On the Criticality of Probabilistic Worst-Case Execution Time Models

Luca Santinelli[1] and Zhishan Guo[2]

[1]ONERA France, luca.santinelli@onera.fr
[2] Missouri University of Science and Technology, guozh@mst.edu

Abstract. Probabilistic approaches to timing analysis derive probability distributions to upper bound task execution time. The main purpose of probability distributions instead of deterministic bounds, is to have more flexible and less pessimistic worst-case models. However, in order to guarantee safe probabilistic worst-case models, every possible execution condition needs to be taken into account.
In this work, we propose probabilistic representations which is able to model every task and system execution conditions, included the worst-cases. Combining probabilities and multiple conditions offers a flexible and accurate representation that can be applied with mixed-critical task models and fault effect characterizations on task executions. A case study with single- and multi-core real-time systems is provided to illustrate the completeness and versatility of the representation framework we provide.

1 Introduction

Nowadays real-time systems are mostly implemented with multi-core and many-core commercial-off-the-shelf platforms. Cache memories, branch predictors, communication buses/networks and other features present in such implementations allow increasing average performance; nonetheless, they make predictability much harder to guarantee.

Task execution times are affected by the large variety of system execution conditions which can happen at runtime. Also, the numerous interferences within real-time systems may result into significant variations of tasks execution times. This acerbates with multi- and many-core real-time systems implementations. In essence, task execution time resembles to a random variable where the value depends on different outcomes.

Timing analysis seeks upper bounds to the task execution time, and the predictability required by real-time systems can be granted. Classically, the bounds are deterministic Worst-Case Execution Times (WCET) which are single values able to upper bound the time needed to finish execution. In order to be safe, WCETs have to account for any case/execution condition possible, including the highly improbable pathological cases such as faults. Deterministic WCETs could be very pessimistic with respect to task actual execution times, and could lead to resource under-utilization.

Probabilistic worst-case models generalize WCETs with worst-case distributions, probabilistic WCETs (pWCETs), which upper bounds any possible task

execution behavior. pWCET representations focus on probability of occurrence of worst-case conditions and abstract them into multiple worst-case values with their correspondent probability of happening. The challenge with probabilistic timing analysis is guaranteeing the pWCET safety.

System faults have a non negligible impact on worst-case models; although as pathological and improbable cases, they have to be considered with timing analysis. Task models have to embed fault manifestations and fault effects in order to provide upper bounds to every possible condition the real-time system can experience at runtime.

With multi- and many-core implementations, it comes the opportunity to combine different applications on the same platform which may have different criticality levels, e.g. safety-critical, mission-critical, non-critical, designating the level of assurance needed against failure. The very low acceptable failure rates (e.g. 10^{-9} failures per hour) for high criticality applications imply the need for significantly more rigorous and costly development as well as verification processes than required by low criticality applications. The gradual transformation of real-time systems into Mixed Criticality (MC) systems demands for timing analysis is able to cope with multiple criticality levels for tasks.

State of the Art: First papers on probabilistic timing modeling describe the worst-case execution time of tasks with random variables, using either discrete [22,2] or continuous [15] distributions. Since Edgar and Burns [10], several papers have worked on obtaining safe and reliable probabilistic Worst-Case Execution Time (pWCET) estimates [13,7,14].

Among probabilistic timing analysis approaches, it is possible to distinguish between Static Probabilistic Timing Analysis (SPTA) and Measurement-Based Probabilistic Timing Analysis (MBPTA). SPTA methods analyze the software and use a model of the hardware behavior to derive an estimate of pWCET; SPTA is applicable when some part of the system or the environment have been artificially time randomized, [8,3]. MBPTA approaches rely on the Extreme Value Theory (EVT) for computing pWCET estimates out of measured behaviors [11,6,16]. Figure 1 depicts key elements for MBPTA which accepts input measurements of task execution times under specific execution conditions and applies the EVT for inferring pWCET estimates.

Fault modeling and fault management intertwines with timing analysis as faults introduce latencies to the task execution behavior which have to be embedded into the task models. As examples, in [19] backups are executed for fault tolerance and recovering form task errors caused by hardware or software faults; in [25] it is proposed an algorithm to abort and restart a task in case of conflicts in shared resources accesses. In [21], an attempt of including faults into MBPTA and apply fault-based task models for schedulability and sensitivity analysis of real-time systems.

Research on MC scheduling focuses upon the Vestal task model which assigns multiple WCET estimates to each task, [23]. This is motivated by the fact that different tools for WCET estimates may be more or less conservative than one another, [4,24] as reviews. To the best of our knowledge, [12] is a first attempt

of a probabilistic MC task modeling with WCET estimates associated to the probability/confidence of being WCET bound.
Contribution: This work proposes a probabilistic representation framework for real-time tasks which composes of multiple probabilistic worst-case models, each estimating the worst-case of a specific possible execution condition that both tasks and the system can encounter.

The probabilistic task model developed can be applied to the MC problem, since probabilities and multiple pWCETs can characterize the criticality modes as different task conditions as well as different confidences. Besides, the probabilistic representation can be used for characterize the effects that faults have on the tasks executions, proving to be flexible and safe. A case study is presented for validating the probabilistic models and illustrating its effectiveness in modeling different conditions/criticalities.
Organization of the paper: Section 2 presents the probabilistic background applied in this work. Section 3 describes the probabilistic worst-case model proposed and based on multiple execution conditions possible. Section 4 details the MC task representation derived from the the probabilistic worst-case model. The impact of faults on task models are also considered. Section 5 shows probabilistic MC task models from three different real-time test cases; Section 6 is for conclusions and future works.

2 System Modeling

A real-time task consists of a sequence of recurring jobs, each has to complete execution by a given deadline.

In a periodic task system, a task is described with the 4-tuple (O_i, T_i, D_i, C_i), where O_i is the offset or starting time that specifies the time instant at which the first job of τ_i is released. T_i is the period as the temporal separation between two successive jobs of τ_i. D_i is the deadline which defines the time interval $[O_i + j \cdot T_i, O_i + j \cdot T_i + D_i)$ in which task execution has to take place (the j-th job of τ_i). C_i is the WCET defining the processing requirements for each job.

The scheduling policy decides the task execution ordering, possibly with preemption or migration between cores; schedulability analysis of task models guarantees the respect of the timing constraints (system predictability) checking if there are enough resources for the tasks to finish executions by their deadlines.

In this work, we consider a set $\Gamma = \{\tau_1, \ldots, \tau_n\}$ of n periodic *probabilistic tasks* τ_i as such their worst-case execution time is modeled with pWCETs. The probabilistic modeling framework proposed applies to either single-core, multi-core or many-core processors.

