Invited Paper: Statistical, Stochastic or Probabilistic (Worst-Case Execution) Execution Time? – What Impact on the Multicore Composability

Liliana Cucu-Grosjean
Kopernic team, Inria, Paris, France
StatInf, Paris, France

Abstract

The problem of estimating worst-case execution times of programs on processors has appeared within the context of critical industries like avionics or space. Rapidly adopted by the real-time scheduling community, worst-case execution time estimates of programs or tasks are mandatory to understand the time behaviour of a real-time system. Analyzing such time behaviour is done, often, with an important pessimism due to the consideration of worst-case scenarios. A decreased pessimism has been obtained by understanding that large execution times of a program have low probability of appearance. Probabilistic (worst-case) execution time notion has been proposed. Nevertheless, independence hypotheses makes difficult today to calculate the probabilistic worst-case execution time of a program and current approaches are built, often, on statistical estimators based on the use of Extreme Value Theory or concentration inequalities. Thus, future probabilistic time analyses are expected to consider worst-case execution times estimates obtained by using statistical estimators on measured execution times instead of probabilistic (worst-case) execution times estimations. Within this paper, we discuss the opportunity of differentiating probabilistic (worst-case) execution times from statistical (worst-case) execution times and how dependence between execution times are better or easier captured by each of the definition, while stochastic execution times could be, also, an appropriate alternative.

2012 ACM Subject Classification Computer systems organization → Embedded systems

Keywords and phrases Worst-case execution time, probabilistic analyses, statistical estimator

Digital Object Identifier 10.4230/OASIcs.WCET.2024.6

Acknowledgements The author would like to thank Marwan Wehaiba, Kevin Zagalo, Ismail Hawila, Slim Ben Amor, Benjamin Rouxel and Yves Sorel for discussions on the appropriate definition of execution times for probabilistic real-time systems. The author thanks, also, Ismail Hawila and StatInf, for providing outputs of the statistical WCET estimator tool, RocqStat, for KDBench programs. This paper puts in writing the keynote presented by the author at the Capital workshop 2024 in Toulouse and the author would like, also, to thank participants of this workshop for their feedback that has, clearly, influenced this paper.

1 Introduction and motivation

Originally introduced to answer DO-178B certification requirements, the problem of estimating worst-case execution time (WCET) of programs or tasks has received increased attention from the real-time community. Static methods for the WCET estimation of a program on a processor, analyzing the program without any execution of the program, has been intensively proposed for different processor architectures [12] and dynamic methods requiring execution of programs have received a recent interest due to the arrival of more complex processors [4]. This interest has motivated the introduction of new methods to overcome the possible optimism of dynamic methods. Indeed, while static methods are recognized to

© Liliana Cucu-Grosjean;
licensed under Creative Commons License CC-BY 4.0
22nd International Workshop on Worst-Case Execution Time Analysis (WCET 2024).
Editor: Thomas Carle; Article No. 6; pp. 6:1–6:10
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

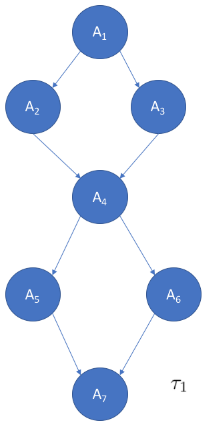

Figure 1 A possible view for a program τ_1.

