On the Meaning of pWCET Distributions and their use in Schedulability Analysis

Robert I. Davis
University of York, UK
and Inria, France
Email: rob.davis@york.ac.uk

Alan Burns
University of York, UK
Email: alan.burns@york.ac.uk

David Griffin
University of York, UK
Email: david.griffin@york.ac.uk

Abstract

This short paper accompanies the keynote talk at RTSOPS 2017. It discusses the different meanings attached to probabilistic Worst-Case Execution Time (pWCET) distributions derived from Static Probabilistic Timing Analysis (SPTA) and Measurement-Based Probabilistic Timing Analysis (MBPTA). These different meanings relate to aleatoric variability (randomness in the systems and its environment) and epistemic uncertainty (lack of knowledge about the system) respectively. The different meanings have significant implications in terms of the valid use of pWCET distributions in probabilistic schedulability analysis.

I. Introduction

Verifying the timing correctness of real-time systems is typically a two step process.

- *Timing Analysis* is used to characterise the maximum amount of time which each software component or task can take to execute on the hardware platform. Typically, this is done by estimating, as a single value, an upper bound on the *Worst-Case Execution Time* (WCET).
- *Schedulability Analysis* is used to characterise the end-to-end response time of functionality involving one or more components, taking into account the way in which the components are scheduled, and any interference between them. Schedulability analysis typically makes use of WCETs to compute an upper bound on the Worst-Case Response Time (WCRT) which can then be compared to the deadline to determine timing correctness.

During the past decade the hardware platforms used, or proposed for use, in real-time embedded systems have become increasingly more complex. Architectures include advanced hardware acceleration features such as pipelines, branch prediction, out-of-order execution, caches, scratchpads, and multiple levels of memory hierarchy, as well as multi-core and many-core processors. These advances, along with increasing software complexity, greatly exacerbate the timing analysis problem. Most acceleration features are designed to optimise average-case rather than worst-case behaviour and can result in significant variability in execution times. This is making it increasingly difficult, if not impossible, to obtain tight WCET estimates from conventional static timing analysis methods that seek to provide a single absolute upper bound on the WCET. Further, increases in software and hardware complexity make it extremely difficult to ensure that the worst-case path(s) through a program have been exercised, and that the worst-case software and hardware states have been encountered in measurement-based analyses.

Probabilistic WCET analysis provides an alternative approach reflecting the fact that the absolute WCET of some software components running on advanced hardware platforms cannot be precisely determined. Instead of assuming a single absolute value for the WCET, probabilistic WCET analysis characterises the worst-case execution time of a component using a probability distribution. The precise meaning of this probabilistic Worst-Case Execution Time (pWCET) distribution is discussed in the next section. A probabilistic description of the worst-case execution time behaviour of a component can be used to estimate the probability that timing overruns may occur, and to size execution time budgets appropriately. The pWCET distribution can also be used via probabilistic schedulability analysis to upper bound the probability that deadlines may be missed.

Research into probabilistic WCET analysis can be classified into two main categories:

- *Analytical methods*: referred to as *Static Probabilistic Timing Analysis (SPTA)* [4], [7], [2], [1], [12]. SPTA is applicable when some part of the system or its environment contributes random or probabilistic timing behaviour, for example a random replacement cache. SPTA methods analyse the software, at both a high level (structural) and a low level (instructions), and use a model of the hardware behaviour to derive an estimate of worst-case timing behaviour, represented by a pWCET distribution, that is valid for any possible inputs, software states, hardware states[1], and paths through the code. SPTA does not execute the code on the actual hardware; rather it relies heavily on the model of the hardware being correct.
- *Statistical methods*: referred to as *Measurement-Based Probabilistic Timing Analysis (MBPTA)* [3], [10], [11], [6], [18], [17], [16], [14], [13]. MBPTA makes use of measurements (*observations*) of the overall execution time of the software

[1]Note any random variable, for example a random number generator within a random replacement cache, that gives rise to variation in timing behaviour is not included in these hardware states.

component, obtained by running it on the actual hardware, using test vectors i.e. inputs that exercise a relevant subset of the possible paths through the code, as well as the different software and hardware states that affect timing behaviour[2]. Rather than taking the maximum observed execution time and then adding some engineering margin, these methods use a statistical analysis of the observations based on *Extreme Value Theory* (EVT) to estimate the pWCET distribution.