2.1 pWCETs and WCET Thresholds

The pWCET $\mathcal{C}_i$ of a task τ_i is defined as the worst-case estimate distribution that upper-bounds any possible execution time the task can exhibit [8]. $\mathcal{C}_i$ generalize

deterministic WCET estimates C_i by including multiple values, each with the probability of being the worst-case of the task execution time[1].

Assuming the pWCET $\mathcal{C}_i$ as continuous distribution, the probability distribution function (pdf) $\mathsf{pdf}_{\mathcal{C}_i}$ describes the probability of happening of certain events from the random variable $\mathcal{C}_i$; it is such that $P(C_1 \leq \mathcal{C}_i \leq C_2) = \int_{C_1}^{C_2} \mathsf{pdf}_{\mathcal{C}_i}(C)dC$ and $\int_0^\infty \mathsf{pdf}_{\mathcal{C}_i}(C)dC = 1$.

$\mathsf{cdf}_{\mathcal{C}_i}$ denotes the cumulative distribution function (cdf) representation of $\mathcal{C}_i$, $\mathsf{cdf}_{\mathcal{C}_i}(C) = P(\mathcal{C}_i \leq C) = \int_0^C \mathsf{pdf}_{\mathcal{C}_i}(x)$ while the inverse cumulative distribution function (icdf) $\mathsf{icdf}_{\mathcal{C}_i}(C)$ outlines the exceedence thresholds. $\mathsf{icdf}_{\mathcal{C}_i}(C) = P(\mathcal{C}_i \geq C)$ is the probability of having execution time greater than C, $\mathsf{icdf}_{\mathcal{C}_i}(C) = 1 - \int_0^C \mathsf{pdf}_{\mathcal{C}_i}(x)$.

In case of discrete distributions pWCETs, it is $\mathsf{pdf}_{\mathcal{C}_i}(C) = P(\mathcal{C}_i = C)$, $\mathsf{cdf}_{\mathcal{C}_i}(C) = \sum_0^C \mathsf{pdf}_{\mathcal{C}_i}(x)$ and $\mathsf{icdf}_{\mathcal{C}_i}(C) = 1 - \sum_0^C \mathsf{pdf}_{\mathcal{C}_i}(x)$.

WCET Thresholds From $\mathcal{C}_i$, it is possible to define *WCET thresholds* $\langle C_{i,j}, p_{i,j} \rangle$ where the value $C_{i,j}$ is associated to the probability $p_{i,j}$ of being the WCET for τ_i. $p_{i,j} \stackrel{def}{=} \mathsf{icdf}_{\mathcal{C}_i}(C_{i,j})$ quantifies the *confidence* on $C_{i,j}$ of being the task worst-case execution time and $1-p_{i,j}$ is the probability of respecting $C_{i,j}$. Depending on the granularity of the pWCET, it would be possible to define WCET thresholds at probability of 10^{-3}, 10^{-6}, 10^{-9}, etc..

Worst-Case Distribution Independence Assuming $\mathcal{C}_i$ to be the probabilistic worst-case distribution estimate of τ_i, it means that in $\mathcal{C}_i$ there have already been included all the possible interferences (and their effects as latencies) that τ_i suffers, [5]. This is the definition of statistical independences between tasks, i.e. the task execution distribution does not change in presence of interferences[2].

For example, the pWCET distribution of task τ_i in presence of τ_j, equivalently while executing together with τ_j (the conditional distribution $\mathsf{pdf}_{\mathcal{C}_i|\mathcal{C}_j}$) does not change from the case where τ_i runs in isolation ($\mathsf{pdf}_{\mathcal{C}_i}$) since all the interferences from τ_j have been taken into account in the worst-case distribution bound and in order to guarantee it. It is $\mathsf{pdf}_{\mathcal{C}_i|\mathcal{C}_j} = \mathsf{pdf}_{\mathcal{C}_i}$ which corresponds to the definition of statistical independence between pWCETs and thus tasks. Previous independence condition holds with all the tasks in Γ and for all the system execution conditions possible, i.e. worst-case distributions guarantee independence between tasks.

[1] In the following, calligraphic letters are used to represent distributions while non-calligraphic letters are for scalars or deterministic values.

[2] The conditional probability is the probability of one event A happening concurrently with another event B, $P(A|B)$. In case of statistical independence it is $P(A|B) = P(A)$.

3 Probabilistic Worst-Case Representations

Whenever correctly applied, the EVT produces a continuous distribution which is a safe estimation of the worst-case behavior of the task. The EVT guarantees that if certain hypotheses are verified, from the actual measured behavior it is possible to infer rare events, where the worst-case execution time lie [6]. The outcome of the MBPTA/EVT is the pWCET estimate $\mathcal{C}_i$.

Figure 1 shows the basics of MBPTA with the EVT applied to measurements of task execution time – average execution time – for inferring pWCET estimation $\mathcal{C}_i$. The task under observation is τ_i while the rest of the real-time application $\Gamma \setminus \tau_i$ contributes to the interferences on τ_i. Besides, the specific measurement execution condition applied for the task executions $s^k = f(I, Env, Map, \ldots)$ and the measurements themselves define the task's actual behavior.

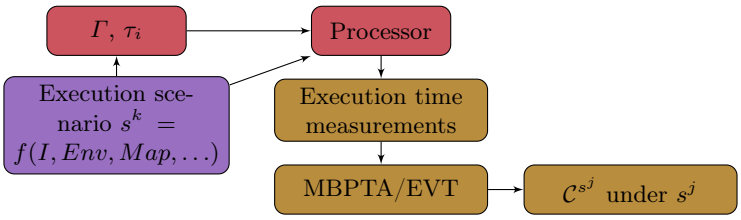


Fig. 1. MBPTA where pWCETs are inferred with the EVT specific execution conditions $s^k = f(I, Env, Map, \ldots)$ applied for measurements.

The guarantees that the EVT provides worst-case task models strongly depend on what has been measured, e.g. the execution conditions for the measurements, the confidence or representativity of the measurements.

A trace of execution time measurements accounts for some of the interfering conditions and inputs (to the system and tasks) which happen at runtime. The pWCET estimate $\mathcal{C}_i$ from the EVT embeds those system conditions and others which have not been measured (the so called rare events which are costly to observe by measurements), i.e. the EVT is able to infer some of the unknowns from the known measurements. Unfortunately, not all the unknowns can be estimated with the only use of the EVT.

An execution scenario $s^j = f(I, Env, Map, \ldots)$ abstracts the execution conditions the system (and the task) subdue to. s^j represents instances of the inputs (for tasks and system) I, of the environment Env, of the task mapping and scheduling policy Map, etc.; s^k is a function of I, Env, Map and more. For a real-time system, there exist a finite set S of all the possible execution scenarios, $S = \{s^1, s^2, \ldots, s^n\}$, since inputs, environment conditions, mapping, etc. are finite.