be safe, but pessimistic, dynamic methods have the reputation of being unsafe since their estimation is based, only, on what it has been observed or measured. Probabilistic worst-case execution time has been proposed within this context by Bernat and Petters [1] in order to underline that based on some probabilistic reasoning, the observed execution times obtained by dynamic methods are enriched in order to achieve safeness. As such, this definition generalizes the static WCET definition but it comes with a strong hypothesis: probability distributions at some granularity level of programs, e.g., basic blocks, should be independent or any combinations of them should include a safe description of possible dependence between those basic blocks. We understand here by basic blocks, linear sequences of instructions, without any branch. One may note that this is different from the problem of dependence between probability distributions describing probabilistic execution times of two instances of the same program or instances of different programs as described in [2], even if a relation does exist between them (see Section 3). Indeed, probabilistic worst-case execution definition has appeared as the result of probabilistic operations which are done in order to obtain the probabilistic distribution of a (worst-case) execution time for a program. For instance, if one considers a program τ_1 with its basic blocks $A_1, A_2, A_3, \cdots, A_7$ as described in Figure 1, then probabilistic operations (convolutions, dominance relation for instance) between the probability distributions of those basic blocks could be done to propose a probabilistic (worst-case) execution time estimation for the program τ_1. Another alternative is building or calculating a dominance relation between two probability distributions, which is possible in absence of any particular mathematical properties, while the convolution between the probability distribution of A_1 and the $max(A2, A3)$ requires to understand the dependence that may exist between them. Such understanding may be built on input variables the two basic blocks share but also on states at which the execution of A_1 leaves processor features after its execution. Within this paper and for the sack of simplicity, we focus on the variation of input variables and/or the existence of several cores.

Our main contributions are the following:

- we provide discussions on the fact that probabilistic (worst-case) execution times and statistical (worst-case) execution times should co-exist. Moreover, we discuss the opportunity of using the identically distributed hypothesis within the context of probabilistic (worst-case) schedulability analyses:

- we underline the misleading relation between dependence hypothesis at basic block level and at the instances of a program and conclude on the possibility that new execution time models are needed to include both types of dependence;
- we propose a first discussion on how statistical WCET estimators may bring time composability to the multicore problem.

Organization of the paper. We provide in Section 2 main definitions and notations used to introduce our contribution. In Section 3 we compare probabilistic (worst-case) execution time and statistical (worst-case) execution time definitions as well as the place of identically distributed and independent hypotheses. In Section 4, we provide hints on how statistical WCET estimators are good candidates for the multicore problem.

2 Notations and related work

Within this paper, we consider a set n programs (or tasks) τ_i executed on a processor π[1]. A program or task τ_i, $\forall i \in \{1, 2, .., n\}$ is defined by C_i its execution time defined by a cumulative distribution function F_{C_i} (see Equation (1)), where $\Omega_0 = Q \times I$ is the product space between Q is the set of possible states of the processor π and I is the set of all possible input values of tasks τ_i [13].

$$F_{C_i}(c) = P_{C_i}((-\infty, c)) = P(\omega_0 \in \Omega_0 : C_i(\omega_0) \leq c) \tag{1}$$

One may underline that $F_{C_i}(c)$ defines the probability for the execution time C_i to be smaller than c. Indeed, within the real-time community, one is interested in the exceedance function $1 - F_{C_i}$ which defines the probability for the execution time C_i to be larger than c. Such exceedance function is, often, addressed as the probabilistic (worst-case) execution time of a task or program. One may consider the obtention of such function by two main classes of methods:

- probabilistic approaches - they combine information at some granularity level, e.g., basic blocks. For instance, for the program τ_1 introduced in Section 1 (see Figure 1), one may calculate this distribution using the distributions of all basic blocks under appropriate mathematical hypotheses;
- statistical approaches - they use statistical estimators on measured execution time at some granularity level, e.g., between the beginning and the end of the program τ_1.

Few results [4] are proposed for the first class of approaches as they require an important understanding on how the probability distribution of a basic block has an impact on the probability distributions of another basic block. This first method has been proposed within the literature together with a definition for the probabilistic worst-case execution time [1] and this definition has been, often, used when execution times are described by exceedance functions. Nevertheless, in [1], no independence, nor identically distributed hypothesis is discussed. The second class of approaches [6] has been proposed in parallel with [1], but the authors do not introduce a definition for the freshly proposed notion of probability for the execution time to exceed a given value. They do provide theoretical bases for an important existing observation from [11], the distribution of execution times is heavily tailed, i.e., the probability of appearance of large values for the execution times are low and large values