II. Uncertainty and pWCET distributions

It is important to understand the precise meaning of a pWCET distribution since this impacts how such information can be used. In fact there are two subtly different meanings originating from SPTA and MBPTA, which is a potential source of confusion. We consider systems as having a *functional* behaviour i.e. what the system does in response to its inputs, and a *timing* behaviour i.e. how long it takes to respond to those inputs. Systems may have functional behaviour which is *deterministic*, in other words, given the exact same inputs, they will produce the exact same outputs. Functional behaviour may also be *non-deterministic*, for example a randomised search. Here we are concerned only with the timing analysis of software that has deterministic functionality. Timing behaviour may also be characterised as deterministic or it may depend on some element that can be characterised by a random variable, for example a random replacement cache. Hardware that supports deterministic timing behaviour is referred to as *time-predictable*, whereas hardware that deliberately introduces some random elements into the timing behaviour is referred to as *time-randomised*.

In general, uncertainty about the timing behaviour of a system can be classified into two categories:

- *Aleatoric variability* depends on chance or random behaviour within the system itself or its environment.
- *Epistemic uncertainty* is due to things that could in principle be known about the system or its environment, but in practice are not, because the information is hidden or cannot be measured or modelled.

While complex software running on advanced time-predictable hardware may in theory exhibit deterministic timing behaviour and therefore have a single absolute WCET associated with a particular set of inputs, software state, and hardware state, in practice this actual WCET cannot be determined and must therefore be estimated. Such an estimate is subject to epistemic uncertainty. In contrast, software running on simple time-randomised hardware exhibits aleatoric variability in its execution time. SPTA can be used to model aleatoric variability, but must deal with any epistemic uncertainty by upper bounding its effects in the model used. (For example if an instruction has a variable latency dependent on its operands, whose values are unknown, then SPTA can model that instruction as always assuming its worst-case latency). MBPTA can be used with systems that are characterised by either or both aleatoric variability and epistemic uncertainty.

As an example, it is instructive to consider a thought experiment involving two hypothetical systems. Both systems have 10 inputs which can take values in the range 1-6.

- **System A**: has two paths through the code. The first path is taken if the sum of the input values is odd, and takes 40 cycles to execute. The second path is taken if the sum of the input values is even, it has 10 instructions, each of which takes a random amount of time from 1-6 cycles to execute (independent of any other instruction or input value). Thus the overall execution time of this path resembles the total from rolling 10 fair dice.
- **System B**: has a single path, it uses a huge internal 10-dimensional array (with 6^{10} entries) that maps from the values of the 10 inputs to a delay. The values for the delays are the totals for each possible permutation of 10 dice rolls; however, they are randomly arranged in the array, and we do not necessarily know what that arrangement is. Further, half of the values have been set to 40 cycles; again, we do not necessarily know which ones. This system looks up its execution time from the table, using the input values, and executes in total for that amount of time.

Intrinsically, System A has only aleatoric uncertainty, while System B has only epistemic uncertainty.

Consider applying SPTA to System A. With an accurate model of the instruction timing behaviour, SPTA could be used to compute a pWCET distribution that upper bounds the timing behaviour of this system *irrespective* of its inputs.

In the context of SPTA, the meaning of a pWCET distribution can be defined as follows, building on the definition in [7]:

Definition 1: The pWCET distribution from SPTA is a tight upper bound[3] on all of the probabilistic execution time (pET) distributions that could be obtained for each individual combination of inputs, software states, and hardware states, excluding the random variables which give rise to variation in the timing behaviour. (Note, each individual pET distribution depends on the random variables, but not on the inputs or states, which are fixed in a particular combination).

