An Empirical Study on the Adequacy of MBPTA for Tasks Executed on a Complex Computer Architecture with Linux

Karila Palma Silva, Luís Fernando Arcaro, Daniel Bristot de Oliveira, Rômulo Silva de Oliveira
Department of Automation and Systems (DAS)
Federal University of Santa Catarina (UFSC)
Florianópolis-SC, Brazil
E-mail: { karila.palma, luis.arcaro, daniel.bristot }@posgrad.ufsc.br, romulo.deoliveira@ufsc.br

***Abstract*—In order to support the computational demand of Real-Time Systems' (RTSs') applications, the use of complex hardware will be fundamental. This fact makes analysis of software to derive Worst-Case Execution Time (WCET) an increasingly harder challenge. State-of-the-art WCET analysis techniques are hampered by the ever-growing cost and complexity of obtaining accurate knowledge of the internal operation of advanced processors, and by difficulties in reliably associating variable execution times with tasks' worst-case behaviour through measurement-based techniques. In this work, we performed an empirical assessment of the adequacy of Measurement-Based Probabilistic Timing Analysis (MBPTA) to estimate the pWCET — through Extreme Value Theory (EVT) — of a task executed on a complex computer architecture with Linux. The proposed evaluation has the objective of detecting whether EVT can be deemed suitable for determining pWCET estimates for tasks executed on complex computers.**

***Index Terms*—MBPTA, EVT, pWCET, complex architectures**

I. Introduction

As the use of computational systems proliferates in our society, systems with real-time requirements, i.e. Real-Time Systems (RTSs), become ever more common. It is necessary for the RTSs' industry to provide effective, reliable and flexible systems, making them available to the market as quickly as possible. This results in increasing software complexity, where we have more and more equipment with computers controlling and making decisions that affect the physical world and, consequently, that must use complex hardware elements (e.g., cache memories, pipelines, and branch prediction mechanisms). Moreover, such systems comprise today industrial equipment with millions of lines of code and even Linux-based embedded controllers. Consequently, providing guarantees that the temporal requirements are met becomes increasingly difficult [1], [2].

At one extreme we have the verification of critical RTSs with rigorous Worst-Case Execution Time (WCET) analysis techniques to ensure compliance with timing requirements, but with high cost and limited applicability. These systems potentially need to use simple processors, for which classical analysis techniques can be applied. At the other extreme, the practice in the RTSs industry regarding low-criticality applications is to perform measurements and add arbitrary safety margins to deal with the uncertainty of having or not observed the WCET in the tests, but this method can prove either pessimistic or unsafe [2], [3].

Measurement-Based Probabilistic Timing Analysis (MBPTA) is a modern technique that employs statistical tools, such as Extreme Value Theory (EVT), on measurements of the time taken to execute the analysed tasks on the target hardware platform. It has gained increasing attention from both the industry and the scientific community for deriving bounds for the WCET of tasks that compose RTSs. The application of EVT for performing MBPTA involves fitting an extreme value distribution to the maximum execution times observed while the analysed task is executed on the target hardware platform. The distribution is used to determine a value which is expected to be exceeded only with a sufficiently low probability — and which is therefore called a Probabilistic Worst-Case Execution Time (pWCET) estimate [3]–[7].

Standard measurements are not enough to obtain pWCETs since they may lack completeness — through the measurements, there is no guarantee to have experienced all the execution conditions. Nonetheless, measurements are important for extracting observable features such as average behaviours and trends that can appear while executing tasks. EVT makes statistical inference on the tail region of a distribution function, making it possible to determine WCET estimates associated to arbitrarily low exceedance probabilities [8].

EVT has been proposed to obtain pWCETs in relatively simple architectures, preferably randomized ones, with exceedance probability in the order of 10^{-15} targeting hard real-time systems [9]–[11]. However, there are many applications that must run on complex computer architectures using Operating Systems (OSs) such as Linux which are subject to firm real-time requirements. Tasks executed on modern processors with Linux suffer interference from other activities executing on the same system (e.g., OS asynchronous activities that affect cache contents and pipeline progression), hence generating "noise" on the observed execution times, but also potentially making them closer of being analysable through statistical

tools. In this context EVT would be a highly valuable tool for system developers, since it could enable pWCET estimates that cover cases whose witnessing is far beyond the feasible in practice through standard testing, e.g. due to limitations in the number of tests that can be performed.

In this work, we take advantage of the possibility of collecting large samples of execution times (i.e., of size 10^6) for assessing the adequacy of EVT to derive pWCET estimates for a task executed on a complex computer architecture running a Linux operating system.

This paper is structured as follows. Section II provides background on MBPTA through EVT for estimating pWCETs, and on tasks' execution time interference observable in modern processors. Section III presents a brief review of related works. The environment and conditions of the experiment are described in Section IV. Section V presents the experiment's objectives. The results are presented in Section VI, and the conclusions are summarized in Section VII.

II. Background

A. MBPTA through EVT for pWCET estimation

Due to the interest in using complex computer architectures in real-time systems, there is a quest for new methods of WCET analysis. In this context, there is considerable interest in statistical tools such as EVT. The application of EVT for performing MBPTA involves fitting an adequate extreme value distribution — such as Gumbel or Generalized Extreme Value (GEV) — to the maximum execution times observed while the analysed task executes on the target hardware platform, and determining through it a value which is expected to be exceeded only with a maximum probability that can be set to an arbitrarily low value — the pWCET [3], [6], [11]–[13].