[1] For the sake of the simplicity, we consider a simple processor as more complex architectures do not modify the conclusions of our paper

are much larger than average ones. In parallel, a schedulability analysis proposed for a set of tasks with independent and identically distributed (i.i.d) random variables describing execution times is proposed [5]. Its strong i.i.d. hypothesis is required to prove a stationarity property allowing to conclude on the schedulability of the system. This full i.i.d hypothesis is, often, used within schedulability analyses by more recent authors, without necessarily being used entirely. Indeed, while the independence hypothesis is required to operate a convolution, there is no need for two distributions to be identically distributed for such convolution to be operated [3]. Usually the identically distributed hypothesis is required to prove or to use convergence results mainly in statistics or stochastic processes. More precisely, one important and correct use of the identically distributed hypothesis is done within the application of statistical estimators on ordered sequences of measured execution times obtained by following some measurement protocols. Within this context, we resume below latest results on statistical approaches as proposed in [7]. The WCET of a program or task τ_i is defined as its largest execution time for any valid execution scenario S. During each scenario S_j, execution times are collected as ordered sequences of execution times. Statistical estimators are applied on these sequences and i.i.d properties are checked. In [7], an Extreme Value Theory estimator[2] is applied to sub-sequences where a sub-sequence contains i.i.d. execution times. Finally, the WCET estimation is obtained by building an envelop on all sub-sequences WCET estimations. Such mathematical operation introduces **a time composability between sWCET estimations** of different sub-sequences of execution times. One may notice that these sub-sequences correspond to different execution modes.

When dealing with dynamic approaches for WCET estimation of a program on a processor, one may be interested in identifying the paths within the program that produces an execution time. Following the definition proposed in [7], a path P_j of a program τ_i is critical with respect to a pWCET estimation of A if, at least, one measurement of the execution of that path appears within the sub-sequence contributing to the pWCET estimate of A. Always in [7], the authors define a domination relation that formally underlines that a path within one execution scenario may produce execution times that participate to the WCET estimation, while within another execution scenario, this is not the case. For instance, in Figure 2, we consider an ordered sequence (from the left to the right of the figure) measured during a simulated flight for the Sensors program (KDBench programs, more details in [9]). Execution times obtained by exercising the same paths are colored with the same color while execution times within two consecutive vertical lines constitute a sub-sequence on which the statistical estimator is applied.

In this paper and in order to distinguish between WCET estimations obtained by using probabilistic approaches and WCET estimations obtained by using statistical approaches, we call the second ones as the statistical worst-case execution time (sWCET) estimates. In Figure 3, we illustrate the sWCET estimation obtained as an envelop built on top of sWCET estimations obtained for each sub-sequence given in Figure 2.

3 Two definitions and one identically distributed and independent hypothesis within the context of estimating (worst-case) execution times and probabilistic schedulability analyses

In this section, we propose a deeper discussion on hypotheses of independent and identically distributed and their relation with pWCET and sWCET notions as well as their impact on a schedulability analysis.

[2] An interested reader may find more details on such estimators in [4].

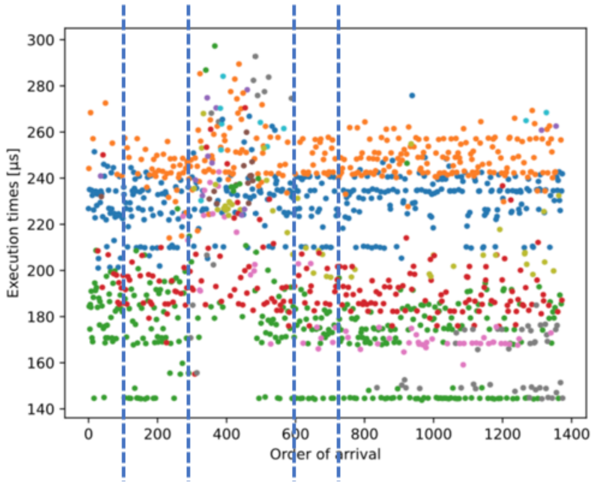

Figure 2 An ordered sequence of execution times for the Sensors program (KDBench programs [8]).