Figure 1 illustrates an example pWCET distribution, showing WCET estimates $C1$ and $C2$ with different probabilities of exceedance. Also shown are the pET distributions for a few fixed combinations of inputs, software states, and hardware states. (Note this figure is only an example, it is not intended to resemble the pWCET for either System A or B).

[2]Exercising all possible paths and states is typically intractable.

[3]In the sense of the greater than or equal to operator defined on the 1 - CDF of the distributions [9].

In the absence of any random variables contributing to probabilistic timing behaviour, then the above definition of a pWCET distribution reduces to the familiar one for a single valued WCET obtained via conventional static WCET analysis. It is a tight upper bound on all the execution times that may be obtained for different combinations of inputs, software states, and hardware states.

If the random variables contributing to a probabilistic execution time behaviour are independent, then it follows that the pWCET distribution obtained by SPTA is independent with respect to any particular execution of that component. This is the case, since the pWCET distribution from SPTA upper bounds every pET distribution valid for a specific combination of inputs, software states, and hardware states. This has implications for the use of pWCET distributions, since they are independent they may be composed using basic convolution to derive probabilistic Worst-Case Response Time (pWCRT) distributions [8], [15], which can then be compared to the appropriate deadline to determine the probability of a deadline miss.

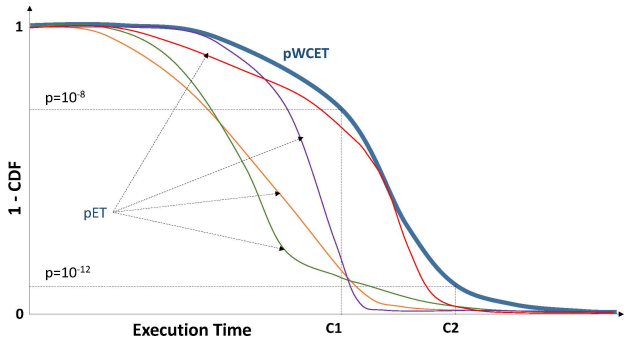

Fig. 1: Exceedance function (1-CDF) of pWCET distribution, and pET distributions for specific inputs.

Next, consider System B. Applying SPTA using a precise and detailed model of the software and hardware would result in a single WCET, since there are no random variables involved, and we assume no information about the frequency of any combination of input values. By contrast, if we apply MBPTA, then we can estimate the WCET; however, this estimate has *epistemic* uncertainty. There are things we do not know about the system when we consider it as a "black box", and we have only taken a sample of execution time observations, hence we cannot be 100% confident that our estimate is correct.

In the context of MBPTA, the meaning of a pWCET distribution can be defined as follows:

Definition 2: The pWCET distribution from MBPTA is a statistical estimate giving an upper bound p on the probability that the execution time of a component will be greater than some arbitrary value x, valid for any possible distribution of input values that could occur during deployment.

Thus the pWCET distribution characterises the probability $(1 - p)$ that the WCET of a component will be no greater than some arbitrary value x [5], or as noted by Edgar and Burns [10] the pWCET distribution reflects the *confidence* we have that the statement, *"the WCET does not exceed x for some threshold x"* is true.

We note that the definitions of a pWCET distribution originating from MBPTA and by SPTA are different. The definition from SPTA reflects aleatoric variability, while that from MBPTA reflects epistemic uncertainty. (Note there could also be an element of aleatoric variability from the system itself, for example if the hardware platform is time-randomised).

Since the pWCET definition from MBPTA reflects epistemic uncertainty, i.e. what isn't known about the system, then if it turns out that a WCET estimate x is exceeded, it is possible that it could be exceeded for *every* one of a number of runs of the component in a sequence, depending on the input values used. This is the case since the pWCET distribution effectively gives the probability that *at least one* run of the component has an execution time which exceeds x, but given that event, it provides no additional information about the execution times of individual runs.