The same reasoning would be applicable to SPTA with different conditions $s^j = f(I, Env, Map, \ldots)$ possible for tasks.

3.1 Worst-Case Bounding

The *absolute* pWCET $\mathcal{C}_i$ is the worst-case distribution that upper bounds every task execution time obtained under any possible execution scenario $s^j \in S$. The absolute pWCET $\mathcal{C}_i$ is safe if it upper bounds every task execution time under any execution scenario.

Given $s^j \in S$, the pWCET $\mathcal{C}_i^{s^j}$ comes from the measurements taken under s^j and the EVT applied to them (Figure 1). $\mathcal{C}_i^{s^j}$ is the pWCET specific to s^j, the *relative* pWCET; the relative pWCET $\mathcal{C}_i^{s^j}$ is safe if it upper bounds any task execution time under s^j.

Measurement representativity is a fundamental requirement for guaranteeing both absolute and relative pWCETs. We hereby focus on representativity as the measurement capability of well characterizing multiple execution conditions (worst-cases included) like in [17,1]. Differently than those works, we do not consider artificially randomized systems that aim at increasing the chances of measuring the worst-case. We believe that the input representativity can be built from an enhanced knowledge of the system and of its scenarios, thus *from a study of the system, its S and the coverage of the execution conditions.*

From the partial ordering between pWCETs [9], it is possible defining a notion of *dominance* for scenarios. With respect to task τ_i, given s^r and s^t from S, s^r dominates s^t if and only if $\mathcal{C}_i^{s^r}$ is greater than or equal to $\mathcal{C}_i^{s^t}$, $\mathcal{C}_i^{s^r} \succeq \mathcal{C}_i^{s^t}$, $\succeq$ being the "greater than or equal to" operator which defines the partial ordering between distributions [9].

It is also possible defining the notion of *equivalence* between scenarios. Given s^k and s^j from S, with respect to task τ_i s^k is equivalent to s^j if and only if there exist values in the support of $\mathcal{C}_i^{s^k}$ and $\mathcal{C}_i^{s^j}$ for which $\mathcal{C}_i^{s^k} \succeq \mathcal{C}_i^{s^j}$ and there exist other values in the support of $\mathcal{C}_i^{s^k}$ and $\mathcal{C}_i^{s'^j}$ for which $\mathcal{C}_i^{s^j} \succeq \mathcal{C}_i^{s^k}$.

For a set of equivalent scenarios $S^j = \{s^j, s^k, \ldots, s^t\} \subseteq S$ ($s^k, \ldots, s^t$ equivalent to s^j), it is possible defining the scenario s^{j*} that dominates all the scenarios in S^j. It would be such that $\mathcal{C}_i^{s^{j*}} \stackrel{def}{=} max_{s^j \in S^j}\{\mathcal{C}_i^{s^j}\}$, while with the icdf representation, it would be $\mathsf{icdf}_{\mathcal{C}_i^{s^{j*}}}(C) \stackrel{def}{=} max_{s^j \in S^j}\{\mathsf{icdf}_{\mathcal{C}_i^{s^j}}(C)\}$. s^{j*} is not a real scenario, but it dominates all the $s^k \in S^j$.

Worst-Case Set. The *Worst-Case Set* task representation is the collection of all the pWCET from S; $\overline{\mathcal{C}}_i$ is the Worst-Case Set representation as a set of pWCET estimates such that:

$$\overline{\mathcal{C}}_i \stackrel{def}{=} (\mathcal{C}_i^{s^1}, \mathcal{C}_i^{s^2}, \ldots, \mathcal{C}_i^{s^n}). \tag{1}$$

With partial ordering between relative pWCETs it is possible ordering scenarios and get $S = \{s^1, s^2, s^3, \ldots, s^k\}$ such that $\mathcal{C}^{s^k} \succeq \mathcal{C}^{s^{k-1}} \succeq \ldots \succeq \mathcal{C}^{s^1}$; the Worst-Case Set becomes:

$$\overline{\mathcal{C}}_i \stackrel{def}{=} (\mathcal{C}_i^{s^1}, \mathcal{C}_i^{s^2}, \ldots, \mathcal{C}_i^{s^k}), \tag{2}$$

with s^k the worst-case scenario for τ_i, $s^{worst} \equiv s^k$.

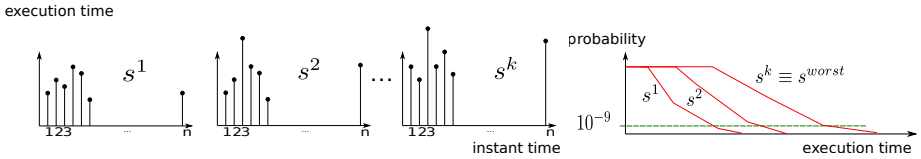
Although, we hereby focus on MBPTA, the Worst-Case Set representation applies to both MBPTA and SPTA with multiple execution conditions possible.

Worst-Case Set & Dominance Although with actual real-time systems it is reasonable to assume a finite number of measurement scenarios, enumerating them all remains a complex problem. With dominance between scenarios, it would be possible neglecting the dominated scenarios in order to ease the task representation from Equation (1) and Equation (2). Moreover, with equivalence between scenarios it would be possible to assume the correspondent dominating scenario $s^{*,j}$ to represent all the equivalent scenarios S^j.

From Equation (1) and Equation (2), fewer dominating scenarios S^* could be considered to represent the task execution behavior. $S^* = \{s^r, s^j, s^k\} \subseteq S$ is such that s^r dominates some scenarios in S, s^j dominates other scenarios as well as s^r and s^k dominates all the scenarios. The Worst-Case Set becomes:

$$\overline{\mathcal{C}}_i \stackrel{def}{=} (\mathcal{C}_i^{s^r}, \mathcal{C}_i^{s^j}, \mathcal{C}_i^{s^k}), \tag{3}$$

as a less complex probabilistic representation to the task executions; Equation (3) remains a safe representation for the task behavior since the worst-cases s^k and $\mathcal{C}_i^{s^k}$ are included.



(a) Traces of execution time measurements (b) icdfpWCET estimates

Fig. 2. Multiple scenarios $S = \{s^1, s^2, \ldots, s^k \equiv s^{worst}\}$, each with a trace of execution time measurements and pWCET estimate.

Figure 2 depicts $S = \{s^1, s^2, \ldots, s^k \equiv s^{worst}\}$, each scenario s^j with a trace of execution time measurements; the resulting pWCETs $\mathcal{C}_i^{s^k}$ are illustrated with the partial ordering guaranteed by $\succeq$.

At this stage, S is assumed to be known; future work will investigate how to obtain the different scenarios and how to guarantee the existence of worst-cases among them.

3.2 Probabilistic Task Models

Combining the orthogonal information of WCET thresholds (with probability associated) and execution scenarios from the Worst-Case Set, Equation (1), Equation (2) or Equation (3), there are possible the *inter-scenario* and *intra-scenario* representations.