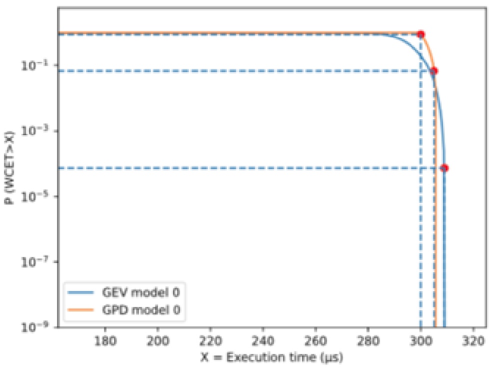

Figure 3 sWCET illustration obtained as an envelope (KDBench programs, see [9]).

Probabilistic approaches do not require any identically distributed hypothesis among basic blocks (or other parts of the program) to produce the pWCET of that program, but they require probability distributions of those basic blocks to be independent, or if not, one has to describe the dependence between these blocks. If we move at a higher level, then a schedulability analysis requires to understand the dependence relation between pWCET of different tasks or their instances. Is a probabilistic approach able to provide such understanding? With our current understanding of the literature, the answer is negative. Indeed, probabilistic approaches are static analysis-based and they do not provide information on different instances of a program, e.g., what path is executed at some time instant. Let us consider the set of tasks $\tau = \{\tau_1, \tau_2, \tau_3\}$, where the internal structure of τ_1 is illustrated in Figure 1, of τ_2 in Figure 4 and of τ_3 in Figure 5.

We consider now a possible schedule illustrated in Figure 6, where exercised paths are colored in green. A pWCET estimation does not distinguish between different paths, neither of different execution contexts. Actually, if one looks at path level, then it does not exist any current probabilistic schedulability analysis considering paths to be associated with some program instances (or jobs). Indeed, considering dependence relations between consecutive

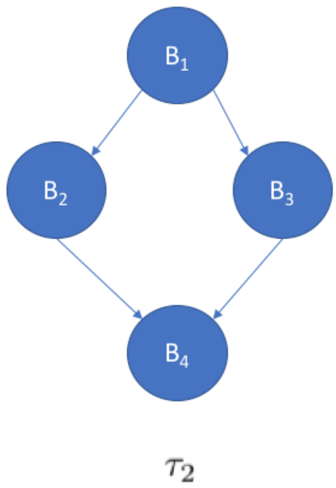

Figure 4 A possible view for a program τ_2.

execution times requires, also, to understand if exercising $B_1 - B_3 - B_4$ in τ_2 increases the probability for τ_3 to exercise $C_1 - C_2 - C_4 - C_5$, for instance. pWCET estimations as defined today within the literature do not include such information. Moreover, the independence hypothesis at basic block level does not help to advance towards estimating inter-programs or inter-instances dependence.

Coming back to the identically distributed hypothesis, its utilization for integrating pWCET estimations within a schedulability analysis is not necessary to provide correct results. As underlined within the introduction, the convolution between probability distributions does not need this mathematical property. Looking at realistic executions, while different paths do indicate the existence of multimodal distributions [14][3], the static analysis-based reasoning of pWCET estimators do not allow to differentiate among these paths and consecutive instances of the same program may have different distributions if they exercise different paths.

