Work-in-Progress paper:

An Analysis of the Impact of Dependencies on Probabilistic Timing Analysis and Task Scheduling

Enrico Mezzetti[†], Jaume Abella[†], Carles Hernandez[†], Francisco J. Cazorla^{†,‡}

[†]Barcelona Supercomputing Center (BSC-CNS)

[‡]Spanish National Research Council (IIIA-CSIC)

Abstract—Recently there has been a renewed interest for probabilistic timing analysis (PTA) and probabilistic task scheduling (PTS). Despite the number of works in both fields, the link between them is weak: works on the latter build upon a series of assumptions on the probabilistic behavior of each task – or instances (jobs) of it – that have not been shown how to be fulfilled by PTA. This paper makes a first step towards covering this gap with emphasis on providing the right meaning of pWCET estimate as understood by both PTA and PTS. We show that the main issue related to ensuring that PTS assumptions on pWCET estimates are captured by PTA relates to the dependencies among tasks, and even jobs of a given task. Both change the scope of applicability of pWCET estimates provided by PTA and hence, their use by PTS.

I. INTRODUCTION

Probabilistic timing analysis (PTA) – and its measurement-based variant (MBPTA) in particular – has arisen as an approach to deliver Worst-Case Execution Time (WCET) estimates for platforms comprising high-performance features, including cache hierarchies and multicores [7]. PTA delivers a probabilistic WCET (pWCET) distribution where each execution time value has an associated exceedance probability that upper-bounds the true probability of exceeding such execution time. However, the particular state of the hardware and software during timing analysis determines the *applicability* of pWCET distributions. For instance, the relation between the state of hardware and software at analysis and during operation for a given task may make pWCET distributions be valid for a task, but not independent across tasks. This relates to execution time dependencies during operation and how they are accounted for at analysis.

The relationship between pWCET distributions and their applicability during operation is of prominent importance for task scheduling as part of the response time analysis. Probabilistic Task Scheduling (PTS) addresses those schedulability analysis approaches where at least one parameter in the task model is defined through a random variable. In this work we focus on those methods where the WCET of each task is characterized as a probability distribution. PTS has been proposed [8] and adapted to different scheduling schemes [6], [12] and paradigms [10]. All those PTS approaches assume that pWCET distributions are *independent*, meaning that subsequent activations of jobs (whether from the same task or not) do not carry time dependencies, that is, the same pWCET distribution can be used across jobs. While this form of independence is assumed to be fulfilled by construction in pWCET distributions computed via Static Probabilistic Timing Analysis (SPTA), the same may not be told for those derived by MBPTA. In particular, whether jobs' pWCET can be regarded as independent or not determines whether PTS is feasible or not.

In this work, we settle the ground for PTS by identifying the key properties that make pWCET distributions be independent, and so compatible with PTS, regardless of the approach used to derive them. To this extent, we abstract away from any specific PTS approach and we focus, instead,

on the more generic concern on whether and how pWCET distributions can be combined together. In particular, we address pWCET independence of tasks and show how this holds despite the existence of dependencies during operation. Moreover, we show examples where pWCET distributions do not provide the independence needed for PTS and how this may influence task scheduling. Part of our future work consists of relating the particular scenarios described to specific hardware/software platforms and evaluate them with benchmarks and industrial case studies to quantify the impact of making pWCET distributions be independent or, conversely, account for dependencies during task scheduling.

II. CONCEPTS AND DEFINITIONS

In this section we describe the main concepts upon which this work builds for the sake of clarity.

pWCET distribution. PTA – and MBPTA in particular – delivers a probability distribution function (PDF), which can be expressed as an exceedance function. For MBPTA, such exceedance function is often obtained by applying Extreme Value Theory (EVT) [3] on a set of execution time measurements of the task under analysis [5], [2], [9], [11]. The exceedance function describes the maximum probability with which each execution time can be exceeded. Hence, it is also referred to as pWCET distribution.

pWCET estimate. A pWCET estimate stands for the lowest execution time whose exceedance probability in the pWCET distribution is below a given exceedance threshold. Such threshold is an exceedance probability per run of the task under analysis (e.g. 10^{-12} per run). It is typically set low enough so that it can be assumed negligible w.r.t. the particular domain and criticality of the task [5].