Inter-Scenario Representation. The inter-scenario representation characterizes task behavior across scenarios. Given an exceeding probability p and the WCET threshold at that exceeding probability for each scenario $s^j \in S$ (equivalently $s^j \in S^*$), it is $\langle \overline{C}_i, p \rangle$ such that:

$$\overline{C}_i \stackrel{def}{=} (C_i^{s^1}, C_i^{s^2}, \ldots, C_i^{s^k}) \tag{4}$$

is the set of WCET thresholds such that $\langle C_i^j, p \rangle \ \forall s^j \in S$. As an example, it is possible picking $p = 10^{-9}$ with $\langle C_i^j, 10^{-9} \rangle \ \forall s^j \in S$. Equation (4) is the inter-scenario representation for the task worst-case execution time.

Intra-Scenario Representation. The intra-scenario representation describes the task behavior focussing on a specific scenario. For a given scenario $s^j \in S$ (equivalently $s^j \in S^*$) and a set of exceeding thresholds probabilities $(p_1, p_2, \ldots p_m)$ it is:

$$\hat{C}_i^{s^j} \stackrel{def}{=} (\langle C_{1,i}^{s^j}, p_1 \rangle, \langle C_{2,i}^{s^j}, p_2 \rangle, \ldots, \langle C_{m,i}^{s^j}, p_m \rangle). \tag{5}$$

Equation (5) is the intra-scenario representation for the task worst-case execution time on a specific scenario with all the meaningful WCET thresholds and exceeding probabilities.

Inter- and intra-scenario representations will be proven to be handy for schedulability and sensitivity analysis with future work.

4 Task Modeling through Criticalities

Each pWCET estimations composing the Worst-Case Set representation implicitly carries confidence (as safety) of being the absolute task pWCET; execution scenarios may be more or less safe in defining pWCET estimates and WCET thresholds. For example, s^1 from Equation (2) provides the least confident absolute pWCET as $\mathcal{C}_i^{s^1}$: $\mathcal{C}_i^{s^1}$ is the least safe absolute pWCET for τ_i; s^2 provides slightly more confidence that $\mathcal{C}_i^{s^2}$ is the absolute pWCET for τ_i: $\mathcal{C}_i^{s^2}$ is relatively safer than $\mathcal{C}_i^{s^1}$. Going on with the scenarios within S, the safety increases up until s^k which is the worst-case scenario and $\mathcal{C}_i^{s^k}$ is the only 100% safe absolute pWCET for τ_i; $\mathcal{C}_i^{s^k}$ is the safest among the pWCETs.

The MC task model makes use of multiple WCETs for characterizing the task behavior; such bounds results from different timing analysis tools as well as different criticality requirements that task can respect at runtime. For example, in the two-criticality-level case, each task is designated as being of either higher (HI) or lower (LO) criticality, and two WCETs are specified for each HI-criticality task: a LO-WCET determined by a less pessimistic tool or a less demanding safety requirements (e.g. mission-critical or non-critical) , and a larger HI-WCET determined by a more conservative tool or more safety-critical requirements.

For real-time systems, safety and criticality have a strong relationship so that they can be interchanged whenever applied for timing analysis and schedulability

analysis: a safe pWCET is the worst-case models which can apply with high critical modes.

Least critical Scenario LO-critical. In case of s^1 from Equation (2), the task has the least execution time, thus the least dominating relative pWCET $\mathcal{C}_i^{s^1}$. $\mathcal{C}_i^{s^1}$ upper bounds any (and only) possible execution time resulting from s^1; it is the last safe absolute pWCET, equivalently the least critical LO-criticality. $\mathcal{C}_i^{s^1}$ is applied to characterize LO-criticality requirements of τ_1 and the LO-criticality functional mode.

Critical Scenarios MI-critical. From S ordered by dominance, Equation (2), s^2 dominates s^1 because under s^2 the task suffers execution times bigger than under s^1. Considering $\mathcal{C}_i^{s^2}$ as absolute pWCET, it would be slightly safer than $\mathcal{C}_i^{s^1}$, but it is not safe enough to upper bound the other $s^j \in S$. $\mathcal{C}_i^{s^2}$ is the middle criticality (MI-criticality) characterization for τ_i.
With s^3, it is $\mathcal{C}_i^{s^3}$ dominating $\mathcal{C}_i^{s^2}$ and $\mathcal{C}_i^{s^1}$ since s^3 produces larger execution times than s^1 and s^2. Thus, $\mathcal{C}_i^{s^3}$ would be safer than $\mathcal{C}_i^{s^2}$ as absolute pWCET. Also $\mathcal{C}_i^{s^3}$ is a MI-criticality characterization for τ_i but more critical than $\mathcal{C}_i^{s^2}$.
We distinguish between MI-2-criticality and MI-3-criticality, respectively for s^2 and s^3, and $\mathcal{C}_i^{s^3} \succeq \mathcal{C}_i^{s^2}$. Other intermediate criticality levels can be defined from $s^j \in (S \setminus s^k)$.

Most Critical Scenario, HI-critical. The pWCET $\mathcal{C}_i^{s^k}$ from s^k is the safest absolute pWCET. $\mathcal{C}_i^{s^k}$ is also the HI-criticality bound to the task behavior. $\mathcal{C}_i^{s^k} \equiv \mathcal{C}_i^{s^{worst}}$ represents the worst conditions and is the most conservative upper bound for τ_i to be applied in the highest critical modes.
From $\mathcal{C}_i^{s^1}$ it would be possible to extract $\langle C_i(\text{LO}), 10^{-9} \rangle$. We name $C_i(\text{LO})$ the LO-critical WCET threshold as it results from the least safe pWCET model $\mathcal{C}_i^{s^1} \equiv \mathcal{C}_i^{\text{LO}}$. $C_i(\text{LO})$ is the LO-criticality WCET threshold with a confidence of 10^{-9}.
s^j, with $1 < j < k$ form Equation (2), is a MI-j-criticality scenario that upper bounds all the scenarios s^r such that $r \leq j$; $\mathcal{C}_i^{s^j} \equiv \mathcal{C}_i^{\text{MI}-j}$ is the MI-j-criticality pWCET. $\mathcal{C}_i^{s^j} \succeq \mathcal{C}_i^{s^{j-1}}$ and $\langle C_i(\text{MI}-j), 10^{-9} \rangle$ is such that $C_i(\text{MI}-j) \geq C_i(\text{MI}-j-1)$; $C_i(\text{MI}-j)$ is the MI-j-criticality WCET threshold with a confidence of 10^{-9}.
s^k is the worst-case scenario and $\mathcal{C}_i^{s^k}$ is the absolute pWCET for τ_i. $\mathcal{C}_i^{s^k} \equiv \mathcal{C}_i^{\text{HI}}$ is the HI-criticality pWCET and s^k is the HI-criticality scenario for the worst conditions. From $\mathcal{C}_i^{\text{HI}}$, it is $\langle C_i(\text{HI}), 10^{-9} \rangle$ such that $C_i(\text{HI})$ is the HI-criticality WCET threshold. $\mathcal{C}_i^{\text{HI}} \succeq \mathcal{C}_i^{\text{MI}-j}$ and $C_i(\text{HI}) \geq C_i(\text{MI}-j)$.