EVT application is subject to a set of requirements that must be met for supporting the results' reliability. Among these requirements are e.g. that (1) the analysed execution times should be deemed independent and identically distributed (i.e., i.i.d.) and (2) the maximum observed values should effectively adhere to the extreme value distributions used by EVT. We use statistical hypothesis tests and quantile/probability plots to show these requirements are met [14].

EVT is typically applied by [14]:

1) Collecting a representative sample of a variable of interest associated to the phenomenon to be analysed — it is necessary to determine the sample size to be collected, such that it is large enough to properly characterize the analysed phenomenon but is still as small as possible for minimizing analysis costs.
2) Selecting the maximum observed values to be employed in the rest of the analysis process — the maximum values to be analysed are selected through either (A) the Block Maxima (BM) approach, in which the collected sample is divided into fixed-size blocks and only the blocks' highest values are kept, or (B) the Peaks Over Threshold (POT) approach, in which only the observed values that exceed a properly defined high threshold are retained [15].
3) Fitting an adequate extreme value model to the selected maxima. Assuming the BM approach is employed, one of the Weibull, the Gumbel or the Fréchet probability distributions (or their generalization GEV) is fitted to the selected maxima [13]. On the other hand, if the POT approach is employed then it is one distribution of the Generalized Pareto (GP) family that must be used [16].
4) Evidencing that the collected data is in fact analysable through EVT and that the fitted model properly represents its maxima behaviour — statistical tests and plots are employed for evidencing that the data is adequate to analysis through EVT and that the fitted distribution properly models its maxima behaviour [7].
5) Determining through the model the value expected to be exceeded only with a sufficiently low probability.

In this work, we apply EVT using the *R* statistical software [17] associated to the *extRemes* [18] package, which provides a well-documented suite of extreme value modelling tools together with a set of diagnostic resources.

The conditions in which MBPTA execution times are collected must be either representative or pessimistic in relation to those expected in the environment in which the system will operate. In this work, we fix the input data and configure the hardware state (e.g., enable hyper-threading) such that high execution times are induced, and evaluate the adequacy of EVT for timing analysis on a complex computer architecture running a Linux OS. We highlight, however, that (A) the determination of tasks' input data must be performed in a per-scenario basis and is still considered an open problem within MBPTA [19], and (B) these conditions may not lead the worst possible execution times being yielded (e.g., due to cache effects) [20].

We employ the BM approach for selecting the values analysed through EVT, and consequently one of the Weibull, Gumbel, Fréchet or GEV models are used for producing pWCETs. The GEV distribution is capable of modelling different BM right tails depending on the values assigned to its location (μ), scale (σ), and shape (ξ) parameters. Negative shape values lead GEV to represent light tails (i.e., Weibull), while positive ones lead it to model heavy asymptotic tails (i.e., Fréchet). The shape $\xi = 0$ leads it to model unbounded tails whose density decrease exponentially (i.e., Gumbel) [21], [22]. Confidence intervals on the GEV shape parameter also enable expressing uncertainty regarding which among these three models better represents the analysed data maxima.

In line with [13], we use the GEV model to evaluate the maxima shape negativity through its confidence interval, and then use the Gumbel model for effectively obtaining pWCET estimates with increased reliability.

B. Complex Computer Architectures with Linux

Multiple runs of a task can produce different times when executed on the same hardware platform. Execution times are affected by the latencies of the internal elements of the processor that perform the instructions' execution. The introduction of acceleration elements in modern processors (e.g.,

pipeline, cache, branch prediction) decreases execution times, but also makes them variable and dependent on execution history — i.e., execution history has to be taken into account for predicting the processor state at the start of the execution of a particular instruction [2].

Hardware acceleration elements make tasks' WCET analysis more difficult. *Cache memories* may allow the slower path to run faster. Usually, as the number of cache misses increases so does the WCET [2]. Some authors advocate the use of the null initial state [23], but it does not necessarily lead to the worst case. Others suggest randomizing the cache initial state during measurements [24], however, this does not guarantee that the worst case will actually be observed since it can be very rare. The *pipeline* introduces three types of complexity (1) instruction level parallelism, (2) resource sharing and allocation, and (3) dynamic scheduling, where instructions can even be executed out of order. The *branch prediction* mechanism of the Central Processing Unit (CPU) recognizes branch patterns for improving the pipeline fetching behaviour. Similarly to cache memories, there is an aspect of global history to decide which instructions' information should be stored. Moreover, the prediction decision logic can also be complex.

When executed on Linux, there are OS activities that interrupt an application's execution flow. There are asynchronous activities that interrupt the task, which include e.g., process scheduling, kernel events, OS services, and also maintenance tasks that generate indirect interference by changing the contents of cache, Translation Lookaside Buffer (TLB), etc. [25].

The use of complex architectures associated to OSs such as Linux will potentially become fundamental for supporting the computational demand of RTSs' applications. Timing analysis of tasks executed in such scenarios is therefore necessary, but made difficult due to timing effects that arise from both the hardware and the OS.