Probabilistic Task Scheduling (PTS). Conventional task scheduling builds upon individual WCET estimate values. However, PTS builds upon pWCET distributions with the aim of obtaining shorter probabilistic worst-case response times. In particular, PTS aims at exploiting the fact that, probabilistically, some tasks will run below their pWCET estimates for the applicable exceedance threshold, which allows some other tasks to overrun their exceedance thresholds. Hence, by jointly considering their pWCET distributions, lower pWCET estimates may be accounted for while still preserving the risk of exceeding the overall execution time budget below acceptable limits (in relation with the particular domain and certification constraints).

Probabilistic Independence of pWCET. Two tasks are said to exhibit probabilistically independent pWCET distributions if and only if there exists no correlation among them. Probabilistic independence of pWCET distributions, as required by PTS to enable convolutions among distributions, must not be confused with statistical independence of measurements. Statistical independence, which is often assessed in the application of MBPTA, is a property of a sample, and it is typically assessed by applying statistical tests. Probabilistic pWCET independence between tasks, instead, refers to WCET distributions that do not depend on a specific job instance or job sequence.

III. INDEPENDENT pWCET DISTRIBUTIONS AND PTS

MBPTA aims at obtaining pWCET distributions at analysis that hold during operation. To establish such a relationship, one must consider the hardware and software state that will determine the execution time of each instance (job) of the task during operation. In general, those conditions can be, at most, only partially determined at analysis and some assumptions need to be made that affect both the interpretation and applicability of the pWCET distributions obtained. For instance, the execution path of a job has a direct influence on its execution time. Therefore, the particular paths triggered at analysis and how they are considered determines how the pWCET distribution relates to the execution time of the jobs during operation. This ultimately defines the *scope* of a pWCET distribution: one job, each job, or any sequence of jobs. Since enumerating all potential combinations of conditions at analysis and operation to analyze their relationship is unaffordable, next we describe a reference scenario that allows using pWCET distributions unconstrainedly for PTS by accounting implicitly for any dependence that may occur during operation.

A. Execution Time Distributions

A number of sources may affect the execution time of a task. Those may relate to, for instance, the initial hardware state (i.e. cache contents before execution), the initial software state (i.e. input values determining the execution path), and hardware/software state during execution (i.e. contention in the access to shared hardware resources or state of software shared resources). Whether their impact is deterministic or probabilistic depends on how their state is modified. For instance, if a given input value determines the execution path of the task and such value is relevant for the functional behavior of the task, such input value is determined by the functional needs during operation. Hence, given a fixed state for all other hardware/software sources of execution time variation (*setv*), each different value of this particular input leads to a deterministic execution time.

Instead, if such input value has no influence on the functional behavior of the task (e.g. seed of the pseudo-random number generator of a random replacement cache), it can be set to a random value. Hence, given a fixed value for any other *setv*, the population of (random) values for this input leads to a probabilistic execution time distribution. In general, we regard all *setv* as deterministic except those that can be set (and are set during operation) to a random value without impacting the functional behavior of the task.

When considering together all *setv*, some of them are regarded as deterministic, whereas others are random. This makes that any given combination of values for the deterministic parameters *determines* the particular execution time distribution obtained due to random parameters. Hence, if the number of combinations of the deterministic parameters is N , we have N different execution time distribution functions at analysis. During operation each job will instantiate one of those N distributions. If no further information is available, we cannot establish any kind of probabilistic relation between the particular distribution instantiated by each job during operation, and whether dependencies exist across the distributions instantiated by jobs belonging to different tasks.

In order to account at analysis for the execution time dependence on the particular conditions experienced during operation, while still obtaining independent pWCET estimates that can be used for PTS, we propose enforcing *worst-case independence during analysis*. This is illustrated with the example in Figure 1, where a task has two *setv*. The first *setv* is deterministic and can take 4 different values only ($N = 4$), whereas the second *setv* is random. Hence, we have an exceedance function for each of the 4 different values of the

Fig. 1. Example of the execution time distributions with 4 different conditions and 2 pWCET distributions in the form of exceedance functions.

deterministic *setv*. In this example, worst-case independence is achieved by using as pWCET distribution any distribution whose exceedance probability for any execution time is equal or higher than all the exceedance probabilities for such execution time across the different potential execution time distributions. This is illustrated in Figure 1 where I1, I2, I3, I4 stand for the 4 potential execution time distributions, and P1 and P2 stand for two pWCET distributions. As shown, P1 and P2 upper-bound all 4 I_n distributions (where $1 \leq n \leq 4$). Obviously, due to tightness reasons, P1 would be the preferred choice.