For example, for System B, let us assume that MPBTA [6] estimates that there is a probability of 10^{-y} that the WCET exceeds x However, if that WCET estimate is exceeded, then it could be that it is exceeded *every* time the component runs, depending on the particular input values used. This has implications for how the pWCET distribution may be used in probabilistic schedulability analysis. Assuming a pWCET distribution derived via MBPTA where a WCET of x has an exceedance probability of 10^{-y}. We may only infer that N runs of the component have a probability of no more than 10^{-y} of exceeding a total execution time of Nx. Contrast this with a similar pWCET distribution derived via SPTA. In this case, assuming the aleatoric variability was due to independent random variables, then it would be valid to apply basic convolution to upper bound the overall execution time of N runs. This conclusion would not in general be sound with a pWCET distribution derived via MBPTA, due to its different meaning.

In the case of System A, the pWCET distribution from SPTA tells us that the probability that the execution time on any single run will exceed x is 10^{-y}. If we observe a value larger than x at some point in a large number of runs, then that is not in itself incompatible with the information that we have, which characterises aleatoric variability. By contrast, in the case of system B, the pWCET distribution from MBPTA gives us a *measure of confidence* that the WCET is no more than x. If we observe a value larger than x then that confidence falls to zero.

Another way of looking at the information provided by MBPTA, is to consider that among a universe of systems similar to system B that could produce the observations seen during analysis, then the probability that we are observing a system that has a WCET of more than x is estimated at 10^{-y}. Stated otherwise, among this universe of similar systems, the frequency of occurrence of a system with an actual WCET exceeding x is estimated at 1 in 10^{y}.

III. Conclusions

In this short paper, we have discussed the different meaning attached to pWCET distributions derived from Static Probabilistic Timing Analysis (SPTA) and Measurement-Based Probabilistic Timing Analysis (MBPTA) and their relation to aleatoric variability and epistemic uncertainty. These differences are crucial to the valid composition of these distributions in probabilistic schedulability analysis [15].

We conclude with a discussion of the interesting trade-off between epistemic uncertainty and aleatoric variability. Systems that exhibit *needle-in-a-haystack* problems, of rare very large execution times in the form of isolated outliers, present a serious challenge to any form of measurement-based timing analysis. Replacing the advanced hardware elements that lead to such isolated outliers with elements that exhibit randomised timing behaviour (for example random replacement caches) may smooth the timing behaviour making it more predictable (statistically), providing better management of epistemic uncertainty. Thus epistemic uncertainty may be reduced at a cost of increasing aleatoric variability. Since MBPTA can be applied to systems that exhibit both epistemic uncertainty and aleatoric variability, a number of interesting questions arise:

- Is there a benefit in trading epistemic uncertainty for aleatoric variability if the former cannot be completely eliminated?
- Is there any way to distinguish aleatoric variability from epistemic uncertainty in the results from MBPTA?
- How can we obtain a valid composition of pWCET distributions from different runs of different components?
- Is it beneficial to directly add aleatoric variability (random noise) to observations of a system that has epistemic uncertainty?
- Is a Static Probabilistic Timing Analysis necessary to show that there is no remaining epistemic uncertainty?
- How can we resolve the problem of representativity in MBPTA?

Acknowledgments

The ideas in this paper were first presented and discussed in an ad-hoc working group comprising Liliana Cucu-Grosjean, Adriana Gogonel, Iain Bate, Philipa Conway, Zoe Stephenson, Alan Burns and Robert Davis at the Dagstuhl Seminar on *Mixed Criticality on Multicore / Manycore Platforms* http://www.dagstuhl.de/17131. This work was funded in part by the Inria International Chair program and the ESPRC grant MCCps (EP/P003664/1). EPSRC Research Data Management: No new primary data was created during this study.

References

[1] S. Altmeyer, L. Cucu-Grosjean, and R. I. Davis. Static probabilistic timing analysis for real-time systems using random replacement caches. *Springer Real-Time Systems*, 51(1):77–123, 2015.