III. Related works

In [15] there is a discussion about EVT sustainability for tackling the WCET problem. They used real traces taken from an Intel(R) Xeon(R) E5620 2.4 GHz dual socket, and applied the SchedMcore runtime support and Linux Trace Toolkit new generation (LTTNG) tracing framework to guarantee real-time execution and extract execution time measurements.

In [26] the authors estimate WCET using the MBPTA technique for tasks executed on an Odroid-XU4 hardware platform, which is equipped with an ARM Big. Little Exynos 5422 chipset. The processor hosts two heterogeneous ARM A15 and A7 clusters of 4 cores each, and the board was running LUbuntu 12.04 with the 3.10.y Linux kernel.

The recent work [27] evaluates the application of EVT for tasks executed on the Atmel SamaA5D4 board running the vanilla kernel version 4.4.11. This board uses a processor equipped with one ARM Cortex-A5 600 MHz core that belongs to the ARMv7-A architecture generation.

Our work mainly differs from [15], [26] and [27] by (1) evaluating the application of MBPTA in highly complex computer architectures, (2) considering measurements taken from Linux distributions not specifically tailored to RTSs, and (3) performing evaluations based on relatively large validation samples. To our knowledge, there is no previous work that evaluated MBPTA adequacy on such scenarios.

IV. Experiment environment

In the following sections, we present the environment used and how we proceeded with the collection of the analysed data.

A. Platform

The tests were performed on a microcomputer with an Intel(R) Core(TM) i7-4770 running the Ubuntu 16.04 operating system with the vanilla kernel version 4.13.0. Ubuntu is a computer operating system based on the Debian Linux distribution that is distributed as free and open source software. The hardware is not tuned for deterministic execution. For instance, the processor uses so-called hyper-threading, which means that the instruction scheduler, which issues instructions out of order, intermingles instructions from several threads to create a mix that improves average execution time. This makes the execution time of one specific piece of code non-deterministic, because it depends on the other (hyper-threading) tasks running at the same time. These conditions were established for ensuring our evaluations are performed under a scenario in which such complex execution environments are typically employed in real computing systems.

B. Task

The software task we chose to perform the analysis is the *bsort*, part of the Mälardalen WCET Benchmarks suite [28], which sorts an array of elements using the bubble sort method. This task is frequently used in temporal analysis with input data fixed as a reverse-sorted integer array, because it is the execution path that performs the largest possible number of elementary operations.

The task was implemented as a C function of a thread that repeats the sorting operation each 10 milliseconds, i.e., we define the number of executions of the task (sample size), generate the input data, and collect one measurement every 10 milliseconds. We highlight that (1) the process is loaded only once into memory, (2) all the execution times are in microseconds (μs), and that (3) the sample sizes used are 10^5 for analysis through EVT and 10^6 for evaluation purposes.

C. Data collection

In this work, the execution times were measured from hardware counters using the Perf tool. Perf is a profiler tool for Linux 2.6+ based systems that abstracts away CPU hardware differences in Linux performance measurements. It is based on the *perf_events* interface, exported by recent versions of the Linux kernel. The Perf tool is an integrated collection of subcommands that enables various levels of inspection and analysis [29]. We run the *perf record* command — as presented below — defining a set of dynamic tracepoints for the task through the *perf probe* command [30]. We also observe the

tracing event for the external interrupts, Interrupt Request (IRQ) and Non-Maskable Interrupt (NMI)), for filtering the execution time of the task under analysis. The IRQ is an interrupt request sent from the hardware level to the CPU. While receiving the interrupt, the CPU switches to an interrupt context for handling the signalled event. A NMI is an interrupt type that differs from the standard interrupt mechanism by not being maskable, hence enforcing attention from the processor on interrupts [31].

sudo perf record -e probe_bsort:bsort_task -e probe_bsort:bsort_task_return -e irq:irq_handler_entry -e irq:irq_handler_exit -e nmi:nmi_handler -e irq:softirq_entry -e irq:softirq_exit -e power:cpu_frequency -e sched:sched_switch −−filter "prev_comm == bsort || next_comm == bsort" -a

Samples collected by *perf record* are saved into a binary file called, by default, *perf.data*. We use the *perf script* command to read *perf.data* and produce a trace output (see Figure 1). We parse the trace to get only the execution time of the task.

```
         swapper     0 [007] 500344.285353:             power:cpu_frequency: state=3900000 cpu_id=7
         swapper     0 [007] 500344.285358:              sched:sched_switch: prev_comm=swapper/7
prev_pid=0 prev_prio=120 prev_state=R ==> next_comm=bsort next_pid=8083 next_prio=120
          bsort  8083 [007] 500344.285367:        probe_bsort:bsort_task: (4009d1)
          bsort  8083 [007] 500344.285933: probe_bsort:bsort_task_return: (400a94)
          bsort  8083 [007] 500344.285946:              sched:sched_switch: prev_comm=bsort
prev_pid=8083 prev_prio=120 prev_state=S ==> next_comm=swapper/7 next_pid=0 next_prio=120
         swapper     0 [001] 500344.285986:          irq:irq_handler_entry: irq=29 name=i915
         swapper     0 [001] 500344.285993:           irq:irq_handler_exit: irq=29 ret=handled
           gedit  7812 [006] 500344.287763:              irq:softirq_entry: vec=1 [action=TIMER]
           gedit  7812 [006] 500344.287763:               irq:softirq_exit: vec=1 [action=TIMER]
           gedit  7812 [006] 500344.291762:             power:cpu_frequency: state=3900000 cpu_id=6
           gedit  7812 [006] 500344.291764:              irq:softirq_entry: vec=1 [action=TIMER]
           gedit  7812 [006] 500344.291764:               irq:softirq_exit: vec=1 [action=TIMER]
         swapper     0 [000] 500344.294437:             power:cpu_frequency: state=3900000 cpu_id=0
         swapper     0 [007] 500344.295336:              sched:sched_switch: prev_comm=swapper/7
prev_pid=0 prev_prio=120 prev_state=R ==> next_comm=bsort next_pid=8083 next_prio=120
          bsort  8083 [007] 500344.295342:        probe_bsort:bsort_task: (4009d1)
          bsort  8083 [007] 500344.295736: probe_bsort:bsort_task_return: (400a94)
          bsort  8083 [007] 500344.295748:             power:cpu_frequency: state=3900000 cpu_id=7
          bsort  8083 [007] 500344.295749:              sched:sched_switch: prev_comm=bsort
```