From the difference in safety/criticality between s^{worst}, s^3, s^2 and s^1 execution conditions, it is $\mathcal{C}_i^{\text{HI}} \succeq \mathcal{C}_i^{\text{MI}} \succeq \mathcal{C}_i^{\text{LO}}$. Also, $C_i(\text{HI}) \geq C_i(\text{MI}) \geq C_i(\text{LO})$ for the same probability p from respectively $\mathcal{C}_i^{\text{HI}}$, $\mathcal{C}_i^{\text{MI}}$ and $\mathcal{C}_i^{\text{LO}}$. How much they differ depends on the relationship between the scenarios and the impact that the scenarios have on the execution times of tasks. $p = 10^{-9}$ is chosen arbitrarily, but the probabilistic modeling proposed can make use of any probability, depending on the confidence requirements.

The MC Worst-Case Set representation for τ_i is:

$$\overline{\mathcal{C}}_i \stackrel{def}{=} (\mathcal{C}_i^{\text{LO}}, \ldots, \mathcal{C}_i^{\text{MI}-j}, \ldots, \mathcal{C}_i^{s^k}). \tag{6}$$

For the intra- and inter-scenario perspective, adding criticality levels to Equation (4) and Equation (5) it is $\overline{C}_i \stackrel{def}{=} (C(\text{LO})_i, \ldots, C(\text{HI})_i)$ for the inter-scenario MC representation $\langle \overline{C}_i, p \rangle$ at probability p and $\hat{C}(\updownarrow)_i \stackrel{def}{=} (\langle C(\updownarrow)_{1,i}, p_1 \rangle, \langle C(\updownarrow)_{2,i}, p_2 \rangle, \ldots, \langle C(\updownarrow)_{m,i}, p_m \rangle)$ for the intra-scenario MC representation at the criticality level l and probabilities $p_1, p_2, \ldots p_m$.

MC Probabilistic Task Model. With three criticality levels, the MC task model based on the Worst-Case Set is:

$$\tau_i = ([\overline{\mathcal{C}}_i, \langle \overline{C}_i, 10^{-9} \rangle, (\hat{C}_i^{\text{LO}}, \hat{C}_i^{\text{MI}}, \hat{C}_i^{\text{HI}})], T_i, D_i), \tag{7}$$

where $\overline{\mathcal{C}}_i = (\mathcal{C}_i^{\text{LO}}, \mathcal{C}_i^{\text{MI}}, \mathcal{C}_i^{\text{HI}})$ and $\langle \overline{C}_i, p \rangle = (C_i^{\text{LO}}, C_i^{\text{MI}}, C_i^{\text{HI}})$. The intra-scenario representation is such that $\hat{C}_i^{\text{LO}} = (\langle C_{1,i}^{\text{LO}}, 10^{-3} \rangle, \langle C_{2,i}^{\text{LO}}, 10^{-6} \rangle, \langle C_{3,i}^{\text{LO}}, 10^{-9} \rangle)$ for the LO-safety, $\hat{C}_i^{\text{MI}} = (\langle C_{1,i}^{\text{MI}}, 10^{-3} \rangle, \langle C_{2,i}^{\text{MI}}, 10^{-6} \rangle, \langle C_{3,i}^{\text{MI}}, 10^{-9} \rangle)$ for the MI-safety scenario and $\hat{C}_i^{\text{HI}} = (\langle C_{1,i}^{\text{HI}}, 10^{-3} \rangle, \langle C_{2,i}^{\text{HI}}, 10^{-6} \rangle, \langle C_{3,i}^{\text{HI}}, 10^{-9} \rangle)$ for the HI-safety scenario.

The MC task model is essentially asserting that depending on the conditions for the timing analysis applied it is possible to have more or less guarantees on the pWCET and the WCET thresholds estimates. Only by considering most of the possibilities (necessarily the dominating ones) the MC worst-case models are safe. The MC task model can be generalized to multiple criticality levels and different probabilities in order to better cope with the requirements.

Worst-Case Sets Independence It is necessary to investigate the statistical independence between criticality levels and pWCETs for the MC Worst-Case Set representation. It has already been showed that there exist independence between absolute pWCETs, thus between HI-criticality representations $\mathcal{C}_i^{\text{HI}}$ and $C_i(\text{HI})$ from s^k.

Supposing s^j represents a criticality level other than HI-criticality, what happens to the pWCET estimates of τ_i and τ_k? Under s^j, the conditional probability $\mathsf{pdf}_{\mathcal{C}_i^{s^j} | \mathcal{C}_k^{s^j}}$ equals $\mathsf{pdf}_{\mathcal{C}_i^{s^j}}$ (equivalently $\mathsf{pdf}_{\mathcal{C}_k^{s^j} | \mathcal{C}_i^{s^j}}$ equals $\mathsf{pdf}_{\mathcal{C}_k^{s^j}}$) because all the effects from s^j have been included into the relative pWCETs $\mathcal{C}_i^{s^j}$ and $\mathcal{C}_k^{s^j}$. This assures tasks independence with the same scenario.

With s^r dominating s^j for both τ_i and τ_k, what happens to $\mathcal{C}_i^{s^r}$ and $\mathcal{C}_k^{s^j}$? It is $\mathsf{pdf}_{\mathcal{C}_i^{s^r} | \mathcal{C}_k^{s^j}} = \mathsf{pdf}_{\mathcal{C}_i^{s^r}}$, since all the effects of s^j and τ_k on τ_i have been already taken into account by $\mathcal{C}_i^{s^r}$. This guarantees the independence between τ_i and τ_k under s^r and s^j, respectively for τ_i and τ_k.

Worst-Case Set representations guarantee tasks independence and independence between criticality levels which will ease task combination and schedulability analysis.

4.1 Fault Occurrence and Effect

Faults can be modeled with the probability $f(t,T)$ of a fault occurring; $f(t,T)$ is the probability of fault in a system component by time t, $failure \leq t$, given that the component was still functional at the end of the previous interval $t-T$, $failure > t-T$. T is the scrubbing period, i.e. the time interval between two consecutive fault detection to avoid error accumulation.

