Improvements to Static Probabilistic Timing Analysis for Systems with Random Cache Replacement Policies

Robert I. Davis
Real-Time Systems Research Group, Computer Science Department, University of York, UK
rob.davis@york.ac.uk,

I. Introduction

Critical real-time systems such as those deployed in space, aerospace, transport, and medical applications require guarantees that the probability of the system failing to meet its timing constraints is below an acceptable threshold (e.g. a failure rate of less than 10^{-9} per hour). Advances in hardware technology and the large gap between processor and memory speeds, bridged by the use of cache, make it difficult to provide such guarantees without significant over-provision of hardware resources. The use of deterministic cache replacement policies means that pathological worst-case behaviours need to be accounted for, even when in practice they may have a vanishingly small probability of actually occurring. Further, the quality of deterministic WCET estimates for such systems can be highly sensitive to missing information, making them overly pessimistic. Random cache replacement policies negate the effects of pathological worst-case behaviours while still achieving efficient average-case performance, hence they provide a means of increasing guaranteed performance in hard real-time systems [6].

Determining the timing behaviour of applications running on a processor with a random cache replacement policy requires probabilistic analysis of worst-case execution times. This can be achieved using Static Probabilistic Timing Analysis (SPTA) to compute an upper bound on the exceedance function (1 - CDF) for the probabilistic Worst-Case Execution Time (pWCET) of a program. An example exceedance function is given in Figure 1, taken from [3]. From the exceedance function, it is possible to read off for a specified probability, an execution time that has that probability of being exceeded on any single run. Static Probabilistic Timing Analysis (SPTA) has been developed for single processor systems assuming both evict-on-access [2], [1] and evict-on-miss random cache replacement policies [3].

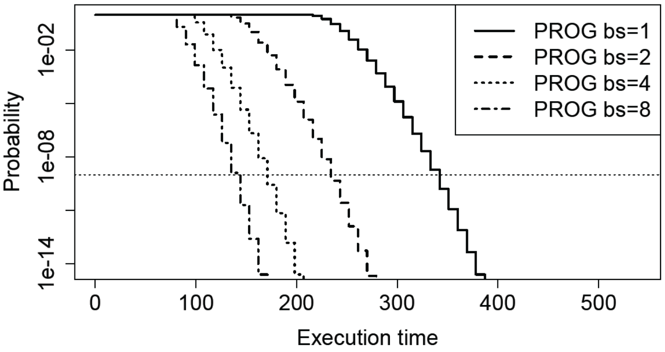

Figure 1: pWCET distributions (1-CDF) for different memory block sizes

II. Existing Static Probabilistic Timing Analysis

We now recap on SPTA for an evict-on-miss random cache replacement policy [3] for the instruction cache and no data cache. With the evict-on-miss policy, whenever an instruction is requested and is not found in the cache, then a randomly chosen cache line is evicted and the memory block containing the instruction is loaded into the evicted location. We assume an N-way associative cache, and hence the probability of any cache line being evicted on a miss is $1/N$.

For simplicity, we assume a single path program comprising a fixed sequence of instructions. We represent these instructions via the memory blocks they access; with a superscript indicating the *re-use distance k.* (The re-use distance is the maximum number of evictions since the last access to the memory block containing that instruction, and is omitted if it is infinite). For example, $a, b, a^1, c, d, b^3, c^2, d^2, a^5$. For each instruction, it has been shown [3] that the probability of a cache hit is lower bounded by:

$$P_{hit}(k) = \left(\frac{N-1}{N} \right)^k \tag{1}$$

provided that $k < N$, otherwise $P^{hit}(k) = 0$ (details of this latter restriction are given in [3]). Given fixed costs for the cache-hit latency (e.g. $H = 1$) and the cache miss latency (e.g. $M = 10$), then an upper bound pWCET distribution of a program can be computed as the convolution ($\otimes$) of the probability mass functions (PMFs) of each instruction. For example, given two instructions with PMFs with cache hit probabilities of 0.8 and 0.7 respectively, we get a pWCET distribution for the 'program' (comprising the two instructions) that has a probability of the execution time being 2 on any given run of 0.56, a probability that it will be 11 of 0.38, and a probability that there will be two cache misses and hence an execution time of 20 of 0.06.

$$\begin{pmatrix} 1 & 10 \\ 0.8 & 0.2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 10 \\ 0.7 & 0.3 \end{pmatrix} = \begin{pmatrix} 2 & 11 & 20 \\ 0.56 & 0.38 & 0.06 \end{pmatrix}$$

We note that for larger numbers of instructions, the probability of a large number of cache misses quickly becomes vanishingly small, as illustrated by the graphs of 1-CDFs or exceedance functions in [3] one of which is reproduced in Figure 1.

We note that the SPTA given for evict-on-miss [2], [1] and evict-on-access [3] policies are somewhat *pessimistic*. This pessimism arises because when computing the probability of a hit for a particular instruction, the analysis assumes that all of the intervening instructions that could potentially be misses are in fact misses, which is a pessimistic assumption. For example, for the sequence $a, b, a^1, c, d, b^3, c^2, d^2, a^5$, when computing the probability of a hit for the third occurrence of '*a*' i.e. 'a^5', it is assumed that the five intervening instructions that could potentially be misses are all misses. The corresponding memory blocks are c, d, b^3, c^2, d^2; however, the probability that b^3, c^2, d^2 are all misses is actually very small, for example if N=256, then this probability is no greater than $7.1\text{x}10^{-7}$.

III. Optimistic Static Probabilistic Timing Analysis

In an attempt to remove the source of pessimism described above, an alternative formula is given in [5], for computing the probability of a cache hit as follows:

$$P_{hit} = \left(\frac{N-1}{N}\right)^{\sum P_{miss}} \tag{2}$$

where the summation in the exponent is over the probabilities of misses of the intervening instructions. No proof is given for this formulation [5]. Below, we show that it is optimistic. We first illustrate what led us to look very carefully at this formula: the fact that it can produce irrational numbers. This was suspicious given that probabilities must necessarily be rational, as in this case, each probability is computed by counting the number of scenarios that result in a particular outcome and then dividing by the total number of possible scenarios. As an example, we assume that $N = 2$ and the summation is over just one instruction that has a probability of being a hit of 1/2 (i.e. the second 'b' in the sequence a, b, a^1, b^1), hence we have:

$P_{hit} = \left(\frac{1}{2}\right)^{1/2} = 1/\sqrt{2}$, which is irrational.

Counter example:

We now show that (2) is optimistic. For simplicity we consider the same sequence of four instructions a, b, a^1, b^1, now with a cache size $N = 4$. Further, we assume that the latency of a cache hit is 1 and the latency of a cache miss is 10. In our example, the first two instructions are certain misses, so using (1), the probability distributions for the first three instructions are as follows: $\begin{pmatrix}10\\1\end{pmatrix}, \begin{pmatrix}10\\1\end{pmatrix}, \begin{pmatrix}1 & 10\\0.75 & 0.25\end{pmatrix}$ According to (2), the probability of the 4th instruction being a hit is then $\left(\frac{3}{4}\right)^{0.25} = 0.9307$

So the overall pWCET according to [5] is $\begin{pmatrix}10\\1\end{pmatrix} \otimes \begin{pmatrix}10\\1\end{pmatrix} \otimes \begin{pmatrix}1 & 10\\0.75 & 0.25\end{pmatrix} \otimes \begin{pmatrix}1 & 10\\0.9306 & 0.0694\end{pmatrix} = \begin{pmatrix}22 & 31 & 40\\0.69795 & 0.2847 & 0.01735\end{pmatrix}$

Now let us consider the only two possible scenarios separately and compute the exact pWCET distribution.

Case 1: the second '*a*' is a cache hit. This scenario has a probability of occurrence of 0.75 (as the first '*b*' is a certain cache miss and has a probability of 0.75 of *not* evicting '*a*' from the cache). Given that the second '*a*' is a cache hit, then the second '*b*' is also certain to be a cache hit, hence the partial pWCET for this scenario is $\begin{pmatrix}10\\1\end{pmatrix} \otimes \begin{pmatrix}10\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0.75\end{pmatrix} \otimes \begin{pmatrix}1\\1\end{pmatrix} = \begin{pmatrix}22\\0.75\end{pmatrix}$

Case 2: the second '*a*' is a cache miss. This scenario has a probability of occurring of 0.25 (as the first '*b*' is a certain cache miss and has a probability of 0.25 of evicting '*a*' from the cache). Given the second '*a*' is a cache miss, the second '*b*' has a probability of 0.75 of being a cache hit, hence the partial pWCET is: $\begin{pmatrix}10\\1\end{pmatrix} \otimes \begin{pmatrix}10\\1\end{pmatrix} \otimes \begin{pmatrix}10\\0.25\end{pmatrix} \otimes \begin{pmatrix}1 & 10\\0.75 & 0.25\end{pmatrix} = \begin{pmatrix}31 & 40\\0.1875 & 0.0625\end{pmatrix}$

Combining Case 1 and Case 2, we have an overall pWCET of $\begin{pmatrix}22 & 31 & 40\\0.75 & 0.1875 & 0.0625\end{pmatrix}$

Notice that the exact pWCET distribution derived above by examining all possible scenarios is different from that obtained using the formula (2) from [5]. The precise calculation gives a higher probability of 0.0625 (versus 0.01735) of the absolute WCET of 40 occurring. Thus the formula from [5] does not deliver a valid upper bound pWCET distribution, instead it provides an optimistic pWCET that is unsafe to use. (Formally, we may say that a pWCET distribution (describing a random variable Y) is a valid upper bound on the exact pWCET distribution (describing a random variable Z) if $P\{\text{Y} \le x\} \le P\{\text{Z} \le x\}$ for any x).

We can also compute an upper bound pWCET for our example using the analysis given in [3] i.e. using (1) as follows: $\begin{pmatrix} 10 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 10 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 10 \\ 0.75 & 0.25 \end{pmatrix} \otimes \begin{pmatrix} 1 & 10 \\ 0.75 & 0.25 \end{pmatrix} = \begin{pmatrix} 22 & 31 & 40 \\ 0.5625 & 0.375 & 0.0625 \end{pmatrix}$ Notice that in this case the pWCET obtained is not itself exact, but it is a valid upper bound on the exact distribution.

IV. Open Problem

In this short paper, we have refuted the SPTA formula for the probability of a cache hit given in [5] used to compute pWCET distributions. We have also shown that previous SPTA methods do not compute an exact pWCET distribution even for the simplest of programs. Computation of an exact pWCET distribution appears to require enumeration and composition of all the different possible scenarios. While this was possible for our simple example, in general it would lead to a combinatorial (exponential) number of cases to consider, hence rendering such an approach intractable even for moderately sized examples.

The open problem that we propose is therefore how to improve upon the simple SPTA analysis [2], [1], and [3] that exists today, so as to obtain tighter bounds on the actual pWCET distribution while keeping the amount of computation required within acceptable limits.

References

[1] F.J. Cazorla, E. Quinones, 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 analysable real-time systems”, ACM TECS, 2013.

[2] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzeti, E. Quinones, and F. J. Cazorla. “Measurement-Based Probabilistic Timing Analysis for Multi-path Programs”. In Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS), 2012.

[3] 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), 2013.

[4] J. Lopez, J. L. Diaz, J. Entrialgo, D. Garca. Stochastic analysis of real-time systems under preemptive priority-driven scheduling. Real-time Systems, 40(2), 2008.

[5] L. Kosmidis, J. Abella, E. Quinones, F. J. Cazorla, “A Cache Design for Probabilistic Real-time Systems” In proceedings Design, Automation, and Test in Europe (DATE) 2013.

[6] E. Quinones, E. Berger, G. Bernat, and F. Cazorla. “Using Randomized Caches in Probabilistic Real-Time Systems”. In Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS), pages 129–138, 2009.