Statistical approaches may require the i.i.d. hypothesis for sequences of measured execution times in order to obtain a sWCET estimated. For instance, in Figure 2 i.i.d sub-sequences of execution times are considered in order to apply statistical estimators only to the execution times within the same sub-sequence. Finally, the sWCET may be obtained by building a probability distribution upper-bounding all sWCET estimations obtained per sub-sequence. Thus, a dominance relation is built within the final sWCET estimation. To the best of our knowledge, using such sWCET within a schedulability analysis has never been considered in the literature. Even if pWCET and sWCET provide a probability distribution to the schedulability analysis, their estimations require different level of information on the variation of execution times. Since sWCET estimation is expected to consider as input, execution times obtained in real execution conditions, then the sequences of execution times are measured with respect to a given scheduling algorithm. The evolution of execution times do capture dependence relations introduced by the scheduling algorithm. While the paths exercised by a program during its execution are, mainly, imposed by the variation of input variables of that program, the variation of execution times per path may be impacted by the choice of the order in which programs are executed.

[3] We introduce intuitively the notion of multimodal distribution as a distribution with several peaks

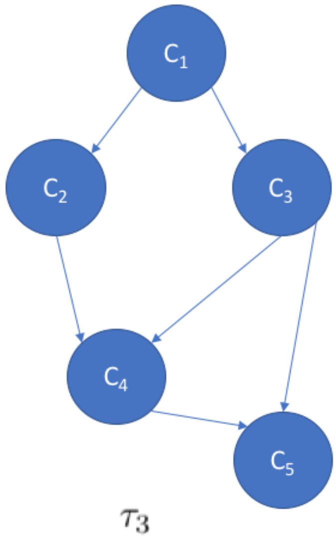

Figure 5 A possible view for a program τ_3.

If one wants to capture this evolution and introduce it within a schedulability analysis, stochastic processes are an alternative - existing results as [10] (to cite the latest, to the best of our knowledge) are promising and one may consider what is a stochastic (worst-case) execution and its relation to pWCET and sWCET definitions. Moreover, a formal pWCET definition provided in [2] indicates that stochastic processes fulfill hypotheses allowing to built correct schedulability analysis.

4 From single to multicore processors

While the probabilistic WCET estimation has received important attention in the case of programs executed on one core processor, the statistical WCET estimation is, in our opinion, a more promising candidate for programs executed on multicore processors or in presence of operating systems. The main limitation is the strong pWCET hypothesis of independence between probability distributions describing the execution time of basic blocks. By adding new interference sources, the multicore case increases the complexity of describing such dependence relations. In the case of sWCET estimators, dependence relations between consecutive executions of a program does help to detect different sub-sequences of execution. These sub-sequences identify different execution modes that could be provoked either by the execution of different paths, but also by the evolution of hardware states like multicore interferences. In Figure 7, the execution times of a program are obtained on a 4 cores processor with a variation of the execution time dependent on an input variable. Each core has local cache (data and instruction) memories and there is one global cache data memory that is shared by all cores. The execution times are presented from the left to the right in the order of their measurement, while on the vertical axis, the values of execution times are provided in cycles.

We underline the existence of 4 groups of execution times - the lower group is obtained when only the core executing the program is active, the second from the bottom is obtained when a second core is active, etc. Within the same group, execution times are obtained respectively, without active cache memory, with active cache instruction memory and with active cache data memory. The highest value is obtained in presence of cache memory shared