Fig. 1. Trace output

We use the Perf tool in order to deduct the direct interference from other tasks and interrupt handlers. This deduction may not be complete due to the complexity of the target system and the Perf command used. In our tests we found evidence that context switches in Linux have a large effect on complex processors, e.g., by causing indirect interference via pipeline, cache, TLB, etc. This indirect interference was not removed from the measured execution times. Notwithstanding, this is a simple method that can be widely applied, and it proved good enough for the objectives of this work, by keeping the environmental noise typical of complex computer architectures.

V. Experiment objectives

In this work, we performed an empirical assessment of the pWCET estimates obtained through EVT for a task executed on a complex computer architecture. The experiment we developed consists of verifying (1) the assumption of i.i.d. through statistical hypothesis tests, (2) that the measured execution times' block maxima, considering different block sizes, can be deemed to adhere to an extreme value model, and (3) the reliability and tightness of the results produced through the adjusted EVT model.

A. Statistical tests

We used statistical hypothesis tests for showing evidence of independence and identical distribution. Statistical hypothesis testing is based on a hypothesis to be tested which is assumed to be true unless evidence is found to refute it (H_0), and an alternative opposite hypothesis (H_1) that is accepted if and only if H_0 is rejected through proper evidence. Hypothesis tests typically produce p-values that are associated to the probability of the test statistic yielding a result as far from the expected as the one observed in the sample, which are then used to assess the possibility of a certain result being produced by the test solely by chance. These tests are subject to both false negative and false positive errors with respect to H_0, which must thus be controlled for taking decisions based on the yielded results. When only small samples can be obtained this is done through the determination of a limit α to the probability of false negatives, known as significance level, whose value is typically between 0.05 and 0.01. By comparing a test's p-value p with α, one can take the decision of rejecting H_0 if $p < \alpha$ or not rejecting it if $p \geq \alpha$, with a confidence level given by $\gamma = 1 - \alpha$.

In order to control hypothesis tests' errors, we employ large samples and split them into a set of randomly chosen segments — we use 100 segments, each of size 100. Our conclusions are drawn based on the statistical behaviour of the yielded results, taking into account that the tests we employ produce p-values which are either uniformly distributed or present tendency to high values in the range $[0, 1)$ when H_0 in fact holds [32]. The results are presented in the form of a box and whisker plot that highlights the 0%, 5%, 50%, 95% and 100% quantiles of the obtained p-values, i.e. the minimum, the median, the maximum, and the 5% and 95% quantiles. This approach provides increased confidence on the employed statistical tests' outcomes, since the probability of witnessing false results tends to become smaller as tests are replicated.

For evidencing independence of the random variables that influence an analysed phenomenon we used the Wald-Wolfowitz (WW) and Ljung-Box (LB) statistical tests. The WW or *runs* test, which tests the observations' randomness null hypothesis, classifies the observed values as (H)igher or (L)ower than the sample median and then examines the normality of the lengths of the "runs" (i.e., H or L contiguous sequences). The LB test has as null hypothesis the absence of correlations (trends) between observations, and is based on the detection of autocorrelations between a number of sequential observations in a time series (i.e., lags) [7], [15], [33], [34].

Similarly, identical distribution can be evidenced through the Kolmogorov-Smirnov (KS) and the k-sample *Anderson-Darling* (*AD*) statistical tests. The KS test has as null hypothesis that the observations follow the same distribution, and produces a p-value that is related to the distance between the empirical distributions of the assessed samples [7], [11]. The k-sample *AD* test is available in two versions that mainly differ by the empirical distribution function used one adjusts for possibly different sample sizes, and the second focuses on differences on the distributions' tail [35].

The application of these tests is expected to produce p-values which are either uniformly distributed or present tendency to high values in the range $[0, 1)$, hence indicating

that there are no strong evidence that the observed values (1) were produced by a non-random process, (2) present relevant dependency, or (3) were drawn from different distributions.

B. Block size

Related works generally use blocks of size 50, since often no substantial improvement is perceived by increasing the block size in simple processors' execution times [6]. The block size must be determined such that the analysed maxima best fits the target EVT model, as different values can lead to largely different fitness properties. In this work, we have assessed the experimental scenario with different block sizes (i.e., 50, 250, 500, 1000 and 1250) for detecting possible differences in our conclusions. We observed that, due to the nature of our phenomenon, substantial improvement in model fitness could be perceived by increasing the block size.

According to [6] the size of the blocks determines the portion of the original distribution that is considered to belong to the tail. If we use as many blocks as elements in the original distribution, the distribution of the block maximum values will gather the entirety of the original distribution, rather than capturing the essence of the tail. The larger the block size the better the fit to the tail, however, increasing the block size results in fewer blocks and thus fewer values in the final sample. In this work, we can increase the chance of observing rare events through the use of large samples (10^5) with relatively small cost. The larger the sample set is the more representative it will be, however, collecting large samples may not be feasible for requiring too much testing time. For this reason, deciding the size of the sample to be collected can also be considered a challenging issue.

C. Model fitting

We use quantile and probability plots for showing evidence that the analysed maxima adhere to EVT models. Such plots evidence good model fitting if the points are disposed over or randomly distributed around and close to the $y = x$ line, indicating that the observed values are compatible with those expected when the fitted distribution properly models them. On the other hand, if significant systematic discrepancies are evident the sample must be deemed not adherent to the model, and the analysis results cannot be considered trustworthy [14].

For fitting the GEV model to modelling samples there are three methods commonly used, namely Maximum Likelihood Estimation (MLE) [36], Generalized Maximum Likelihood Estimation (GMLE) [37], and L-moments [38]. MLE is known to have convergence problems when $\xi < -0.5$ and cannot be used if $\xi < -1$ [36], while GMLE may fail in providing parameters' estimates and/or confidence intervals under a number of conditions [13]. In this work, we focus on results produced through the L-moments method.

For fitting the Gumbel distribution to modelling samples we apply the MLE method [39], which proved robust in determining estimates both for the distribution parameters' values and for their confidence intervals. Alternatively, we could have used (1) the quantile-quantile regression approach presented in [40], or (2) the moments approach presented in [39]. Despite (1) and (2) being simple methods for fitting the Gumbel model, their simplest form does not produce confidence intervals on parameters' values as provided by MLE.

As previously mentioned and in line with [13] we also do not directly apply MBPTA using the adjusted GEV model, but we use instead GEV fitting only to evaluate maxima shape negativity and derive pWCET estimates by adjusting a Gumbel model to maxima with $\xi < 0$. As shown in [13] this approach often leads to more reliable pWCET estimates, despite the introduction of some level of pessimism.

We highlight that, due to the nature of the phenomenon we analyse, a sample of size 10^5 proved necessary for the maxima to adhere to the GEV model. Using smaller samples we have witnessed significant systematic discrepancies both in quantile and in probability plots, evidencing that MBPTA results could not be considered trustworthy in such scenarios due to lack of fitness to EVT models.

D. pWCET estimates

Once the EVT applicability requirements are satisfied (i.e., the analysed execution times are deemed independent and identically distributed) and fitting an adequate extreme value model to the selected maxima proves possible, we can determine through the model the value expected to be exceeded only with a sufficiently low probability.

The yielded pWCET estimates with, e.g., an exceedance probability of 10^{-10}, serve as an estimator for the highest value expected to be observed in 10^{10} executions of the analysed task. Since the execution of such a large number of tests is generally infeasible in practice, EVT inference techniques can be considered a valuable tool for developers of systems such as those considered in this work.

E. pWCET reliability and tightness

We use the reliability and tightness analyses presented in [13] to verify whether the obtained pWCET estimates can be considered adequate.

The reliability analysis assesses whether the pWCET estimates derived through EVT using the Gumbel model can be deemed reliable when compared to the highest values effectively observed in large samples. For that we plot the pWCET estimates with 10^{-10} exceedance probability and their associated 95% confidence intervals as the sample size is increased from 3000 (i.e., 3 maxima blocks) to 100000 (i.e., 100 blocks) in steps of 1000 (i.e., steps of 1 block and 1 maximum). The horizontal line indicates the High Water Mark (HWM) observed for the validation sample of size 10^6. For the pWCET estimates to be considered reliable, we expect them not to be exceeded by the effectively observed HWM, since the magnitude of the validation sample is much smaller than that of the pWCET exceedance probability used.

The tightness analysis assesses whether the pWCET estimates derived through the Gumbel model can be considered tight, by analysing the distance between the HWMs on a pair

of large samples (we use 10^4 and 10^6) and the pWCET estimates derived with an exceedance probability of intermediate magnitude (we use 10^{-5}). For that we plot the pWCET(10^{-5}) estimates and their associated 95% confidence intervals, as the sample size is increased from 3000 to 100000 measurements in steps of 1000 (i.e., steps of 1 block). The horizontal lines indicate the HWMs observed for samples of size 10^4 and 10^6. For the estimates to be considered tight we expect them to be either between the samples' maxima or slightly above the higher of them, evidencing that the estimates are consistently higher than smaller magnitude samples' HWMs and exceed with small pessimism maxima of larger magnitude samples.

VI. Experiment results

A. Applicability evidence

To verify the applicability of MBPTA in this work, we employ the diagnostic artefacts presented in Section V-A. Figure 2 presents a box and whisker plot of the p-values yielded by the independence and identical distribution statistical tests. From the analysis of these artefacts one can conclude that all statistical tests' results are acceptable, since the produced p-values are either approximately uniformly distributed or present tendency to high values.

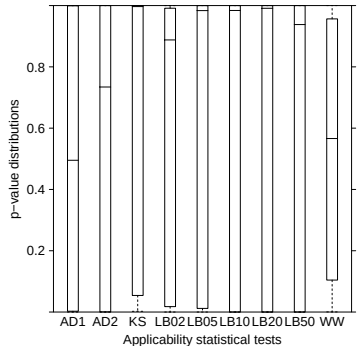


Fig. 2. Statistical tests' p-values

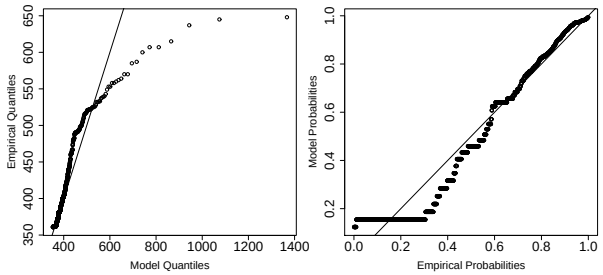
B. Model fitting

We evaluate the model fitting and the confidence interval of the shape parameter for different block sizes. The shape and its confidence interval (CI) produced through the GEV fitting process, presented in Table I, show that the sample's maxima shape tends to become negative and the fitness of the data to a GEV model is improved when the block size is increased (see Figures 3, 4, 5, 6 and 7). From this evaluation, we can conclude with 95% certainty that the maxima adhere to the Weibull distribution ($\xi < 0$) when the block size is greater than or equal to 500.

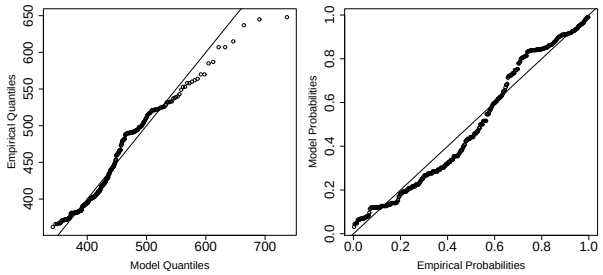
Since $\xi < 0$ and following the recommendations of [13], we obtain pWCET estimates through the Gumbel model using blocks of size 1000. The quantile and probability plots of the resulting Gumbel model are shown in Figure 8, which we consider satisfactory for the purpose of this work.

TABLE I
SAMPLE 10^5

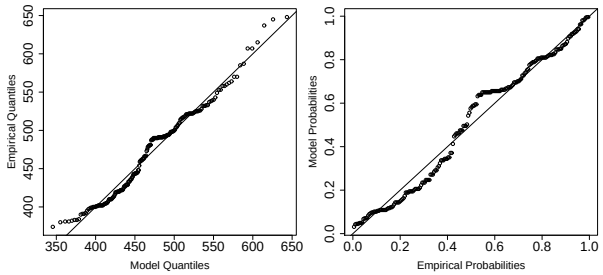
Block size	Maxima blocks	LB CI ξ	ξ	UB CI ξ
50	2000	0.4084	0.4853	0.5676
250	400	0.0052	0.0835	0.1689
500	200	-0.2592	-0.1514	-0.0557
1000	100	-0.3299	-0.1820	-0.0510
1250	80	-0.3934	-0.2194	-0.0741



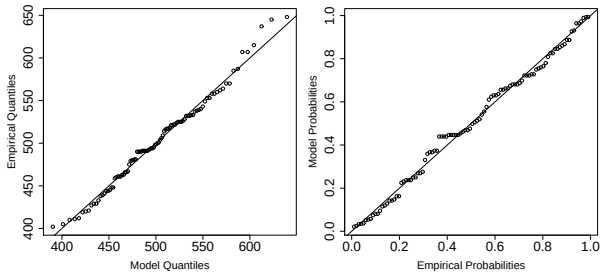
(a) Quantile plot (b) Probability plot
Fig. 3. GEV - Block size 50



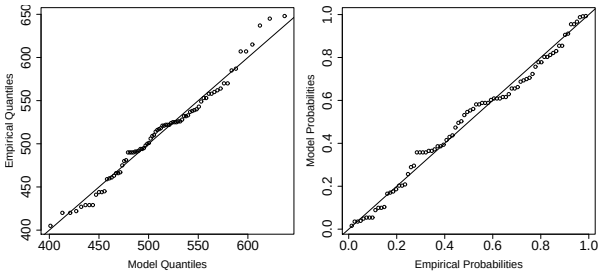
(a) Quantile plot (b) Probability plot
Fig. 4. GEV - Block size 250



(a) Quantile plot (b) Probability plot
Fig. 5. GEV - Block size 500

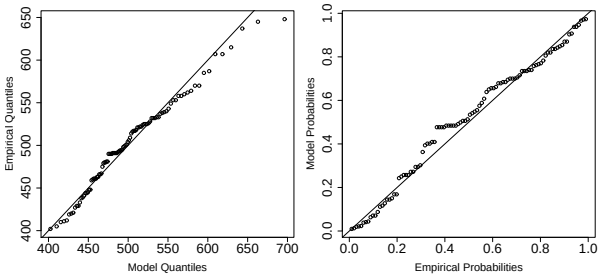


(a) Quantile plot (b) Probability plot
Fig. 6. GEV - Block size 1000



(a) Quantile plot (b) Probability plot

Fig. 7. GEV - Block size 1250



(a) Quantile plot (b) Probability plot

Fig. 8. Gumbel - Block size 1000

C. pWCET estimates

The pWCET estimates (in μs) derived with different exceedance probability through EVT using the Gumbel model — considering blocks of size 1000 — are presented in Table II. The confidence intervals (CI) of the Gumbel parameters μ and σ (location and scale, respectively) are derived using a significance level $\alpha = 0.05$, that is, they were defined at the confidence level of 95%.

TABLE II
PWCET ESTIMATES

Parameter	Estimate
μ	475.6370
σ	47.8768
LB CI μ	465.7097
UB CI μ	485.5643
LB CI σ	40.7464
UB CI σ	55.0071
pWCET	**Estimate**
pWCET(10^{-5})	1026.8386
pWCET(10^{-6})	1137.0792
pWCET(10^{-8})	1357.5600
pWCET(10^{-10})	1578.0408
pWCET(10^{-12})	1798.5226
pWCET(10^{-15})	2129.2809

D. pWCET reliability and tightness

As previously mentioned, we use the reliability and tightness analyses presented in [13] to evaluate pWCET estimates. Figure 9 presents the reliability assessment plot, which shows that all pWCET(10^{-10}) estimates and their associated confidence intervals stay consistently above the 10^6 HWM as the modelling sample size is increased, hence being considered reliable.

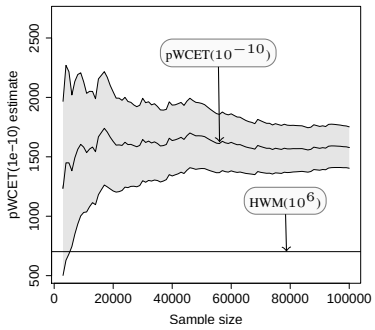


Fig. 9. pWCET reliability

Figure 10 presents the tightness assessment plot, which shows that both the pWCET estimates with exceedance probability of 10^{-5} and their associated confidence intervals remain above the upper HWM (i.e., 10^6) as sample size is increased. This suggests that the produced estimates present limited tightness, but also reinforce their reliability.

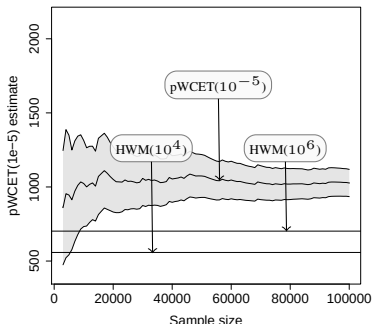


Fig. 10. pWCET tightness

VII. CONCLUSION

In this work we performed an empirical assessment of the adequacy of the pWCET estimates produced through MBPTA using EVT for a task executed on a complex computer architecture within a typical Linux system environment. For that we employed a large (i.e., of size 10^6) execution time validation sample collected using the Perf tool, and compared the produced pWCET estimates with the highest values (i.e., HWMs) effectively observed in the validation sample.

We conclude, from the assessments presented in this paper, that EVT applied to a sample of 10^5 tests seems to generate an acceptable pWCET with exceedance probability of e.g. 10^{-10}, which would normally require 10^{10} tests — what is generally not feasible in practice. The produced estimates appear to present limited tightness, but our validation experiment also give confidence on the reliability of the yielded results whenever EVT applicability conditions hold.

We highlight that our conclusions are specific to the task and hardware used, and that experiments on different scenarios, employing, e.g., simpler hardware platforms or other kernel flavours, could lead to different results. Notwithstanding, the possibility of deriving pWCET estimates for different probabilities of exceedance, regarding a task running on a complex computer architecture with Linux, is a relevant contribution to the toolset available for RTSs' developers.

ACKNOWLEDGMENT

This research was partially funded by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) and CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior).

REFERENCES

[1] J. W.-S. Liu, *Real-Time systems*, 1st ed. Prentice Hall, 2000.

[2] R. Wilhelm, T. Mitra, F. Mueller, I. Puaut, P. Puschner, J. Staschulat, P. Stenström, J. Engblom, A. Ermedahl, N. Holsti *et al.*, "The Worst-Case Execution-Time Problem - Overview of Methods and Survey of Tools," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 7, pp. 36:1–36:53, 2008.

[3] F. J. Cazorla, J. Abella, J. Andersson, T. Vardanega, F. Vatrinet, I. Bate, I. Broster, M. Azkarate-askasua, F. Wartel, L. Cucu-Grosjean *et al.*, "PROXIMA: Improving Measurement-Based Timing Analysis through Randomisation and Probabilistic Analysis," in *Euromicro Conference on Digital System Design 2016 (DSD'16)*. IEEE, 2016, pp. 276–285.

[4] R. I. Davis, T. Vardanega, J. Andersson, F. Vatrinet, M. Pearce, I. Broster, M. Azkarate-Askasua, F. Wartel, L. Cucu-Grosjean, M. Patte *et al.*, "PROXIMA: A Probabilistic Approach to the Timing Behaviour of Mixed-Criticality Systems," *Ada User Journal (AUJ)*, pp. 118–122, 2014.

[5] J. Abella, D. Hardy, I. Puaut, E. Quiñones, and F. J. Cazorla, "On the Comparison of Deterministic and Probabilistic WCET Estimation Techniques," in *Euromicro Conference on Real-Time Systems 2014 (ECRTS'14)*. IEEE, 2014, pp. 266–275.

[6] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzetti, E. Quiñones, and F. J. Cazorla, "Measurement-Based Probabilistic Timing Analysis for Multi-path Programs," in *Euromicro Conference on Real-Time Systems 2012 (ECRTS'12)*. IEEE, 2012, pp. 91–101.

[7] L. Kosmidis, E. Quiñones, J. Abella, T. Vardanega, C. Hernandez, A. Gianarro, I. Broster, and F. J. Cazorla, "Fitting Processor Architectures for Measurement-Based Probabilistic Timing Analysis," *Microprocessors and Microsystems (MICPRO)*, vol. 47B, pp. 287–302, 2016.

[8] K. Berezovskyi, L. Santinelli, K. Bletsas, and E. Tovar, "WCET Measurement-Based and Extreme Value Theory Characterisation of CUDA Kernels," in *Proceedings of the 22nd International Conference on Real-Time Networks and Systems*. ACM, 2014, p. 279.

[9] L. Kosmidis, J. Abella, E. Quiñones, and F. J. Cazorla, "A Cache Design for Probabilistically Analysable Real-time Systems," in *Design, Automation and Test in Europe Conference and Exhibition 2013 (DATE'13)*. IEEE, 2013, pp. 513–518.

[10] L. Kosmidis, E. Quiñones, J. Abella, T. Vardanega, I. Broster, and F. J. Cazorla, "Measurement-Based Probabilistic Timing Analysis and Its Impact on Processor Architecture," in *Euromicro Conference on Digital System Design 2014 (DSD'14)*. IEEE, 2014, pp. 401–410.

[11] J. Abella, E. Quiñones, F. Wartel, T. Vardanega, and F. J. Cazorla, "Heart of Gold: Making the Improbable Happen to Increase Confidence in MBPTA," in *Euromicro Conference on Real-Time Systems 2014 (ECRTS'14)*. IEEE, 2014, pp. 255–265.

[12] S. Milutinovic, E. Mezzetti, J. Abella, T. Vardanega, and F. J. Cazorla, "On Uses of Extreme Value Theory Fit for Industrial-Quality WCET Analysis," in *International Symposium on Industrial Embedded Systems 2017 (SIES'17)*. IEEE, 2017, pp. 1–6.

[13] K. P. Silva, L. F. Arcaro, and R. S. de Oliveira, "On Using GEV or Gumbel Models when Applying EVT for Probabilistic WCET Estimation," in *2017 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, Dec 2017.

[14] S. G. Coles, *An Introduction to Statistical Modeling of Extreme Values*, 1st ed., ser. Springer Series in Statistics. Springer, 2001.

[15] L. Santinelli, J. Morio, G. Dufour, and D. Jacquemart, "On the Sustainability of the Extreme Value Theory for WCET Estimation," in *International Workshop on Worst-Case Execution Time Analysis 2014 (WCET'14)*, vol. 39. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014.

[16] L. F. Arcaro, K. P. Silva, and R. S. d. Oliveira, "On the Reliability and Tightness of GP and Exponential Models for Probabilistic WCET Estimation," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 23, no. 3, pp. 39:1–39:27, 2018.

[17] R, "R: A Language and Environment for Statistical Computing," 2017. [Online]. Available: http://www.r-project.org/

[18] E. Gilleland and R. W. Katz, "extRemes 2.0: An Extreme Value Analysis Package in R," *Journal of Statistical Software*, vol. 72, p. 39, 2016.

[19] S. Law and I. Bate, "Achieving Appropriate Test Coverage for Reliable Measurement-Based Timing Analysis," in *Euromicro Conference on Real-Time Systems 2016 (ECRTS'16)*. IEEE, 2016, pp. 189–199.

[20] J. Abella, C. Hernandez, E. Quiñones, F. J. Cazorla, P. R. Conmy, M. Azkarate-askasua, J. Perez, E. Mezzetti, and T. Vardanega, "WCET Analysis Methods: Pitfalls and Challenges on their Trustworthiness," in *International Symposium on Industrial Embedded Systems 2015 (SIES'15)*. IEEE, 2015, pp. 1–10.

[21] G. Lima, D. Dias, and E. Barros, "Extreme Value Theory for Estimating Task Execution Time Bounds: A Careful Look," in *Euromicro Conference on Real-Time Systems 2016 (ECRTS'16)*. IEEE, 2016.

[22] J. Abella, M. Padilla, J. D. Castillo, and F. J. Cazorla, "Measurement-Based Worst-Case Execution Time Estimation Using the Coefficient of Variation," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 22, pp. 72:1–72:29, 2017.

[23] D. Griffin and A. Burns, "Realism in Statistical Analysis of Worst Case Execution Times," in *International Workshop on Worst-Case Execution Time Analysis 2010 (WCET'10)*, vol. 15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010, pp. 44–53.

[24] S. M. