuta pendulum control. Deadline miss probability bounds are derived with a workload accumulation scheme. The bounds are compared to 1) measured deadline miss ratios of tasks running under the Linux Constant Bandwidth Server with SCHED_DEADLINE, 2) estimates derived from a Markov Model with

discrete-emission distributions (PROSIT), 3) simulation-based estimates, and 4) an estimate assuming independent execution times. The results suggest that the proposed method successfully upper bounds actual deadline miss probabilities. Compared to the discrete-emission counterpart, the computation time is independent of the range of the execution times under analysis, and resampling is not required.

1 Introduction

Real-time systems are commonly characterized as *hard* or *soft*. Whereas in a hard real-time system deadlines must always be met, in a soft real-time system deadline misses lead to a deterioration of the Quality of Service (QoS) [1] or Quality of Control (QoC) [2], but can be tolerated to a certain extent. This implies that the number of deadline misses is bounded such that QoS or QoC can be retained at an acceptable level [3].

Hidden Markov Models (HMMs) have been utilized to model execution times in systems with dependencies, where there is regularity in the variation of the execution times. In [4, 5], 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 [6, 7].

The missing link is that the application of HMMs with continuous emission distributions has been limited to only estimating execution times, but not the workload distributions and deadline miss probabilities.

This paper focuses on the problem of bounding the deadline miss probability of a real-time task, using HMMs with continuous emission distributions. In the literature, two concepts related to probabilistic deadlines are commonly used. The Deadline Miss Probability (DMP) is interpreted as the expected 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 [8]. This paper focuses on DMP, i.e., the long-run frequency interpretation.

More specifically, in this paper, we address the problem of upper bounding the deadline miss probability under reservation-based scheduling 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 the 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 for the overall HMM, and for each of its states separately.

The method is evaluated with a task controlling a Furuta pendulum [9–11]. The obtained bound is compared with the deadline miss ratio of the control task running under the Linux kernel implementation of CBS, `SCHED_DEADLINE` [12]. The bound is also compared to the deadline miss probability estimates using a discrete-emission distribution Markov Model [5, 13], a simulation of the fitted

continuous-time HMM, and an estimate with independence assumption.

The paper is structured as follows. In Section 2, related work is discussed. The notation used in the paper and the system model is outlined in Section 3. The analysis for upper bounding the deadline miss probability is described in Section 4, and in Section 5 the parts are combined in an overall workload accumulation process. In Section 6 the evaluation is presented and results are provided. Conclusions and future work are discussed in Section 7.

2 Related work

Davis and Cucu-Grosjean provide a comprehensive survey on probabilistic schedulability analysis techniques [8], along with a survey on probabilistic timing analysis [14].

Díaz et al. [15] presented a response time analysis for periodic tasks where execution times are independent random variables and showed that the backlog is a Markov chain.

Ivers and Ernst [16] addressed the case where execution times are dependent and proposed the use of Fréchet bounds.

Extreme Value Theory (EVT) has been applied in measurement-based statistical analysis of execution times [17–19] and response times [20–22] to find the probabilistic Worst-Case Execution Time (pWCET) and probabilistic Worst-Case Response Time (pWCRT). The pWCRT is an upper bound on the probability of exceeding a response time for every valid sequence of program executions based on finding the distribution of the extreme values. Maxim et al. [23] have shown that EVT-based methods provide sound results. EVT is applicable in cases of dependence, as long as there is stationarity [24] or extremal independence [25]. The pWRCT can be obtained from convolutions of pWCET distributions. Marković et al [26] provide an optimal down-sampling algorithm minimizing the pessimism and a convolution algorithm with better time complexity compared to previous techniques.

Real-time queuing theory [27] provides methods for analyzing the response time distribution specifically in the case of heavy traffic when utilization is close to 1. Zagalo et al. [28] provide approximations based on queuing theory for response-time distributions of a system with fixed-priority preemptive scheduling. Execution times and interarrival times are independent random variables and deadlines are implicit.

Bozhko et al. [29] proposed a response time analysis with Monte Carlo simulation for fixed-priority preemptive scheduling with execution times as independent random variables.

Safe analytical approximations of accumulated workload distributions from independent tasks under fixed-priority preemptive scheduling have been provided by Marković et al [30].

Chen et al. [31] showed how a bound for the WCDFP can be found from synchronous release combined with additional carry-in or inflation of the execution times. This analysis regards systems of sporadic tasks where execution times are

independent random variables, with fixed-priority preemptive scheduling, where jobs are aborted at their deadline. The analysis refutes and corrects the work of Maxim and Cucu-Grosjean [32].

Von der Brüggen et al. [33] 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 [34] 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 [35] further proposed the use of independence thresholds, where independence is assumed for execution times exceeding a determined threshold value.

The CBS is used in the evaluation and fulfills the requirements outlined in Section 3. It was introduced by Abeni and Buttazzo [36] and used to obtain probabilistic deadlines for QoS guarantees [37]. Analysis under CBS has been performed with execution times [38, 39] and interarrival times [40, 41] modeled with probability distributions.

Tasks with dependent execution times have been modeled as Markov chains and analyzed under CBS by Frías et al. [4, 5]. The steady-state response time distribution was calculated, while the results were compared to running the task under Linux `SCHED_DEADLINE`. The time of such analysis depends on the range of execution time values, the number of states, and the scaling factor for resampling [42].

Execution times have been also modeled as continuous Gaussian distributions in the context of emission distributions of a Markov chain [6, 7]. 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 HMMs with Gaussian emission distributions for schedulability analysis. Similarly as in the work of Frías et al. [4, 5], dependencies are explicit in HMM, and the task is running in a CBS. The CBS provides isolation from other tasks in the system so that the pending workload considers only the workload from previous jobs of the same task, instead of the workload from other tasks as in most works concerning response times.

For a general comparison of the probabilistic model with discrete- and continuous emission distributions, refer to Table 1. The shaded entries represent the contributions of this paper.

3 System model and Notation

The notation used in the paper is outlined in Table 2. We use the superscripts *, $^\uparrow$, and $^\downarrow$ to indicate the true values, upper, and lower bounds, respectively.

We will use the concept of upper bounding random variables, as defined in Definition 3.1. This is also referred to as the usual stochastic order [43] or the

Table 1: Contribution overview.

	Discrete Emission	**Continuous Emission**
Overview of the theoretical contributions		
Timing Analysis (TA)	[4, 5]	[6]
DMP analysis	[4, 5]	**This paper**
Comparison of the models and their analysis properties		
State No. identification	–	[6]
Adaptive TA	–	[7]
DMP analysis: Time and space complexity	Dependent on: - resampling scale - execution time range Requires full steady-state distribution [5].	**Independent of** - resampling scale - execution time range Iterative procedure with an adjustable complexity.

first-order statistical dominance [44]. In this paper we use the term upper bound as in Davis and Cucu-Grosjean [14].

Definition 3.1 (cf. [14, 43, 44])**.** 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 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 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:

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

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

In the derivation of workload distributions, we use convolutions as defined in Definition 3.3.

Definition 3.3. The convolution of f and g, denoted with the $*$ operator is:

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

Table 2: 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
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.

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 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 overall duration 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

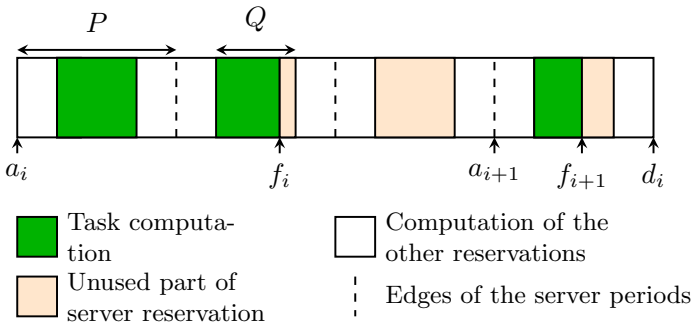

Figure 1: An illustration of the task model and the reservation-based server.

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, $p_{dm} = \mathbb{P}(\mathcal{R} > D)$.

3.2 Scheduling Algorithm

The analyzed task is the single task served by a reservation-based server. Within each server period P, the task is guaranteed to receive Q units of processing time. The fraction of the processing time dedicated to a task (bandwidth) is $B = Q/P$. The server period divides the task period evenly, i.e., $T = n \cdot P$, 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, $D = k \cdot P$. The CBS with a properly selected server period is one example of a server fulfilling these requirements, and is used in the evaluation.

An illustration of the task model and reservation-based server is shown in Fig. 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.