Several probability distributions are used to model failure times [18]. One commonly used is the log-normal failure distribution:

$$f(t,T) = \frac{\mathsf{cdf}_{norm}\frac{ln(t)-\mu}{\sigma} - \mathsf{cdf}_{norm}(\frac{ln(t-T)-\mu}{\sigma})}{1-\mathsf{cdf}_{norm}(\frac{ln(t-T)-\mu}{\sigma})}, \tag{8}$$

where cdf_{norm} the cumulative density function of the normal distribution. The mean and standard deviation parameters of such distribution can be computed from the Mean Time To Failure (MTTF) such that $\mu = ln(MTTF^2/\sqrt{var_{MTTF}+MTTF^2})$ and $\sigma = \sqrt{ln(1+var_{MTTF}/MTTF^2)}$. In Equation (8), $f(t,T)$ depends on the actual time t and the scrubbing period T.

Fault & Criticalities Faults (either transient or permanent) translate into penalties δ (latencies) to the task execution time which depends on the time t the fault happens, $\delta(t)$. With $C(t)$ the expected task execution time at time t, in presence of fault it would be $C(t)+\delta(t)$ the task execution time accounting for the fault penalty on task computations. With a measurement-based approach, fault effects on task execution can be measured and directly embedded into traces of execution time measurements; then, with EVT it is possible infer pWCET estimates which upper bounds faulty execution conditions.

Different scenarios are possible with respect to faults. By considering non-faulty conditions (fault never happening), it is scenario s^{NF} that describes the task behavior. Here, the execution times observed exploit only the task functional behavior due to the absence of faults. For s^{NF} it exists $\mathcal{C}_i^{s^{NF}}$; s^{NF} is the LO-criticality scenario with $\mathcal{C}_i^{\text{LO}} \equiv \mathcal{C}_i^{NF}$, $C(\text{LO})$ and $\hat{C}_i(\text{LO})$ representing it.

It could also exist s^{FW} which assumes that the worst fault condition manifests at runtime; $\mathcal{C}_i^{FW}$ is the pWCET estimate for s^{FW}. s^{FW} is the wort-case scenario where the task executes always under the most critical conditions; $\mathcal{C}_i^{\text{HI}} \equiv \mathcal{C}_i^{FW}$, $C_i(\text{HI})$ and $\hat{C}_i(\text{HI})$ represents it.

In between these two extreme scenarios, it exist a set of possible faulty scenarios where faults are not as extreme as s^{FW} and s^{NF}, nonetheless they happen and affect the normal task behavior. For example, it could exist s^{F1} which is the MI-1-criticality scenario with $\mathcal{C}_i^{\text{MI}-1} \equiv \mathcal{C}_i^{s^{F1}}$, $C_i(\text{MI}-1)$ and $\hat{C}_i(\text{MI}-1)$; $C_i(\text{MI}-1) \geq C_i(\text{LO})$ and $\mathcal{C}_i^{\text{MI}-1} \succeq \mathcal{C}_i^{\text{LO}}$. It could also exist s^{F2} as the MI-2-criticality scenario with $\mathcal{C}_i^{\text{MI}-2} \equiv \mathcal{C}_i^{s^{F2}}$, $C(\text{MI}-2)$ and $\hat{C}_i(\text{MI}-2)$; $C_i(\text{MI}-2) \geq C_i(\text{MI}-1)$ and $\mathcal{C}_i^{\text{MI}-2} \succeq \mathcal{C}_i^{\text{MI}-1}$.

Specific to faults and faulty scenarios, it is $S = \{s^{NF} \equiv s^{\text{LO}}, s^{F1} \equiv s^{\text{MI}-1}, s^{F2} \equiv s^{\text{MI}-2}, \ldots, s^{FW} \equiv s^{\text{MI}}\}$ with the task MC Worst-Case Set given by $\overline{\mathcal{C}}_i = (\mathcal{C}_i^{s^{\text{LO}}}, \mathcal{C}_i^{s^{\text{MI}-1}}, \mathcal{C}_i^{s^{\text{MI}-2}}, \ldots, \mathcal{C}_i^{s^{\text{HI}}})$.

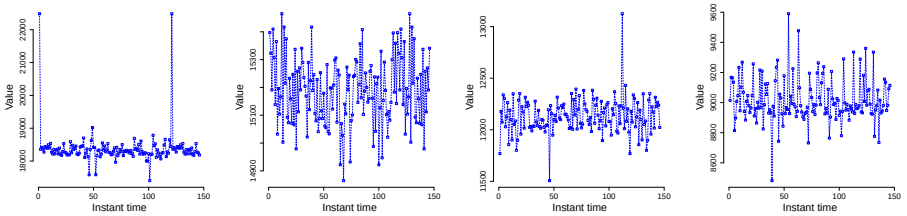
What we are hereby proposing is a representation framework that applies to faults effects. It could abstract different faults and fault tolerant mechanisms implemented as recovery functions or task extra-executions resulting into larger task execution times and worst-case execution times.

5 Case Study

Three case studies are presented for illustrating the flexibility and the effectiveness of the Worst-Case Set representation in modeling execution conditions, MC tasks and fault effects.

5.1 Test Case 1

As a first test case, it is the *ns* task from the Mälardalen WCET benchmark executed a multi-core platform, [20]; in this configuration, *ns* executes alone on one core while other tasks and the OS execute on different cores making interference through shared resources. The *ns* considered here has four scenarios $S = \{s^1, s^2, s^3, s^4\}$ depending only on the task inputs; $s^4 \equiv s^{worst}$ dominates all the other scenarios, s^3 dominates s^2 and s^1, and so on. The four traces of execution time measurements are *trace_1*, *trace_2*, *trace_3* and *trace_4*, respectively for s^1, s^2, s^3 and s^4 and they embed all the known task behaviors, worst-case included. From $S = \{s^1, s^2, s^3, s^4\}$, it is possible to define four criti-



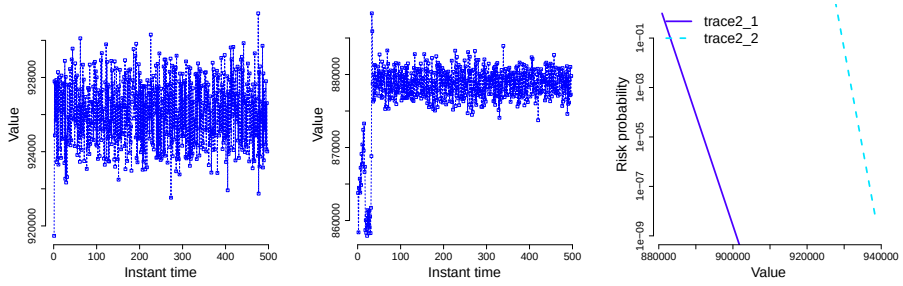
(a) HI-criticality trace (b) MI-3-criticality trace (c) MI-2-criticality trace (d) LO-criticality trace