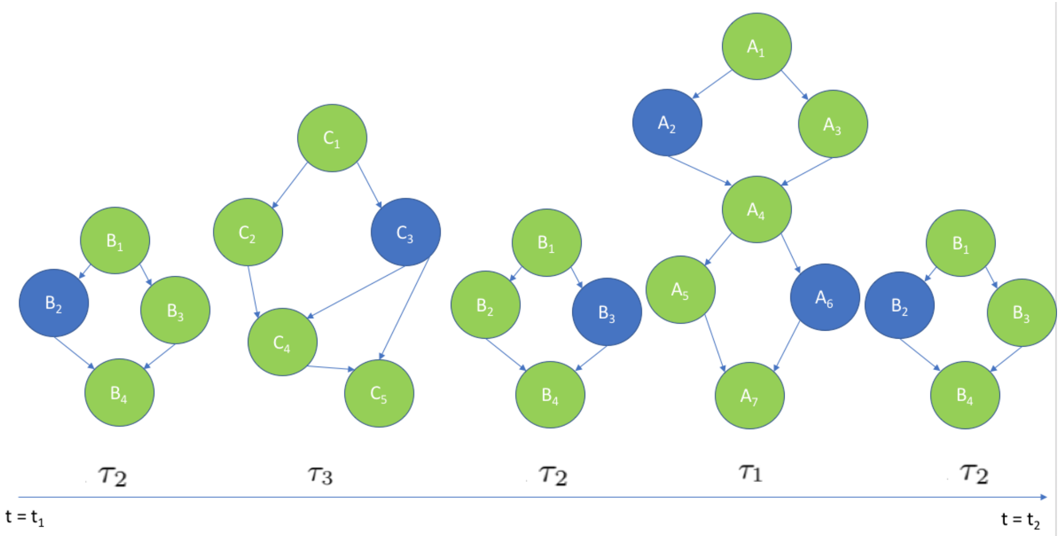

Figure 6 A schedule from t_1 to t_2 of programs τ_1, τ_2 and τ_3 and their paths. The exercised paths are illustrated in green.

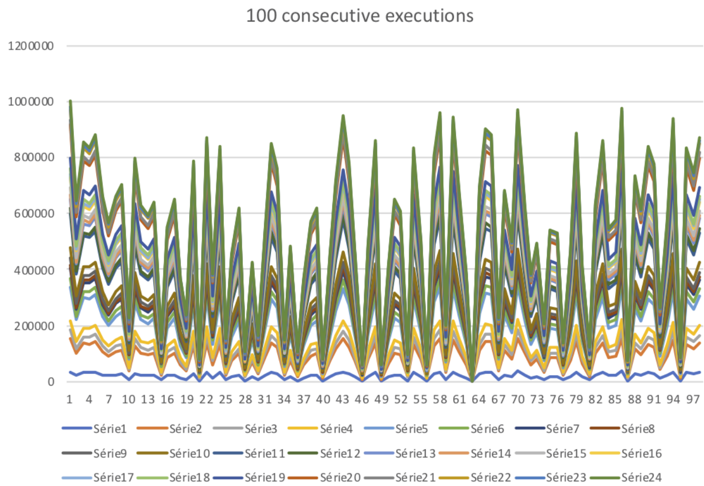

Figure 7 The evolution of execution times for a program executed 100 times with the same sequence of input variables and the execution time is dependent of the variation of one input variable illustrated in blue.

by all cores. Since the processor does not include pipelines, nor branch predictors, the execution times are ordered in layers and a statistical estimator provides sWCET estimations with a nice dominance property. In reality, these layers of execution times may cross each other and the sWCET estimations per core could be visualized as in Figure 8. Comparing the envelop built per core is a possible way to estimate the penalty of a core but the most important is that building a global envelop ensures **a time composability between sWCET estimated for a program executed in the presence of several cores**. Considering stochastic WCET estimation for programs executed on several cores is an open problem, but its common hypotheses with the sWCET estimation is promising. Including such estimations (stochastic or statistical) within multicore schedulability analysis is an open problem.

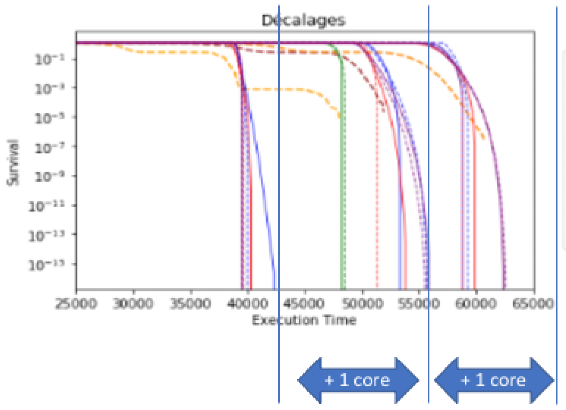

Figure 8 SWCETs cross each other and the envelop per core allows to calculate the penalty for each new core.