4 Execution time model and analysis

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 $\langle \mathbb{S}, M, \mathbb{C} \rangle$. $\mathbb{S} = \{1,2,...,S\}$ is the set of S states, $S \in \mathbb{N}$. 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. $\mathbb{C} = \{C_1, C_2, ..., C_S\}$ is the set of execution time distributions, or *emission distributions* related to the respective

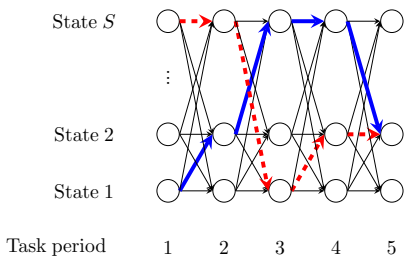

Figure 2: Illustration of the period-by-period workload accumulation sequence.

state. These are modeled as Gaussian distributions with mean μ_s, and variance σ_s^2, i.e., $C_s \sim \mathcal{N}(\mu_s, \sigma_s^2)$.

The motivation for using Gaussian distributions is based in simplicity and tractability. The representation requires only a few distribution parameters, as opposed to the discrete case where the probabilities of each execution time value need to be stored, subject to a defined scaling factor. Applicability of the Gaussian model has been demonstrated in [6], where an HMM with Gaussian emission distributions was shown to be a valid model in a video decompression use case. Also, it has been successfully used in the context of applications with dynamic behavior where adaptability is relevant [7]. Although the modeling of execution times of each HMM state as a Gaussian distribution may seem simplistic, several states with such distributions can be combined to form a more general distribution shape. The state-independence assumption does not severely hinder the model's generality due to the transition probabilities. Trivially, we can envision a model where arbitrarily many states are modeled with Normal distributions with close to zero variance while transition probabilities define the essence of the job execution time dependence.

4.2 Overview of the Proposed Approach

We will obtain an upper bound on the expected deadline miss probability of a randomly selected job of the task in a reservation-based server. The proposed method is 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 [37]:

$$v_i = \underbrace{\max(0, v_{i-1} - nQ)}_{\text{carry-in workload}} + c_i. \tag{2}$$

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 Fig. 2.

We start with zero initial pending workload and for each job arrival with carry-over workload, one task period of workload is accumulated. In the event of an idle

point with no carry-over workload, the next job arrival results in a new workload accumulation sequence at task period 1 in Fig. 2. The dashed red and solid blue lines depict two possible workload accumulation sequences resulting from job arrivals in state 2 at task period 5 from the most recent point of 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 Fig. 2, with the dashed red path we denote one possible value $h=(S,S,1,2,2)$, taken by $\mathcal{H}$.

Definition 4.1. Each arrival of a job J_i results in an accumulation sequence $h(J_i)$. We assume that the task is in state s when J_i arrives. If there is an idle point directly prior to the arrival, the resulting $h(J_i)=(s)$. If there is carry-over workload from the previous job, the resulting $h(J_i)$ is the $h(J_{i-i})$ resulting from the previous job arrival, with s appended as the last component.

In this way, each job that arrives is related to one specific h that models the accumulated workload since the last idle point, and the state of this job is always in the last component of the corresponding h.

Davis and Cucu-Grosjean [8] list three interpretations of the probability of a deadline miss: "1. As a probability with a long-run frequency interpretation equating to the expected number of missed deadlines divided by the total number of deadlines in a long (tending to infinite) time interval. 2. As the probability that a randomly selected job will miss its deadline, which is broadly equivalent to the long-run frequency interpretation. 3. As a bound on the probability that any specific job will miss its deadline." We use interpretation 1, the long-run frequency interpretation. In the steady-state this is equivalent to the expected probability that a randomly selected job misses its deadline.

Let $H(j)$ be the set of all the permutations with repetitions of accumulation sequences h of length j, of the states $\mathbb{S}=\{1,2,...,S\}$.

Definition 4.2. The Deadline Miss Probability $DMP(j)$ for the j-th job since the last depletion point is defined as

$$DMP(j)=\frac{1}{\sum_{\forall h\in H(j)} p_{in}(h)}\sum_{\forall h\in H(j)} p_{in}(h)\cdot p_{dm}(h) \tag{3}$$

where the set $H(j)$ represents accumulation sequences resulting from job arrivals at the j-th task period from the last idle point. $p_{in}(h)$ is the probability of the event where a job arrival results in the accumulation sequence h, and $p_{dm}(h)$ is the conditional deadline miss probability for jobs resulting in accumulation sequence h at their arrival.

Definition 4.3. The Deadline Miss Probability DMP is the expected deadline miss probability of a randomly selected job from the task, and it is obtained as

$$DMP=\sum_{i=1}^{\infty} DMP(i)\sum_{\forall h\in H(i)} p_{in}(h). \tag{4}$$

Note that, since each job arrival results in one specific accumulation sequence, the sum of $p_{in}(h)$ over all h equals 1.

Problem: *The sum of Eq.* (4) *has a countably infinite number of terms. This paper investigates how to find a bound for DMP with a finite number of terms.*

Introducing the main idea with an example

The main idea builds upon the observation that a job arriving j task periods after the last idle point succeeds a job arriving $j-1$ task periods after the last idle point. The probability of a job arriving with potential workload accumulation from j periods is never higher than the probability of a job arriving with workload accumulation from $j-1$ periods. This idea is not only applicable in the case of Markov models with Gaussian emission distributions, but in a more general case.

As an example, we use the periodic task τ with the execution time described by the probability mass function X

$$X = \begin{pmatrix} C_1 = 1 & C_2 = 3 \\ P_1 = 0.75 & P_2 = 0.25 \end{pmatrix} \tag{5}$$

The task is executed in a server with guaranteed computation time of $n \cdot Q = 2$ in each task period, and the deadline is implicit, $n = k$. In this example, the overall probability of completing all workload in a task period is $\frac{2}{3}$. This means that there is a $\frac{2}{3}$ chance of a job arriving with workload consisting of only its own execution time. These job arrivals result in a workload accumulation sequence of length 1, referred to as h_1 and $p_{in}(h_1) = \frac{2}{3}$. In this case, the probability of carry-over workload to the next task period is 0.25, and because of the implicit deadline this is also $p_{dm}(h_1) = DMP(1)$. The probability of a job arriving with carry-in workload of 1 remaining from the first period is $p_{in}(h_2) = \frac{2}{3} \cdot \frac{1}{4} = \frac{1}{6}$. For this job, the pending workload is 2 with probability 0.75 and 4 with probability 0.25, so $p_{dm}(h_2) = DMP(2) = 0.25$. Further, the probability of a job arriving with carry-in workload of 2, two task periods after the last idle point is $p_{in}(h_3) = \frac{1}{6} \cdot \frac{1}{4} = \frac{1}{24}$. A job arriving with this carry-in workload will miss its deadline, giving $p_{dm}(h_3) = DMP(3) = 1$. Since we know the probabilities of jobs arriving with workload from one, two or three task periods, we can calculate the probability of a job arriving with workload accumulated over a longer period as $1 - \frac{2}{3} - \frac{1}{6} - \frac{1}{4} = \frac{1}{8}$. Even in this example with a high probability of missing the deadline and relatively low probability of an idle point, most of the jobs arrive with short workload accumulation.

To upper bound the deadline miss probability, we weight the deadline miss probabilities of jobs with workload accumulated over 1, 2 or 3 task periods with the probabilities of a job arrival resulting in this workload accumulation. Then we add the probability of longer workload accumulation, assuming a deadline miss probability of 1 for these jobs. This gives an upper bound of the deadline miss probability of this example as $DMP^{\uparrow} = \frac{2}{3} \cdot \frac{1}{4} + \frac{1}{6} \cdot \frac{1}{4} + \frac{1}{24} \cdot 1 + \frac{1}{8} = \frac{3}{8} = 0.375$. The deadline miss probability of this example is $\frac{1}{3}$. Adding one more period in the summation gives a bound of the deadline miss probability of 0.352, approaching the true value.

Outline of the remainder of this section

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}. We show that $p_{in}(h)$ depends on the

probability of workload depletion in each state, the transition probabilities and the execution time distributions along h. The conditional probability of deadline miss $p_{dm}(h)$ for jobs resulting in an accumulation sequence h depends only on the execution time distributions along h. The workload accumulation 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 upper bound on *DMP* in Eq. (36).
We further enlist the **steps for deriving a safe bound** on *DMP*.

Section 4.3: To determine upper bounds on p_{dm} and p_{in} in Eq. (4) we need to find upper bounds on the pending workload distributions associated with each state and accumulation sequence. This is done in Eqs. (22) and (26).

Section 4.4: Bounds on p_{in} depend on the probability of carry-over workload p_{co} from the previous step and the transition probabilities $m_{a,b}$. 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.

Section 4.5: Bounds on p_{wd} are derived, relying on the sum of p_{in} in accumulation periods after N, denoted as β. **Section 4.6**: In this section, we derive a bound on β. This is given in Eq. (33), and is utilized for computing the lower bounds on p_{in}, p_{co} and finally $\mathcal{V}$.

Section 4.7: In this section bounds on p_{dm} are presented, using the bounds on $\mathcal{V}(h)$ (workload distribution associated with an accumulation sequence), p_{in} and β. The upper bound of p_{dm} for a state is defined in Eq. (35). The deadline miss probability of a job resulting in a specific accumulation sequence is accounted for with this sequence, even if the relative deadline is longer and the actual deadline miss does not occur until after other jobs have arrived.

The parts are tied together in the iterative workload accumulation algorithm presented along with an example in Section 5. In Fig. 3 the process is illustrated with reference to the different sections. Section 4.3 is not referenced in the figure as it is a basis for all the remaining sections.

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. The upper bounding distributions are needed for the upper bounds of p_{dm}. The upper and lower bounding distributions are used to derive bounds on p_{in}, p_{co}, β and p_{wd}. As an example from Fig. 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

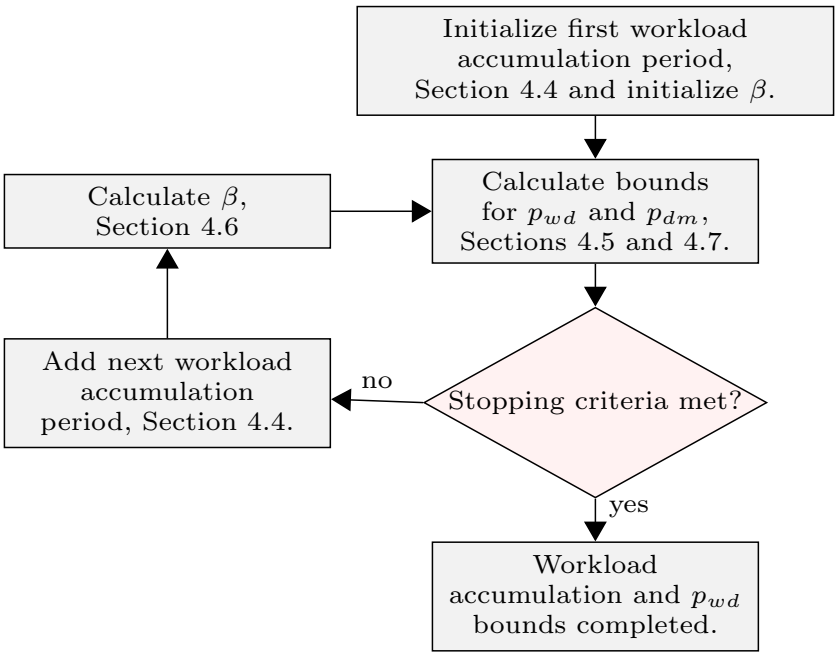

Figure 3: The workload accumulation process.

dashed red, $h=(S,S,1,2,2)$.

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

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 Fig. 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 $\tilde{h}=[1,2,...,2]$. 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 accumulation vectors 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 accumulation sequences where ordering is taken into account. For a fixed number of states S, the number of accumulation vectors to consider increases with the number of accounted periods as $\mathcal{O}(N^{S-1})$.

Recalling Definition 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 Defini-

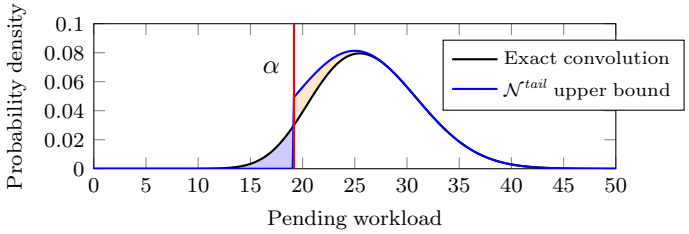

Figure 4: Illustration of a convolution result with an upper bounding partial Gaussian distribution.

tion 3.2) upper bounds the conditional pending workload distribution. An illustration is shown in Fig. 4. The black curve illustrates the exact convolution result. When we replace it with the partial Gaussian distribution (the blue curve and the red line), the probabilities of lower workloads (the blue area) are moved to higher workloads (the orange area), which gives an upper bound.

Theorem 1. *The conditional pending workload distribution $\mathcal{V}_{\tilde{h}}$ associated with each state s and accumulation vector $\tilde{h}$ is upper bounded by $\mathcal{N}^{tail}\Big(\mu(\tilde{h}),\sigma^2(\tilde{h}),\alpha(\tilde{h},s)\Big)$.*

We prove this by induction. For clarity, we state Lemma 2 for the base case, and Lemma 3 for the inductive step.

Lemma 2. *The partial Gaussian distribution $\mathcal{N}^{tail}\big(\mu_s,\sigma_s^2,0\big)$ 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, $\mathcal{N}^{tail}\big(\mu_s,\sigma_s^2,0\big)$ is an upper bound. □

Consider a 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 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 Eqs. (6) and (7) we define $\mu(\tilde{h}_{+s})$ and $\sigma^2(\tilde{h}_{+s})$. Eqs. (8) and (9) are used to simplify the expression of the starting value $\alpha(\tilde{h}_{+s},s)$ of the resulting upper bounding distribution, defined in Eq. (10). 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. Eq. (9) defines $K(\tilde{h},s_p)$, the normalization factor needed for the conditional probability calculation. In the proof, the execution time distribution C_s is convolved with an upper bound of the carry-over workload distribution to get a bound on the pending workload distribution. The upper bound of the carry-over workload is the part extending past the task period of the

upper bounding workload distribution in s_p with $\tilde{h}$. $K(\tilde{h},s_p)^{-1}$ is the integral of this part, to get a probability distribution integrating to one.

$$\mu(\tilde{h}_{+s}) = \mu_s + \sum_{i=1}^{S} \tilde{h}[i] \cdot (\mu_i - n \cdot Q) \tag{6}$$

$$\sigma^2(\tilde{h}_{+s}) = \sigma_s^2 + \sum_{i=1}^{S} \tilde{h}[i] \cdot \sigma_i^2 \tag{7}$$

$$\alpha_\Delta(\tilde{h},s_p) = \max(0, \alpha(\tilde{h},s_p) - n \cdot Q) \tag{8}$$

$$K(\tilde{h},s_p) = \left[\Phi\left(\frac{\mu(\tilde{h}) - n \cdot Q - \alpha_\Delta(\tilde{h},s_p)}{\sigma(\tilde{h})} \right) \right]^{-1} \tag{9}$$

$$\alpha(\tilde{h}_{+s},s) = \begin{cases} 0 & \tilde{h} = \mathbf{0} \\ sf^{-1}(\frac{1}{K(\tilde{h},s_p)}, \mu(\tilde{h}_{+s}), \sigma^2(\tilde{h}_{+s})) & \tilde{h} \neq \mathbf{0} \end{cases}. \tag{10}$$

Lemma 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. This can be formally expressed with the following term, $\mathcal{N}^{tail}\Big(\mu(\tilde{h}) - n \cdot Q \ , \ \sigma^2(\tilde{h}) \ , \ \max(0, \alpha(\tilde{h}, s_p) - n \cdot Q)\Big)$.

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 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:

$$\mu_R(z) = \frac{(z - \mu_s) \cdot \sigma^2(\tilde{h}) + (\mu(\tilde{h}) - n \cdot Q) \cdot \sigma_s^2}{\sigma_s^2 + \sigma^2(\tilde{h})} \tag{11}$$

$$\sigma_R^2 = \frac{\sigma_s^2 \cdot \sigma^2(\tilde{h})}{\sigma_s^2 + \sigma^2(\tilde{h})} \tag{12}$$

$$\mu_{\Sigma\Delta} = \mu_s + \mu(\tilde{h}) - n \cdot Q \tag{13}$$

$$\sigma_\Sigma^2 = \sigma_s^2 + \sigma^2(\tilde{h}). \tag{14}$$

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

$$\begin{aligned}
&\int_{-\infty}^{\infty} f\big(z-x|\mu_s,\sigma_s^2\big)\cdot f^{tail}\Big(x|\mu(\tilde{h})-n\cdot Q,\sigma^2(\tilde{h}),\alpha_\Delta\Big)\mathrm{d}x\\
&=K(\tilde{h},s_p)\int_{\alpha_\Delta}^{\infty} f(z-x|\mu_s,\sigma_s^2)\cdot f(x|\mu(\tilde{h})-nQ,\sigma^2(\tilde{h}))\mathrm{d}x\\
&=K(\tilde{h},s_p)\cdot f\big(z|\mu_{\Sigma\Delta},\sigma_\Sigma^2\big)\cdot\int_{\alpha_\Delta}^{\infty} f\big(x|\mu_R(z),\sigma_R^2\big)\mathrm{d}x, \qquad (15)
\end{aligned}$$

where the last step isolated the part of the expression that is independent of x. We recognize the integral in the second factor of Eq. (15) as the survival function or 1-CDF at α_Δ of $\mathcal{N}(\mu_R(z),\sigma_R^2)$. This is monotonically increasing and converges to 0 as z goes to $-\infty$, and it converges 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, $\mathcal{N}^{tail}(\mu_{\Sigma\Delta},\sigma_\Sigma^2,\alpha(\tilde{h}_{+s},s))$ from $\alpha(\tilde{h}_{+s},s)$. An illustration is provided in Fig. 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. Finding $\alpha(\tilde{h}_{+s},s)$ gives:

$$K(\tilde{h},s_p)\cdot\int_{\alpha(\tilde{h}_{+s},s)}^{\infty} f\big(x|\mu_{\Sigma\Delta},\sigma_\Sigma^2\big)\,\mathrm{d}x=1. \qquad (16)$$