Fig. 3. *ns* multi-scenario benchmark with four execution time measurements traces; the execution time in ordinate are in CPU cycles.

cality levels ($\text{LO}, \text{MI}-2, \text{MI}-3, \text{HI}$) for the scenarios and their respective pWCETs, $S = \{s^1 \equiv s^{\text{LO}}, s^2 \equiv s^{\text{MI}-2}, s^3 \equiv s^{\text{MI}-3}, s^4 \equiv s^{\text{HI}}\}$. Figure 3 presents the four traces of measurements under s^{LO}, $s^{\text{MI}-1}$, $s^{\text{MI}-2}$ and s^{HI}, respectively *trace_1*, *trace_2*, *trace_3* and *trace_4*. The criticality levels depend on the execution considered and the dominance between them. Figure 6 compares the pWCETs $(\mathcal{C}^{\text{LO}}, \mathcal{C}^{\text{MI}-2}, \mathcal{C}^{\text{MI}-3}, \mathcal{C}^{\text{HI}})$. s^{HI} is confirmed to be the worst-case scenario and $\mathcal{C}^{\text{HI}}$ is the HI-criticality task pWCET.

5.2 Test Case 2

The second test case is the *lms* task from the Mälardalen WCET benchmark. The task is executed in a multi-core platform concurrently with other interfering task and RTEMS OS within the same core as well as outside it, [20]. While executing, *lms* experience two scenarios $S = \{s^1, s^2\}$ from OS and environmental conditions possible at runtime; two traces of execution time exist, *trace2_ 1* and *trace2_ 2* and they cover all the conditions *lms* can experience. *trace2_ 2* dominates *trace2_ 1* in terms of measured execution times. The two scenarios define LO-criticality and HI-criticality conditions and pWCET models.



(a) *lms* HI-criticality trace; the execution time in ordinate are in CPU cycles

(b) *lms* LO-criticality trace; the execution time in ordinate are in CPU cycles

(c) icdf pWCET with execution time in abscissa are in CPU cycles; ordinate has log scale

Fig. 4. *lms* benchmark with two traces of execution time measurements *trace2_ 1* and *trace2_ 2*.

Figure 4 presents the two traces of measurements and a comparison between $\mathcal{C}^{\text{LO}}$ and $\mathcal{C}^{\text{HI}}$. In particular, Figure 4(c) illustrates the partial ordering between the two criticality level with *trace2_ 2* dominating *trace2_ 2*.

5.3 Test Case 3

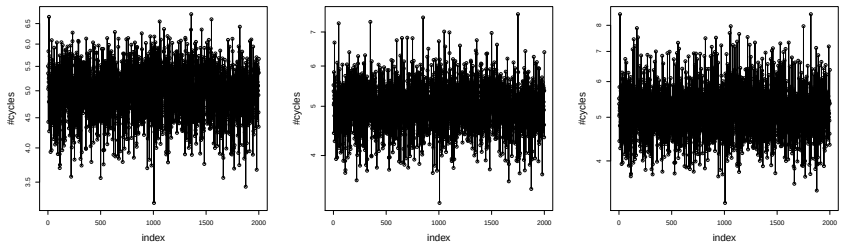
The last test case composes of an artificial task τ_1, $\tau_1 = (([\overline{\mathcal{C}}_1, \langle\overline{C}_1, 10^{-9}\rangle, (\hat{C}_1^{\text{LO}}, \hat{C}_1^{\text{MI}}, \hat{C}_1^{\text{HI}})], 50, 50)$. Task period and deadline coincides and are equal to 50 CPU cycles.

τ_1 can execute under its normal functional behavior (no-fault present) s^{LO}, under fault condition s^{MI} or under the worst fault condition s^{HI}.

τ_1 is an artificial tasks since its normal execution time are considered to follow a Normal distribution and not measured from a benchmark. For s^{MI}, the penalties δ are extracted randomly from a uniform distribution with a defined MTTF applied with Equation (8). Finally, for s^{HI}, the penalties δ are extracted the same uniform distribution but with a smaller MTTF and exhibiting more frequent faults (more critical).

The MC Worst-Case Set representation combines criticality levels (scenarios) and probabilities such that: $\langle C_{1,1}^{\text{LO}} = 8, 10^{-6}\rangle$, $\langle C_{1,2}^{\text{LO}} = 10, 10^{-9}\rangle$, $\langle C_{1,1}^{\text{MI}} =$

$12, 10^{-6}\rangle$ and $\langle C^{\text{MI}}_{1,2} = 14, 10^{-9}\rangle$, and $\langle C^{\text{HI}}_{1,1} = 16, 10^{-6}\rangle$ and $\langle C^{\text{HI}}_{1,2} = 17, 10^{-9}\rangle$, from the Normal and Uniform laws applied. $p = 10^{-6}$ and $p = 10^{-9}$ are chosen arbitrarily, but any exceeding probability applies.



(a) Trace for τ_1 LO (b) Trace for τ_1 MI (c) Trace for τ_1 HI

Fig. 5. Measurements of τ_1 execution time under $S = \{s^{\text{LO}}, s^{\text{MI}}, s^{\text{HI}}\}$ execution conditions. Execution times in ordinate are in CPU cycles.

Figure 5 illustrates traces of measurements for τ_1 obtained as formerly described. With s^{HI} the faults are more frequent, making it the most-critical scenario, and that reflects in the pWCET threshold. Figure 7 details the pWCETs for $S = \{s^{\text{LO}}, s^{\text{MI}}, s^{\text{HI}}\}$; $\mathcal{C}^{\text{HI}}$ is named *task1_hi*, $\mathcal{C}^{\text{MI}}$ is named *task1_mi* and $\mathcal{C}^{\text{LO}}$ is named *task1_lo*. The dominance between scenarios and criticality levels is confirmed validating the MC task model for faults.

To note that between pWCETs in Figure 7 there is not strict dominance, since curves overlaps for small values. The broader dominance as well as the validity of the probabilistic representation is guaranteed at larger values and smaller probabilities, $p \leq 10^{-3}$.

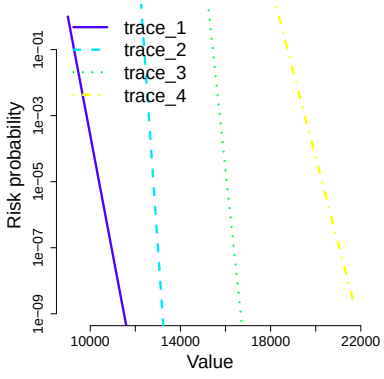


Fig. 6. *ns* pWCET comparison; pWCET represented as icdf and execution time in abscissa are in CPU cycles and ordinate has log scale.