[2] S. Altmeyer and R. I. Davis. On the correctness, optimality and precision of static probabilistic timing analysis. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, pages 26:1–26:6, 2014.

[3] A. Burns and S. Edgar. Predicting computation time for advanced processor architectures. In *Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS)*, pages 89–96, 2000.

[4] F. J. Cazorla, E. Quiñones, T. Vardanega, L. Cucu, B. Triquet, G. Bernat, E. Berger, J. Abella, F. Wartel, M. Houston, L. Santinelli, L. Kosmidis, C. Lo, and D. Maxim. Proartis: Probabilistically analyzable real-time systems. *ACM Transactions on Embedded Computing Systems*, 12(2s):94:1–94:26, May 2013.

[5] L. Cucu-Grosjean. Independence a misunderstood property of and for probabilistic real-time systems. *In Real-Time Systems: the past, the present and the future*, pages 29–37, 2013.

[6] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzetti, E. Quiones, and F. J. Cazorla. Measurement-based probabilistic timing analysis for multi-path programs. In *Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS)*, pages 91–101, July 2012.

[7] R. I. Davis, L. Santinelli, S. Altmeyer, C. Maiza, and L. Cucu-Grosjean. Analysis of probabilistic cache related pre-emption delays. In *Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS)*, pages 168–179, July 2013.

[8] J. L. Diaz, D. F. Garcia, K. Kim, C-G. Lee, L. Lo Bello, J. M. Lopez, S. L. Min, and O. Mirabella. Stochastic analysis of periodic real-time systems. In *Proceedings of the IEEE Real-Time Systems Symposium (RTSS)*, pages 289–300, 2002.

[9] J. L. Diaz, J. M. Lopez, M. Garcia, A. M. Campos, Kanghee Kim, and L. L. Bello. Pessimism in the stochastic analysis of real-time systems: concept and applications. In *Proceedings of the IEEE Real-Time Systems Symposium (RTSS)*, pages 197–207, Dec 2004.

[10] S. Edgar and A. Burns. Statistical analysis of wcet for scheduling. In *Proceedings of the IEEE Real-Time Systems Symposium (RTSS)*, pages 215–224, Dec 2001.

[11] J. Hansen, S. A. Hissam, and G. A. Moreno. Statistical-based WCET estimation and validation . In *Proceedings of the Workshop on Worst-Case Execution Time Analysis (WCET)*, volume 252, 2009.

[12] B. Lesage, D. Griffin, S. Altmeyer, and R. I. Davis. Static probabilistic timing analysis for multi-path programs. In *Proceedings of the IEEE Real-Time Systems Symposium (RTSS)*, pages 361–372, Dec 2015.

[13] G. Lima and I. Bate. Valid application of evt in timing analysis by randomising execution time measurements. In *Proceedings of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, April 2017.

[14] G. Lima, D. Dias, and E. Barros. Extreme value theory for estimating task execution time bounds: A careful look. In *Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS)*, July 2016.

[15] D. Maxim and L. Cucu-Grosjean. Response time analysis for fixed-priority tasks with multiple probabilistic parameters. In *Proceedings of the IEEE Real-Time Systems Symposium (RTSS)*, pages 224–235, Dec 2013.

[16] L. Santinelli, F. Guet, and J. Morio. Revising measurement-based probabilistic timing analysis. In *Proceedings of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, Apr 2017.

[17] L. Santinelli, J. Morio, G. Dufour, and D. Jacquemart. On the Sustainability of the Extreme Value Theory for WCET Estimation. In *Proceedings of the Workshop on Worst-Case Execution Time Analysis (WCET)*, pages 21–30, 2014.

[18] F. Wartel, L. Kosmidis, C. Lo, B. Triquet, E. Quiones, J. Abella, A. Gogonel, A. Baldovin, E. Mezzetti, L. Cucu, T. Vardanega, and F. J. Cazorla. Measurement-based probabilistic timing analysis: Lessons from an integrated-modular avionics case study. In *Proceedings of the IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 241–248, June 2013.