As we know that the result of the convolution integrates to one, this shows that the two regions described and illustrated in Fig. 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:

$$\mu(\tilde{h}_{+s})=\mu_{\Sigma\Delta} \qquad (17)$$

$$\sigma^2(\tilde{h}_{+s})=\sigma_\Sigma^2 \qquad (18)$$

$$\alpha(\tilde{h}_{+s},s)=s\cdot f^{-1}\left(\frac{1}{K(\tilde{h},s)},\mu_{\Sigma\Delta},\sigma_\Sigma^2\right). \qquad (19)$$

This concludes our proof. □

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

$$\alpha_\Delta(\tilde{h})=\max(0,\max_{\forall s_p}\alpha(\tilde{h},s_p)-n\cdot Q). \qquad (20)$$

We use this instead of Eq. (8) in Eqs. (9) and (10). With these lemmas, we are ready to prove Theorem 1.

Proof. We prove this by induction.
Base case: For the task period after workload depletion, this follows by Lemma 2.

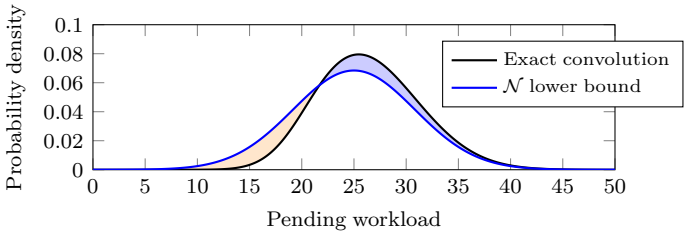

Figure 5: An illustration of a convolution result and the Gaussian distribution that forms a lower bound.

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 3 and taking the maximum α in Eq. (20). □

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 Fig. 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.

4.4 Bounds on the Joint Probability of Being in 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 joint probability of the events being in s and a job arrival resulting in the the accumulation vector $\tilde{h}$ $p^{\downarrow}_{in}(s,\tilde{h})$ 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 which 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 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

$$p_{in}^{\downarrow}(s,\tilde{h}) = \sum_{s_p=1}^{S} \xi(s_p) \cdot p_{wd}^{\downarrow}(s_p) \cdot m_{s_p,s} \tag{21}$$

$$p_{in}^{\uparrow}(s,\tilde{h}) = \sum_{s_p=1}^{S} \xi(s_p) \cdot p_{wd}^{\uparrow}(s_p) \cdot m_{s_p,s}. \tag{22}$$

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 joint probability of the events being in s and a job arrival resulting in $\tilde{h}_{+s}$ 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 being in the state with this accumulation vector times the probability that the conditional pending workload exceeds the available processor time in a task period. We define the random variables $X \sim \mathcal{V}_{\tilde{h}}^{\downarrow}$ and $Y \sim \mathcal{V}_{\tilde{h}}^{\uparrow}$. The probability of carry-over into the next period is bounded by $p_{co}^{\downarrow}(s,\tilde{h})$ and $p_{co}^{\uparrow}(s,\tilde{h})$, further calculated as:

$$p_{co}^{\downarrow}(s,\tilde{h}) = p_{in}^{\downarrow}(s,\tilde{h}) \cdot \mathbb{P}(X > n \cdot Q) \tag{23}$$

$$p_{co}^{\uparrow}(s,\tilde{h}) = p_{in}^{\uparrow}(s,\tilde{h}) \cdot \mathbb{P}(Y > n \cdot Q) \tag{24}$$

with $\mathcal{V}_{\tilde{h}}^{\downarrow}$ given as $\mathcal{N}(\mu(\tilde{h}),\sigma^2(\tilde{h}))$, and $\mathcal{V}_{\tilde{h}}^{\uparrow}$ as $\mathcal{N}^{tail}(\mu(\tilde{h}),\sigma^2(\tilde{h}),\alpha(\tilde{h}))$.

The joint probability of being in s and a job arrival resulting in $\tilde{h}$ is respectively lower and upper bounded by:

$$p_{in}^{\downarrow}(s,\tilde{h}_{+s}) = \sum_{s_p=1}^{S} p_{co}^{\downarrow}(s_p,\tilde{h}) \cdot m_{s_p,s} \tag{25}$$

$$p_{in}^{\uparrow}(s,\tilde{h}_{+s}) = \sum_{s_p=1}^{S} p_{co}^{\uparrow}(s_p,\tilde{h}) \cdot m_{s_p,s}. \tag{26}$$

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 Eqs. (21) and (22), 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 Eq. (21), the sum $p_{in}^{\downarrow\Sigma}$ of the lower bounds on the probabilities associated with all accounted accumulation vectors would be lower than

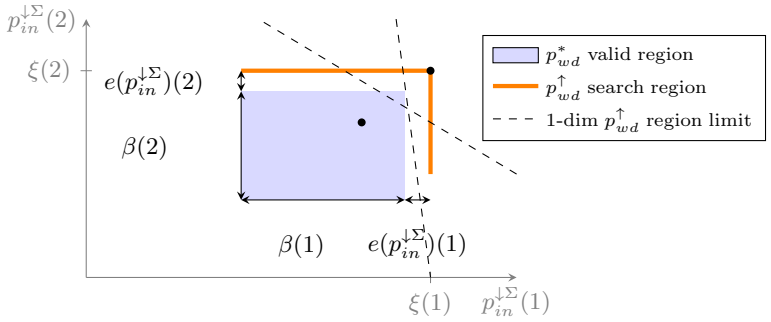

Figure 6: An illustration of the possible valid region of $p_{in}^{\downarrow\Sigma}$ for two states, if the true probabilities of workload depletion would be used as $p_{wd}^{\downarrow}$ in Eq. (21).

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:

$$p_{in}^{\downarrow\Sigma}(s,p_{wd}) = \sum_{i=1}^{N} \sum_{\tilde{h}\in(s,i)} p_{in}^{\downarrow}(s,\tilde{h}) \leq \xi(s), \quad \forall s. \tag{27}$$

We define $\beta(s)_N$ as the probability of being in s with workload accumulation past N.

$$\beta(s)_N = \sum_{i=N+1}^{\infty} \sum_{\tilde{h}\in(s,i)} p_{in}(s,\tilde{h}) \tag{28}$$

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 Eq. (21) that would give values of $p_{in}^{\downarrow\Sigma}$ in the blue area of Fig. 6.

Had the true workload depletion probability p_{wd}^{*} been known and input as $p_{wd}^{\uparrow}$ in Eq. (22), 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 Eq. (29):

$$p_{in}^{\uparrow\Sigma}(s,p_{wd}) = \sum_{i=1}^{N} \sum_{\tilde{h}\in(s,i)} p_{in}^{\uparrow}(s,\tilde{h}) \geq \xi(s) - \beta(s)_N, \forall s. \tag{29}$$

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 workload depletion probabilities as $p_{wd}^{\uparrow}$ in Eq. (22) the resulting $p_{in}^{\uparrow\Sigma}$ would be in the range depicted as green in Fig. 7. This allows us to find a bound of the true probabilities of workload depletion to values that are mapped within both the blue region of Fig. 6 and the green region of Fig. 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 Fig. 6.

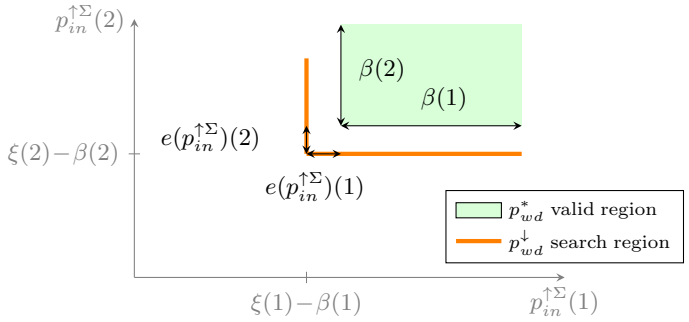

Figure 7: An illustration of the possible valid region of $p_{in}^{\uparrow\Sigma}$ for two states, if the true probabilities of workload depletion would be used as $p_{wd}^{\uparrow}$ in Eq. (22).

Theorem 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 Eqs. (21), (23) and (25). Combined with Eq. (27) it follows that $p_{in}^{\downarrow\Sigma}(s)$ is also a linear combination of p_{wd} for all states, which for some positive factors $A_{i,s}$ is:

$$p_{in}^{\downarrow\Sigma}(s,p_{wd}) = \sum_{i=1}^{S} A_{i,s} \cdot p_{wd}(i) \tag{30}$$

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:

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

For the lowest $\delta_{s,j}$, the first plane we encounter along the line, we have $p_{in}^{\downarrow\Sigma}(i) \leq \xi(i), \forall i \neq s$.

Because the true p_{wd}^{*} gives $p_{in}^{\downarrow\Sigma}(s) \leq \xi(s)$ 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 $p_{in}^{\downarrow\Sigma}(s,p_{wd}) = \xi(s)$, including the point with equality for all s: the upper right corner on the orange lines in Fig. 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 illustrated in Fig. 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 Fig. 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 $p_{in}^{\uparrow\Sigma}$ along the orange

lines of Fig. 7. By using the lower bound from Fig. 7 to determine the endpoints of the orange sections in Fig. 6 and the upper bound from Fig. 6 to determine the endpoints of the orange sections in Fig. 7 $e(p_{in}^{\downarrow\Sigma})$ and $e(p_{in}^{\uparrow\Sigma})$ 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 $p_{in}^{\downarrow\Sigma}(s)$ and $p_{in}^{\uparrow\Sigma}(s)$ depend linearly on all $p_{wd}(s)$, we only need to consider the endpoints of the orange sections.

4.6 Probability Bound on Longer Workload Accumulation

In this section, we derive a bound for β, the sum of p_{in} in task periods beyond N, as defined in Equation (28). 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.

$$\beta(s)_N \leq \beta(s)_{N-1} - \sum_{\tilde{h}\in(s,N)} p_{in}^{\downarrow}(s,\tilde{h}) = \beta(s)_N^{\uparrow a} \tag{31}$$

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

$$\beta(s)_N \leq \xi(s) - \sum_{i=1}^{N} \sum_{\tilde{h}\in(s,i)} p_{in}^{\downarrow}(s,\tilde{h}) = \beta(s)_N^{\uparrow b} \tag{32}$$

Thus, given a safe bound of β in one accumulation period, we can obtain safe bounds for the following period as the minimum of right-hand sides of Inequalities 31, and 32.

$$\beta(s)_N^{\uparrow} = \min(\beta(s)_N^{\uparrow a}, \beta(s)_N^{\uparrow b}) \tag{33}$$

4.7 Upper Bounding the Deadline Miss Probability

We can then calculate an upper bound on the expected deadline miss probability of a randomly selected job as defined in Eq. (4). The deadline miss probability $p_{dm}(s,\tilde{h})$ of a job conditioned on the events being in state s and the job arrival resulting in the accumulation vector $\tilde{h}$ is upper bounded by $p_{dm}^{\uparrow}(s,\tilde{h})$. This bound regards the last job of accumulation sequences h ending in state s and corresponding to $\tilde{h}$. The bound is the probability that a random variable $Y \sim \mathcal{V}_{\tilde{h}}^{\uparrow}$ exceeds the amount of time $k \cdot Q$ allocated to the job, and it can be computed as:

$$p_{dm}^{\uparrow}(s,\tilde{h}) = \mathbb{P}(Y > k \cdot Q) \tag{34}$$

where the distribution $\mathcal{V}_{\tilde{h}}^{\uparrow}$ can be computed as $\mathcal{N}^{tail}(\mu(\tilde{h}),\sigma^2(\tilde{h}),\alpha(\tilde{h},s))$, and it upper bounds the actual distribution $\mathcal{V}_{\tilde{h}}$ as per Theorem 1. When randomly selecting a job, the probability of drawing a job executing in state s with workload accumulation captured by $\tilde{h}$ is the joint probability of being in s with $\tilde{h}$. In Section 4.4 we derived the upper bound of this probability as $p_{in}^{\uparrow}(s,\tilde{h})$. We derive the bound of the deadline miss probability conditioned on being in a state s by

considering all events of being in s with $\tilde{h}$ up until N accumulation periods. For longer accumulation vectors we set p_{dm} to 1. The probability of drawing a job with a longer accumulation vector when randomly selecting a job is the probability of a job arrival in s resulting in a longer accumulation vector, and is upper bounded by $\beta^{\uparrow}_{N}(s)$. The probability of a job arrival in s is the stationary probability $\xi(s)$. The upper bound on the deadline miss probability in a state is:

$$p^{\uparrow}_{dm}(s) = \frac{\beta(s)^{\uparrow}_{N}}{\xi(s)} + \frac{\sum_{i=1}^{N}\sum_{\tilde{h}\in(s,i)} p^{\uparrow}_{in}(s,\tilde{h}) p^{\uparrow}_{dm}(s,\tilde{h})}{\xi(s)}. \tag{35}$$

Theorem 5. *The deadline miss probability DMP is upper-bounded by $p^{\uparrow}_{dm}$, i.e., $DMP \leq p^{\uparrow}_{dm}$, where*

$$p^{\uparrow}_{dm} = \sum_{\forall s}\Big(\beta(s)^{\uparrow}_{N} + \sum_{i=1}^{N}\sum_{\tilde{h}\in(s,i)} p^{\uparrow}_{in}(s,\tilde{h}) p^{\uparrow}_{dm}(s,\tilde{h})\Big). \tag{36}$$

Proof. Eq. (34) is an upper bound on the deadline miss probability conditioned on the events being in state s with accumulation vector $\tilde{h}$, as $\mathcal{N}^{tail}(\mu(\tilde{h}),\sigma^2(\tilde{h}),\alpha(\tilde{h},s))$ is an upper bound on the workload distribution as per Theorem 1.

Eq. (35) upper bounds the expected deadline miss probability of a randomly selected job conditioned on being in state s. For accumulation vectors $\tilde{h}$ up until length N, Eq. (34) is utilized to upper bound $p_{dm}(s,\tilde{h})$, and $p_{in}(s,\tilde{h})$ upper bounds the probability of randomly selecting a job in state s with $\tilde{h}$. $\beta(s)^{\uparrow}_{N}$ upper bounds the probability of randomly selecting a job in s with longer accumulation vectors, and $p_{dm}(s,\tilde{h})$ is upper bounded by 1 in this case. From the definition of conditional probability we divide by $\xi(s)$.

Eq. (36) applies the law of total probability on Eq. (35) over all the states s. □

5 Iterative workload accumulation

We propose an iterative approach where workload periods are successively accumulated. The process is illustrated in Fig. 3 and is ended when one of the following 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 for each state 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 Fig. 6, and by the white space between the green area

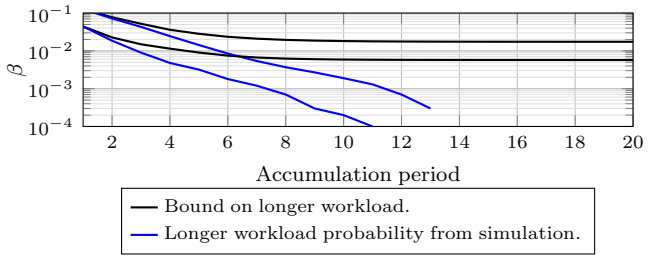

Figure 8: Bounds on β for the two states in black, along with probability estimates of longer accumulation histories obtained from simulation in blue. (Log scale.)

and the orange lines in Fig. 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 the workload accumulation process.

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 $n=4$ and $Q=8$. The deadline is defined by $k=8$.

First, we use the bound on the probability of longer workload accumulation as described in Section 4.6. 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 Fig. 8 the obtained bounds for β 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 Fig. 9, along with estimates obtained by simulation shown as red lines. The workload accumulation stops at the maximum number of task periods, 20.

In Fig. 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 the results from the simulation.

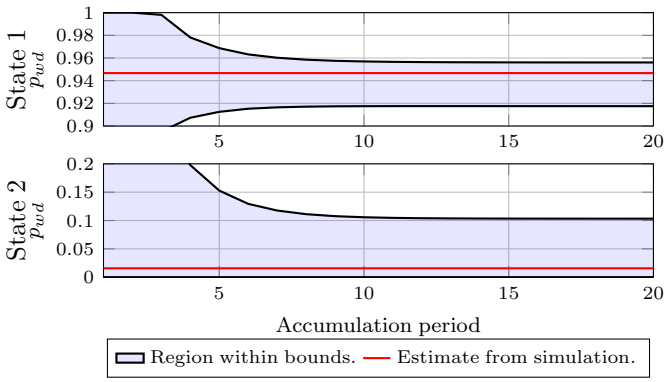

Figure 9: The region between the upper and lower bounds on the per-state probability of workload depletion in the example, along with the estimates obtained from simulation in red.

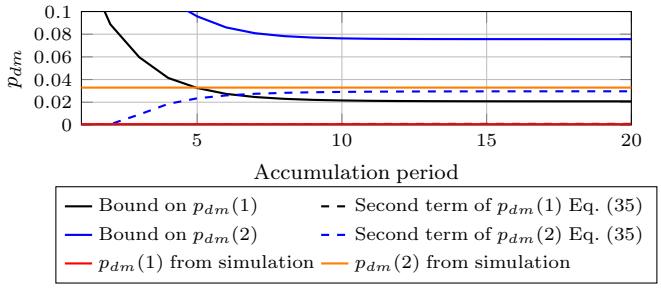

Figure 10: The bounds on the deadline miss probabilities during the workload accumulation process of the example, along with results from simulation.

6 Evaluation

6.1 Goal of the Evaluation

We aim to evaluate the proposed method of bounding the deadline miss probability p_{dm} for a task controlling a Furuta pendulum [10], i.e., a rotary inverted pendulum. The dynamics of the Furuta pendulum are simulated by another task. The arm and pendulum angles are retrieved from the pendulum simulator, which provides an asynchronous TCP server. These values are fed into a Kalman filter [45], that estimates angles and angular velocities around an upright position, and these estimates are used in a PD controller for stabilizing the pendulum in the upright position at angle 0 of the arm. The resulting control signal is then sent to the simulator. An HMM is fitted to the execution time trace, as outlined in [6]. From the fitted HMM we calculate the bound on the p_{dm} for each state according to Eq. (35), and the overall bound according to Eq. (36), which is compared to:

- **Linux-CBS**: The empirical deadline miss ratio (DMR) under Linux `SCHED_DEADLINE`.
- **Sim-Cont**: DMR estimated from entering execution times generated from the

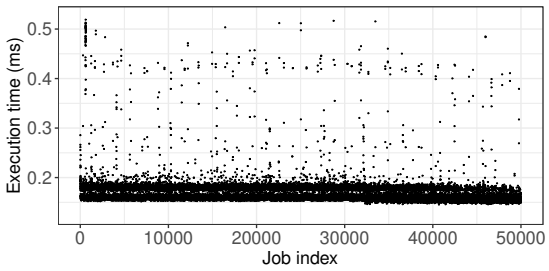

Figure 11: The execution time trace used for fitting the HMM.

fitted Markov Model in a CBS simulator.

- **Ind**: DMR, assuming independence, that is, entering the randomly reordered execution times in a CBS simulator.
- **PROSIT**: Deadline miss probability obtained with PROSITool [13], using a discrete emission distribution HMM fitted with PROSITool from the same execution time trace.

The bound for the state with the highest p_{dm} is compared to the deadline miss ratio of this state from the CBS simulator, as the other methods do not provide per-state estimates.

With these experiments, the applicability of the method for a small, yet realistic, use case is illustrated.

6.2 Test Setup

The tests are run on a Raspberry Pi 3B+ single-board computer with Raspberry Pi OS patched with `PREEMPT_RT`. The control program contains a square root implementation of a Kalman filter [45] and runs at a control frequency of 500Hz. It is scheduled with the Linux CBS implementation `SCHED_DEADLINE` and is pinned to a core set up as an exclusive `cpuset`. The simulator is run at the same frequency, with the highest priority FIFO scheduling, and is pinned to another core, similarly set up as an exclusive `cpuset`. The simulator's TCP server is run in a separate thread. The scaling governor is set to `performance` for all cores while USB Ethernet and WiFi were disabled during the tests.

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. Execution time traces are obtained from running the control program with a high bandwidth, long period setting of `SCHED_DEADLINE`, where each job finishes within the period.

The timing trace recorded from 50 000 jobs of the control task is shown in Fig. 11. The sequence exhibits a run-in period with a higher proportion of execution times at 0.5ms. Therefore, we use the sequence starting from job 2000 for fitting the HMM. The density distribution of the execution times and the autocorrelation of the execution times starting from job 2000 are shown in Fig. 12.

Three different configurations of server budget and period ratios are evaluated:

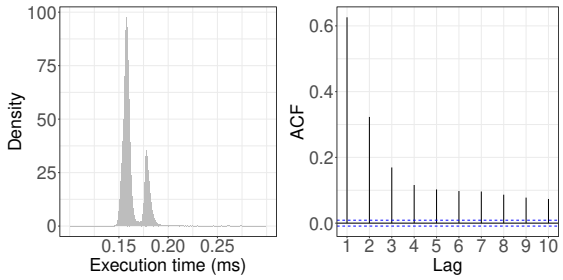

Figure 12: The density distribution of the execution time trace, and the autocorrelation, excluding the first 2000 jobs.

1. $Q = 0.06$ ms, $n = 5$, $k_1 = 8$, $k_2 = 10$,
2. $Q = 0.07$ ms, $n = 4$, $k_1 = 6$, $k_2 = 8$, and
3. $Q = 0.08$ ms, $n = 4$, $k_1 = 6$, $k_2 = 8$.

For each configuration, two relative deadlines are evaluated.

The test program is run under `SCHED_DEADLINE` with different server budget, period ratio, and deadline configurations. The program contains a check for missed deadlines and outputs the number for each 500-job-interval.

6.3 Markov Model

Starting from 10 initial states, the method described in [6] identifies an 8-state HMM. The means, standard deviations, and stationary probabilities of the states are listed in Table 3, and the transition matrix in Eq. (37). State 3 has the highest mean and standard deviation. The stationary probability of this state is low, 0.7%, but the probability of staying in this state once it is reached is 63%, increasing the DMP.

$$\begin{pmatrix} .739 & .051 & .002 & .003 & .162 & .001 & .041 & .001 \\ .056 & .350 & .012 & .000 & .523 & .008 & .051 & .000 \\ .000 & .310 & .633 & .003 & .000 & .044 & .010 & .000 \\ .006 & .000 & .002 & .408 & .004 & .054 & .000 & .526 \\ .000 & .038 & .002 & .003 & .834 & .001 & .121 & .000 \\ .000 & .000 & .004 & .681 & .063 & .225 & .000 & .028 \\ .377 & .011 & .001 & .000 & .500 & .000 & .107 & .003 \\ .009 & .001 & .002 & .296 & .000 & .038 & .001 & .654 \end{pmatrix} \tag{37}$$

6.4 Evaluated Methods

In the evaluation, we compared five different methods:

- **Linux-CBS** : A deadline-miss ratio resulting from experiments. Linux `SCHED_DEADLINE` is configured with each setting of server budget Q, task to server period ratio n and the relative deadline to server period ratio k. The task period is kept at $2ms$, so the bandwidth varies between the tests. For each configuration, 10 runs of the 50000-job task under `SCHED_DEADLINE` are performed. Deadline misses after first 2000 jobs of each run are recorded and used to calculate the deadline-miss ratio.

- **Sim-Cont**: A deadline-miss probability derived with Markov chain simulation. The fitted continuous-emission Markov model with the parameters from Table 3 and transition matrix Eq. (37) is used to simulate a sequence of 10^6 samples. The output execution time sequence is analyzed with the different configurations of server reservation, period ratio, and deadline.

- **Ind**: A deadline-miss probability derived from a sequence of 10^6 samples, randomly sampled from the execution time sequence used in the fitting process. The obtained sequence is analyzed as for **Sim-Cont**.

- **PROSIT**: A deadline-miss probability derived with PROSITool [13]. A discrete-emission HMM with 6 states and scaling factor for resampling of 10 μs is fitted to the execution time trace. This model is used with the different CBS configurations in PROSIT's solver for the calculation of steady-state deadline-miss probabilities.

- **Bound**: A bound on the deadline-miss probability, calculated according to Section 4 with the fitted continuous-emission Markov model with the parameters from Table 3 and transition matrix Eq. (37). The initial β values for the first accumulation period are retrieved from simulation, and the maximum number of accumulation periods is set to 10.

6.5 Results & Discussion

The deadline miss probability bounds and estimates p_{dm} obtained during the workload accumulation process are shown in Fig. 13, together with deadline miss ratios (DMRs) of simulation with the Markov Model, the average DMRs of the executions under `SCHED_DEADLINE`, and under PROSITool.

Table 3: Characterization of the states of the fitted HMM.

State	Mean (ms)	Standard Deviation (ms)	Stationary Prob.
1	0.178	0.002	0.128
2	0.178	0.012	0.045
3	0.323	0.091	0.007
4	0.158	0.003	0.086
5	0.159	0.002	0.509
6	0.169	0.007	0.014
7	0.181	0.003	0.078
8	0.153	0.002	0.133

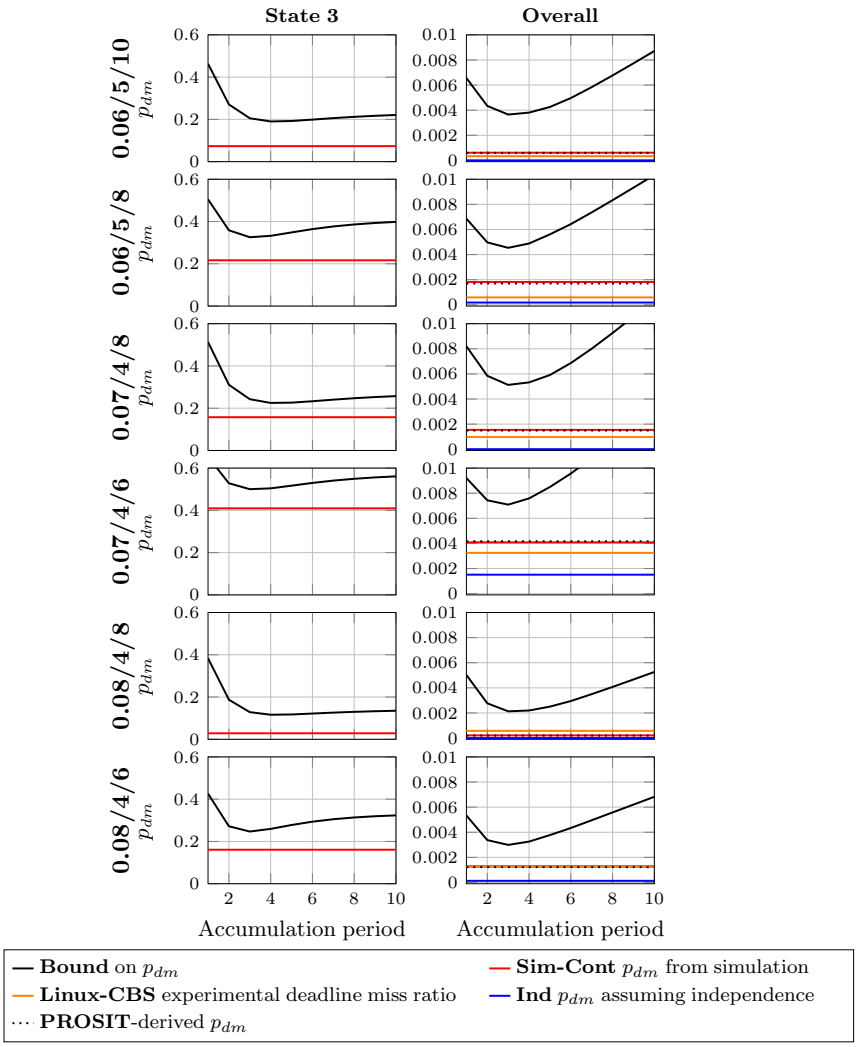

Figure 13: Result from control task analysis.

The experimental DMRs are lower than those from HMM simulation and derived from PROSIT, with the exception of the CBS configuration 0.08/4/8. In this case the p_{dm} from **Linux-CBS** is 0.058% compared to 0.021% for **Sim-Cont** and 0.0020 for **PROSIT**. This exception may be due to chance. It may also be the case that there is a low probability state that is not captured in the Markov Models.

The HMM simulation **Sim-Cont** estimates are close to the **PROSIT** results, which shows that Gaussian emission distributions are a useful approximation in this case.

Comparing the HMM simulation **Sim-Cont** and the resulting **Bound**, the overall bounds are 1.75 to 10 times higher than the simulation results. For the

state with the highest p_{dm}, the bounds are 1.4 to 3.9 times higher. The bounds are tighter for higher utilization and shorter relative deadlines.

When fitting an HMM in PROSIT, the number of states and the scaling factor need to be provided. A number of combinations of these parameters were tried, and for 6 states and scaling factor 10 μs, 4 out of 6 states passed the PROSIT independence tests, the largest part found in the limited manual exploration. The resampling introduces some pessimism. Other fitted PROSIT models have resulted in tighter estimates or optimistic results. The range of execution time values in the input trace and the scaling factor affect the calculation time for PROSIT significantly. Decreasing the scaling factor of the 6-state model from 10 to 1 μs increases the computation time by a factor 3000, from less than 0.5s to about 20 minutes on our platform. With the continuous approach, there is no resampling concept, and the calculation time is independent of the range of execution time values. A direct comparison of the computation time of the proposed bound and PROSIT has not been feasible. The Python implementation of the bound calculation runs the first 4 accumulation periods in less than a second and 10 accumulation periods in around one minute.

We note that in the evaluated use case, the tightest bound is reached already at 3-4 accumulation periods. While this allows for short computation times, the proposed reduction in the number of considered accumulation sequences is unnecessary if we consider a very low number of accumulation periods.

Simulations and bounds of the state with the highest p_{dm} show results 50-100 times higher than the overall p_{dm}. While this should not be conflated with the Worst-Case Deadline Failure Probability, we believe that the concept of workload distribution per state is useful. In future work, we aim to develop the accumulation sequence approach relating to the probability of consecutive deadline misses.

7 Conclusions and future work

In this paper, we proposed a workload accumulation scheme for upper bounding the deadline miss probability of a task executing in a Constant Bandwidth Server (CBS), modeled by a Hidden Markov Model (HMM) with Gaussian emission distributions. Such a model allows for higher automation in the fitting process, compared to the discrete case, without needing to specify the number of states and the scaling factor. The time required to obtain the bound is independent of the range of execution times in the analyzed sequence and the scaling factor. Furthermore, a bound is obtained early in the process and is iteratively improved.

The presented approach was evaluated over a control task of a Furuta pendulum. The deadline miss probability bounds obtained with the method are compared to the deadline miss ratios of the task running under the Linux kernel implementation of CBS. The bounds are also compared to the deadline miss probabilities derived with a discrete emission-HMM [4, 5, 13], results from the simulation with the fitted HMM, and a deadline miss probability estimate assuming independence. All derived bounds in the evaluated case are safe compared to simulation results, i.e., the overall bounds are 1.75 to 10 times higher, and in the

state with the highest deadline miss probability the bounds are 1.4 to 3.9 times higher. All bounds are also higher compared to experimental results.

In future work, we aim to extend the method to evaluate probabilities of missing several consecutive deadlines. The bounds could potentially be used for monitoring changes in the deadline miss probability and adapting the Quality-of-Service (QoS) level.

Acknowledgments

This work was supported by the Swedish Research Council (VR) via the projects "Practical Probabilistic Timing Analysis of Real-Time Systems (PARIS)" and "Pervasive Self-Optimizing Computing Infrastructure (PSI)", and by the Knowledge Foundation (KKS) via the project FIESTA.

References

[1] D. D. Clark, S. Shenker, and L. Zhang, "Supporting real-time applications in an integrated services packet network: Architecture and mechanism," *SIGCOMM Comput. Commun. Rev.*, vol. 22, no. 4, p. 14–26, 1992.

[2] P. Martí, J. M. Fuertes, G. Fohler, and K. Ramamritham, "Improving quality-of-control using flexible timing constraints: metric and scheduling," in *IEEE Real-Time Syst. Symp. (RTSS)*, 2002, pp. 91–100.

[3] G. C. Buttazzo, G. Lipari, L. Abeni, and M. Caccamo, *Soft Real-Time Systems: Predictability vs Efficiency.* Springer, 2005.

[4] L. Abeni, D. Fontanelli, L. Palopoli, and B. V. Frías, "A Markovian model for the computation time of real-time applications," in *IEEE Instrum. & Meas. Tech. Conf. (I2MTC)*, 2017, pp. 1–6.

[5] B. V. Frías, L. Palopoli, L. Abeni, and D. Fontanelli, "Probabilistic real-time guarantees: There is life beyond the iid assumption," in *IEEE Real-Time and Embedded Tech. and Appl. Symp. (RTAS)*, 2017, pp. 175–186.

[6] A. Friebe, A. V. Papadopoulos, and T. Nolte, "Identification and validation of markov models with continuous emission distributions for execution times," in *IEEE Int. Conf. on Emb. and Real-Time Comp. Syst. and Appl. (RTCSA)*, 2020, pp. 1–10.

[7] A. Friebe, F. Marković, A. V. Papadopoulos, and T. Nolte, "Adaptive runtime estimate of task execution times using bayesian modeling," in *IEEE Int. Conf. Emb. and Real-Time Comp. Syst. and Appl. (RTCSA)*, 2021, pp. 1–10.

[8] R. I. Davis and L. Cucu-Grosjean, "A survey of probabilistic schedulability analysis techniques for Real-Time systems," *LITES: Leibniz Transactions on Embedded Systems*, pp. 1–53, 2019.

[9] M. F. Hamza, H. J. Yap, I. A. Choudhury, A. I. Isa, A. Y. Zimit, and T. Kumbasar, "Current development on using rotary inverted pendulum as a benchmark for testing linear and nonlinear control algorithms," *Mechanical Systems and Signal Processing*, vol. 116, pp. 347–369, 2019.

[10] N. Vreman, A. Cervin, and M. Maggio, "Stability and performance analysis of control systems subject to bursts of deadline misses," in *33rd Euromicro Conf. Real-Time Systems (ECRTS 2021)*, 2021.

[11] T. Räsänen and V.-P. Pyrhönen, "State feedback control of a rotary inverted pendulum," in *Automaatiopäivät 23: 15-16.5. 2019, Oulu.* Suomen Automaatioseura, 2019.

[12] J. Lelli, C. Scordino, L. Abeni, and D. Faggioli, "Deadline scheduling in the linux kernel," *Software: Practice and Experience*, vol. 46, no. 6, pp. 821–839, 2016.

[13] B. V. Frías, L. Palopoli, L. Abeni, and D. Fontanelli, "The prosit tool: Toward the optimal design of probabilistic soft real-time systems," *Software: Practice and Experience*, vol. 48, no. 11, pp. 1940–1967, 2018.

[14] R. I. Davis and L. Cucu-Grosjean, "A survey of probabilistic timing analysis techniques for Real-Time systems," *Leibniz Trans. Emb. Syst.*, vol. 6, no. 1, pp. 03–1–03:60, 2019.

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

[16] M. Ivers and R. Ernst, "Probabilistic network loads with dependencies and the effect on queue sojourn times," in *Int. Conf. Heterogeneous Netw. for Qual., Reliab., Sec. and Robust. (QShine)*, 2009, pp. 280–296.

[17] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzetti, E. Quiñones, and F. J. Cazorla, "Measurement-Based probabilistic timing analysis for multi-path programs," in *Euromicro Conf. on Real-Time Systems (ECRTS)*, 2012, pp. 91–101.

[18] G. Lima and I. Bate, "Valid application of evt in timing analysis by randomising execution time measurements," in *2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2017, pp. 187–198.

[19] G. Lima, D. Dias, and E. Barros, "Extreme value theory for estimating task execution time bounds: A careful look," in *2016 28th Euromicro Conference on Real-Time Systems (ECRTS).* IEEE, 2016, pp. 200–211.

[20] Y. Lu, T. Nolte, J. Kraft, and C. Norström, "Statistical-based response-time analysis of systems with execution dependencies between tasks," in *IEEE Int. Conf. Eng. of Compl. Comp. Syst. (ICECCS)*, 2010, pp. 169–179.

[21] ——, "A statistical approach to response-time analysis of complex embedded real-time systems," in *IEEE Int. Conf. Emb. and Real-Time Comp. Syst. and Appl. (RTCSA)*, 2010, pp. 153–160.

[22] Y. Lu, T. Nolte, I. Bate, and L. Cucu-Grosjean, "A statistical response-time analysis of real-time embedded systems," in *IEEE Real-Time Syst. Symp. (RTSS)*, 2012, pp. 351–362.

[23] D. Maxim, F. Soboczenski, I. Bate, and E. Tovar, "Study of the reliability of statistical timing analysis for real-time systems," in *Int. Conf. Real Time and Networks Systems (RTNS)*, 2015, pp. 55–64.

[24] M. R. Leadbetter, G. Lindgren, and H. Rootzén, "Conditions for the convergence in distribution of maxima of stationary normal processes," *Stoch. Proc. and their Appl.*, vol. 8, no. 2, pp. 131–139, 1978.

[25] L. Santinelli, J. Morio, G. Dufour, and D. Jacquemart, "On the sustainability of the extreme value theory for WCET estimation," in *Int. W. on Worst-Case Exec. Time Anal.*, 2014.

[26] F. Marković, A. V. Papadopoulos, and T. Nolte, "On the convolution efficiency for probabilistic analysis of real-time systems," in *33rd Euromicro Conference on Real-Time Systems (ECRTS 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[27] J. P. Lehoczky, "Real-time queueing theory," in *IEEE Real-Time Systems Symp.*, 1996, pp. 186–195.

[28] K. Zagalo, Y. Abdeddaim, A. Bar-Hen, and L. Cucu-Grosjean, "Response time stochastic analysis for fixed-priority stable real-time systems," *IEEE Transactions on Computers*, vol. 72, no. 1, pp. 3–14, 2022.

[29] S. Bozhko, G. von der Brüggen, and B. Brandenburg, "Monte carlo response-time analysis," in *IEEE Real-Time Syst. Symp. (RTSS)*, 2021, pp. 342–355.

[30] F. Marković, T. Nolte, and A. V. Papadopoulos, "Analytical approximations in probabilistic analysis of real-time systems," in *2022 IEEE Real-Time Systems Symposium (RTSS)*, 2022, pp. 158–171.

[31] K.-H. Chen, M. Günzel, G. von der Brüggen, and J.-J. Chen, "Critical instant for probabilistic timing guarantees: Refuted and revisited," in *2022 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2022, pp. 145–157.

[32] D. Maxim and L. Cucu-Grosjean, "Response time analysis for fixed-priority tasks with multiple probabilistic parameters," in *IEEE Real-Time Syst. Symp. (RTSS)*, 2013, pp. 224–235.

[33] G. von der Brüggen, N. Piatkowski, K.-H. Chen, J.-J. Chen, K. Morik, and B. B. Brandenburg, "Efficiently approximating the worst-case deadline failure probability under EDF," in *IEEE Real-Time Syst. Symp. (RTSS)*, 2021, pp. 214–226.

[34] A. F. Mills and J. H. Anderson, "A multiprocessor server-based scheduler for soft real-time tasks with stochastic execution demand," in *IEEE Int. Conf. Emb. and Real-Time Comp. Syst. and Appl. (RTCSA)*, 2011, pp. 207–217.

[35] R. Liu, A. F. Mills, and J. H. Anderson, "Independence thresholds: Balancing tractability and practicality in soft real-time stochastic analysis," in *IEEE Real-Time Syst. Symp. (RTSS)*, 2014, pp. 314–323.

[36] L. Abeni and G. Buttazzo, "Integrating multimedia applications in hard real-time systems," in *IEEE Real-Time Syst. Symp. (RTSS)*, 1998, pp. 4–13.

[37] ——, "QoS guarantee using probabilistic deadlines," in *Euromicro Conf. Real-Time Syst. (ECRTS)*, 1999, pp. 242–249.

[38] ——, "Stochastic analysis of a reservation based system," in *Int. Workshop on Par. and Distr. Real-Time Syst.*, vol. 1, 2001.

[39] L. Palopoli, D. Fontanelli, L. Abeni, and B. V. Frías, "An analytical solution for probabilistic guarantees of reservation based soft real-time systems," *IEEE Trans. Par. and Distr. Syst.*, vol. 27, no. 3, pp. 640–653, 2016.

[40] L. Abeni, N. Manica, and L. Palopoli, "Efficient and robust probabilistic guarantees for real-time tasks," *J. of Syst. and Soft.*, vol. 85, no. 5, pp. 1147–1156, 2012.

[41] N. Manica, L. Palopoli, and L. Abeni, "Numerically efficient probabilistic guarantees for resource reservations," in *IEEE Int. Conf. Emerg. Tech. & Factory Autom. (ETFA)*, 2012, pp. 1–8.

[42] B. V. Frías, "Bringing probabilistic real-time guarantees to the real world," Ph.D. dissertation, University of Trento, 2018.

[43] M. Shaked, *Stochastic orders*, ser. Springer series in statistics. New York: Springer, c2007.

[44] J. L. Diaz, J. M. Lopez, M. Garcia, A. M. Campos, K. Kim, and L. L. Bello, "Pessimism in the stochastic analysis of real-time systems: Concept and applications," in *IEEE Int. Real-Time Syst. Symp. (RTSS)*, 2004, pp. 197–207.

[45] L. Ljung, *System identification : theory for the user*, 2nd ed., ser. Prentice-Hall information and system sciences series. Upper Saddle River, N.J: Prentice Hall, cop. 1999.