5 Conclusions

In this paper, we propose a discussion on how three definitions for describing the probability that the execution time is larger than a given value have been introduced within the real-time community. Understanding their estimation is an important step towards their correct integration within higher-level time analysis and their hypotheses may prevent some of them from such integration. We conclude the paper by presenting hints on how sWCET estimations may provide a time composability answer to the WCET multicore estimation problem.

References

1. Guillem Bernat, Antoine Colin, and Stefan M. Petters. pWCET, a tool for probabilistic WCET analysis of real-time systems. In Jan Gustafsson, editor, *Proceedings of the 3rd International Workshop on Worst-Case Execution Time Analysis, WCET 2003*, volume MDH-MRTC-116/2003-1-SE, pages 21–38, 2003.
2. S. Bozhko, F. Markovic, G. von der Bruggen, and B. Brandenburg. What really is pWCET? In *2023 IEEE Real-Time Systems Symposium (RTSS)*, pages 13–26, 2023.
3. L Cucu-Grosjean. Independence-a misunderstood property of and for probabilistic real-time systems. *Real-Time Systems: the past, the present and the future, the Alan Burns 60th anniversary*, pages 29–37, 2013.
4. R. I. Davis and L. Cucu-Grosjean. A survey of probabilistic timing analysis techniques for real-time systems. *LITES*, 6(1):03:1–03:60, 2019.
5. José Luis Díaz, Daniel F. García, Kanghee Kim, Chang-Gun Lee, Lucia Lo Bello, José María López, Sang Lyul Min, and Orazio Mirabella. Stochastic analysis of periodic real-time systems. In *Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS'02),*, pages 289–300. IEEE Computer Society, 2002.
6. S. Edgar and A. Burns. Statistical analysis of WCET for scheduling. In *the 22nd IEEE Real-Time Systems Symposium*, 2001.
7. Marwan Wehaiba el Khazen, Slim Ben-Amor, Kossivi Kougblenou, Adriana Gogonel, and Liliana Cucu-Grosjean. Work in progress: Towards a statistical worst-case energy consumption model. In *29th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS 2023*, pages 333–336. IEEE, 2023.

8 Marwan Wehaiba el Khazen and et al. Work in progress: Kdbench - towards open source benchmarks for measurement-based multicore WCET estimators. In *28th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS*, pages 309–312. IEEE, 2022.

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

10 Anna Friebe, Filip Markovic, Alessandro Vittorio Papadopoulos, and Thomas Nolte. Continuous-emission markov models for real-time applications: Bounding deadline miss probabilities. In *29th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS*, pages 14–26. IEEE, 2023.

11 Mark K. Gardner and Jane W.-S. Liu. Analyzing stochastic fixed-priority real-time systems. In *Tools and Algorithms for Construction and Analysis of Systems, 5th International Conference, TACAS '99*, volume 1579 of *Lecture Notes in Computer Science*, pages 44–58. Springer, 1999.

12 R. Wilhelm, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G.Bernat, C. Ferdinand, R.Heckmann, T. Mitra, F. Mueller, I. Puaut, P. Puschner, G. Staschulat, and P. Stenströem. The worst-case execution time problem: overview of methods and survey of tools. *Trans. on Embedded Computing Systems*, 7(3):1–53, 2008.

13 Kevin Zagalo, Yasmina Abdeddaïm, Avner Bar-Hen, and Liliana Cucu-Grosjean. Response time stochastic analysis for fixed-priority stable real-time systems. *IEEE Trans. Computers*, 72(1):3–14, 2023.

14 Kevin Zagalo, Liliana Cucu-Grosjean, and Avner Bar-Hen. Identification of execution modes for real-time systems using cluster analysis. In *25th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, pages 1173–1176. IEEE, 2020.