B. PTS with Independent pWCET Distributions

As shown in [4], independent pWCET distributions enable PTS since, on a job level, all *setv* that may exhibit dependencies during operation, are assumed to behave in their worst case at analysis. Note that this holds as long as local worst cases lead to the global worst case (hence, no timing anomalies exist). Alternatively, if timing anomalies exist, then those *setv* must be enforced to lead to the combined worst case, which may be more challenging to control.

We illustrate this behavior with a particular example¹. Let us assume a program P1 with an input parameter that can take two values: I1 and I2. The Execution Time Profile (ETP), aka the discrete PDF, for both values consists of two vectors, one with the different latencies the program can take and another with their corresponding probabilities. In general, the ETP of a program is unknown, but here we use them to ease the explanation. In our example, those ETPs are as follows:

$$ETP_{I1} = \{\{10, 20\}, \{0.9, 0.1\}\}$$

$$ETP_{I2} = \{\{15, 25\}, \{0.99, 0.01\}\}$$

Hence, the tightest pWCET distribution would be described by the following ETP:

$$ETP_{pW(P1)} = \{\{15, 20, 25\}, \{0.9, 0.09, 0.01\}\}$$

Let us assume another program P2, with also two values, J1 and J2, whose ETPs are as follows:

$$ETP_{J1} = \{\{15\}, \{1.0\}\}$$

$$ETP_{J2} = \{\{10, 20\}, \{0.8, 0.2\}\}$$

$$ETP_{pW(P2)} = \{\{15, 20\}, \{0.8, 0.2\}\}$$

Since $ETP_{pW(P1)}$ and $ETP_{pW(P2)}$ are independent, we can convolve them for PTS. In particular, the resulting ETP of the convolution, $ETP_{pW(P1+P2)}$, would be as follows:

$$ETP_{pW(P1+P2)} = \{\{30, 35, 40, 45\}, \{0.72, 0.252, 0.026, 0.002\}\}$$

If, during operation, dependencies exist, the real combination of execution time distributions experienced would still

¹pWCET distributions computed with EVT always correspond to continuous distributions. For the sake of illustration, however, it is common practice to consider discrete pWCET distributions instead, as they are easy to operate on. Discrete distributions can be produced at the required granularity by collapsing probabilities for a given range of values to the highest value in that range.

be upper bounded by $ETP_{pW(P1+P2)}$. For instance, if I2 and J2 occur systematically during operation, the real ETP would be as follows:

$$ETP_{pW(I2+J2)} = \{\{25, 35, 45\}, \{0.792, 0.206, 0.002\}\}$$

It can be observed that $ETP_{pW(I2+J2)}$ is upper bounded by $ETP_{pW(P1+P2)}$ for any exceedance probability. For instance, for exceedance probabilities 0.79 and 0.8, the exceedance values are 35 in both cases for $ETP_{pW(P1+P2)}$, and 25 and 35 respectively for $ETP_{pW(I2+J2)}$.

However probabilistic independence of pWCET turns out to be a demanding assumption, especially in the case of MBPTA. In fact, we contend independent pWCET cannot be simply assumed and must be explicitly enforced.

C. Achieving Independent pWCET Distributions

While SPTA already accounts for the worst-case conditions by construction, appropriate measurement protocols and, potentially, hardware/software support are needed in the case of MBPTA. Such support relates to enforcing each individual condition (and their combinations thereof) to be either random (with the same probability distribution) during both analysis and operation, or upper bounded at analysis. The latter, upper bounding, releases the end user from having to exercise any control on those conditions during operation regardless of the dependencies that may occur in practice. For instance, upper bounding may account for path information and initial cache state. This would imply that the pWCET distribution upper bounds the individual execution time distribution for any potential execution path and for any potential initial cache state. The former could be achieved by either exploring all paths, by exploring at least the set of paths that produce the highest execution time for any exceedance probability, or by manipulating the measurement collection process so that any of the former is accounted for from a limited number of paths, as in the case of EPC technique [13]. The latter could be achieved, for instance, by flushing caches prior to execution at analysis if they are write-through (so no dirty lines exist). Any other initial cache state can only lead to shorter execution times. If caches are write-back, and hence dirty lines may exist, one may need to flush caches both at analysis and during operation so that, despite the initial state is not the worst potential state, the same state considered at analysis is enforced during operation and hence, pWCET distributions remain valid for the execution time behavior during operation.

Regarding randomized *setv*, it is critically important noting that they must be purely random for all jobs of the task, with no dependencies on any other task or previous state. Failing to achieve this may break pWCET distribution independence and hence, break the assumptions needed for PTS. This case is analyzed in the next section.