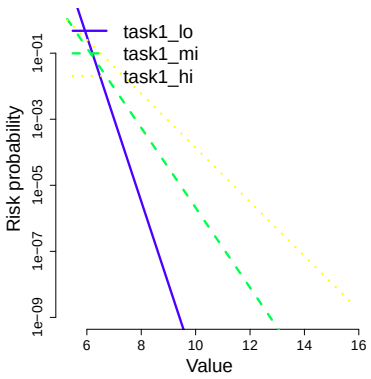


Fig. 7. τ_1 pWCETs represented as icdf; execution times values are in CPU cycles and ordinate has log scale.

6 Conclusion

The work proposed is a probabilistic representation framework for task execution behaviors named Worst-Case Set, which relies on multiple pWCETs for characterizing the diverse scenarios $s^j = f(I, Env, Map, \ldots)$ affecting task executions; probabilities and coverage of multiple execution conditions make the Worst-Case Set flexible and accurate for task models. The Worst-Case Set is applied for MC models (for different scenario, each with a criticality level associated) and for faults and fault effects on task executions (also related to criticality). A case study is presented to validate the probabilistic representation and illustrate its flexibility in modeling multiple task behaviors from diverse scenarios and criticalities.

Future work will apply the Worst-Case Set representations to probabilistic schedulability analysis as well as to develop schedulability strategies that will leverage probabilities and multiple criticality levels. It has been assumed that S was known; future work will be devoted to investigate scenarios complexity and scenarios completeness for safety and criticality guarantees to probabilistic task models.

References

1. Abella, J., Quiñones, E., Wartel, F., Vardanega, T., Cazorla, F.J.: Heart of gold: Making the improbable happen to increase confidence in MBPTA. In: 26th Euromicro Conference on Real-Time Systems, (ECRTS) (2014)
2. Abeni, L., Buttazzo: QoS guarantee using probabilistic deadlines. In: IEEE Euromicro Conference on Real-Time Systems (ECRTS99) (1999)
3. Altmeyer, S., Cucu-Grosjean, L., Davis, R.I.: Static probabilistic timing analysis for real-time systems using random replacement caches. Real-Time Systems 51(1), 77–123 (2015)
4. Burns, A., Davis, R.: Mixed-criticality systems: A review (2015), `http://www-users.cs.york.ac.uk/~burns/review.pdf`
5. Cucu-Grosjean, L.: Independence - a misunderstood property of and for (probabilistic) real-time systems. In: the 60th Anniversary of A. Burns, York (2013)
6. Cucu-Grosjean, L., Santinelli, L., Houston, M., Lo, C., Vardanega, T., Kosmidis, L., Abella, J., Mezzeti, E., Quinones, E., Cazorla, F.J.: Measurement-based probabilistic timing analysis for multi-path programs. In: 23nd Euromicro Conference on Real-Time Systems (ECRTS). IEEE (2012)
7. Cucu-Grosjean, L., Santinelli, L., Houston, M., Lo, C., Vardanega, T., Kosmidis, L., Abella, J., Mezzetti, E., Quiñones, E., Cazorla, F.J.: Measurement-based probabilistic timing analysis for multi-path programs. In: (ECRTS) (2012)
8. Davis, R.I., Santinelli, L., Altmeyer, S., Maiza, C., Cucu-Grosjean, L.: Analysis of probabilistic cache related pre-emption delays. Proceedings of the 25th IEEE Euromicro Conference on Real-Time Systems (ECRTS) (2013)
9. Díaz, J., Garcia, D., Kim, K., Lee, C., Bello, L., J.M., L., Mirabella, O.: Stochastic analysis of periodic real-time systems. In: 23rd of the IEEE Real-Time Systems Symposium (RTSS) (2002)
10. Edgar, S., Burns, A.: Statistical analysis of WCET for scheduling. In: 22nd of the IEEE Real-Time Systems Symposium (2001)

11. Guet, F., Santinelli, L., Morio, J.: On the reliability of the probabilistic worst-case execution time estimates. In: 8th European Congress on Embedded Real Time Software and Systems (ERTS) (2016)
12. Guo, Z., Santinelli, L., Yang, K.: Edf schedulability analysis on mixed-criticality systems with permitted failure probability. In: 21th IEEE International Conference on Embedded and Real-Time Computing System and Applications (2015)
13. Hansen, J., Hissam, S., , Moreno, G.: Statistical- based wcet estimation and validation. In: the 9th International Workshop on Worst-Case Execution Time (WCET) (2009)
14. Hardy, D., Puaut, I.: Static probabilistic worst case execution time estimation for architectures with faulty instruction caches. In: International Conference on Real-Time Networks and Systems (RTNS) (2013)
15. Lehoczky, J.: Real-time queueing theory. In: 10th of the IEEE Real-Time Systems Symposium (RTSS96) (1996)
16. Lima, G., Dias, D., Barros, E.: Extreme value theory for estimating task execution time bounds: A careful look. In: 28th Euromicro Conference on Real-Time Systems, (ECRTS) (2016)
17. Milutinovic, S., Abella, J., Cazorla, F.J.: Modelling probabilistic cache representativeness in the presence of arbitrary access patterns. In: 19th IEEE International Symposium on Real-Time Distributed Computing, (ISORC) (2016)
18. Panerati, J., Abdi, S., Beltrame, G.: Balancing system availability and lifetime with dynamic hidden markov models. In: Adaptive Hardware and Systems (AHS), NASA/ESA Conference on (2014)
19. Pathan, R.M.: Fault-tolerant and real-time scheduling for mixed-criticality systems. Real-Time Systems 50 (2014)
20. Santinelli, L., Guet, F., Morio, J.: Revising measurement-based probabilistic timing analysis. In: Proceedings of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS) (2017)
21. Santinelli, L., Guo, Z., George, L.: Fault-aware sensitivity analysis for probabilistic real-time systems. In: 2016 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT) (2016)
22. Tia, T., Deng, Z., Shankar, M., Storch, M., Sun, J., Wu, L., Liu, J.: Probabilistic performance guarantee for real-time tasks with varying computation times. In: IEEE Real-Time and Embedded Technology and Applications Symposium (1995)
23. Vestal, S.: Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In: Proceedings of the 28th IEEE International Real-Time Systems Symposium. (RTSS), IEEE Computer Society (2007)
24. Wilhelm, R., Engblom, J., Ermedahl, A., Holsti, N., Thesing, S., Whalley, D.B., Bernat, G., Ferdinand, C., Heckmann, R., Mitra, T., Mueller, F., Puaut, I., Puschner, P.P., Staschulat, J., Stenström, P.: The worst-case execution-time problem - overview of methods and survey of tools. ACM Trans. Embedded Comput. Syst. 7(3), 36:1–36:53 (2008)
25. Wong, H.C., Burns, A.: Schedulability analysis for the abort-and-restart (ar) model. In: Proceedings of the 22Nd International Conference on Real-Time Networks and Systems (RTNS) (2014)