IV. DEPENDENT PWCET DISTRIBUTIONS AND PTS

While dependencies across measurements of a given task have already been considered in the literature [11], [3], in this work we focus on dependencies across tasks. If pWCET distributions of different tasks are not independent, they cannot be convolved since convolution is valid only upon the assumption of independence. Let us recall the example in previous section, but assuming that there exists the following dependence: whenever P1 takes 25 cycles (due to random variations), P2 takes systematically 20 cycles. Hence, the timing behavior of P2 is still random, since whether it takes a particular latency depends on random choices, but it is no longer independent. In this particular case, the ETP of both programs combined would be as follows:

$$ETP_{pW(P1+P2)}^{dep} = \{\{30, 35, 40, 45\}, \{0.72, 0.252, 0.018, 0.01\}\}$$

For exceedance probabilities between 0.002 and 0.01 $ETP_{pW(P1+P2)}^{dep}$ is no longer upper bounded by $ETP_{pW(P1+P2)}$. Hence, if pWCET distributions are not independent, PTS cannot be applied freely building upon convolutions. Still, if dependencies exist, one may resort to deterministic task scheduling by building upon the pWCET estimates at the desired exceedance probabilities, and operating them as absolute WCET values. For instance, if such exceedance probability is 0.005, one would use as pWCET estimates 25 and 20 for P1 and P2 respectively. Hence, both programs would be assumed to take 45 cycles in total. Analogously, if the exceedance threshold was 0.05, then the pWCET estimates would be 20 for both programs and thus, 40 cycles in total. Note that such value would only be exceeded with a probability of 0.01 according to $ETP_{pW(P1+P2)}^{dep}$, thus below the exceedance threshold deemed as acceptable for this particular example.

V. SUMMARY AND FUTURE WORK

The relationship between PTS and the pWCET distributions obtained with MBPTA has not been studied yet to a sufficient extent to understand when and how PTS is applicable. This paper makes a first step towards identifying the conditions needed to enable PTS on the pWCET distributions provided by MBPTA. Our ongoing and future work consists of further analyzing the types of dependencies that may exist across pWCET distributions and estimates, the causes behind those dependencies, to what extent they can be removed or mitigated, and their implications on PTS.

ACKNOWLEDGEMENTS

This work has been partially supported by the Spanish Ministry of Science and Innovation under grant TIN2015-65316-P. Jaume Abella has been partially supported by the Ministry of Economy and Competitiveness under Ramon y Cajal fellowship number RYC-2013-14717. Carles Hernández is jointly funded by the Spanish Ministry of Economy and Competitiveness and FEDER funds through grant TIN2014-60404-JIN.

REFERENCES

- [1] J. Abella et al. WCET analysis methods: Pitfalls and challenges on their trustworthiness. In *SIES*, 2015.
- [2] J. Abella et al. Measurement-based worst-case execution time estimation using the coefficient of variation. *ACM Trans. Des. Autom. Electron. Syst.*, 22(4):72:1–72:29, June 2017.
- [3] S. Coles. *An Introduction to Statistical Modeling of Extreme Values*. Springer, 2001.
- [4] L. Cucu-Grosjean. Independence - a misunderstood property of and for probabilistic real-time systems. In N. Audsley and S. Baruah, editors, *Alan Burns 60th Anniversary*, York, United Kingdom, 2013.
- [5] L. Cucu-Grosjean et al. Measurement-based probabilistic timing analysis for multi-path programs. In *ECRTS*, 2012.
- [6] J.L. Diaz et al. Stochastic analysis of periodic real-time systems. In *RTSS*, 2002.
- [7] L. Kosmidis et al. Fitting processor architectures for measurement-based probabilistic timing analysis. *Microprocessors and Microsystems*, 47:287 – 302, 2016.
- [8] J. P. Lehoczky. Real-time queueing theory. In *RTSS*, 1996.
- [9] G. Lima et al. Extreme value theory for estimating task execution time bounds: A careful look. In *ECRTS*, 2016.
- [10] A. Masrur. A probabilistic scheduling framework for mixed-criticality systems. In *DAC*, 2016.
- [11] L. Santinelli et al. On the sustainability of the extreme value theory for WCET estimation. In *WCET Workshop*, 2014.
- [12] H. Zhu et al. Optimal partitioning for quantized EDF scheduling. In *RTSS*, 2002.
- [13] M. Ziccardi et al. EPC: extended path coverage for measurement-based probabilistic timing analysis. In *RTSS*, 2015.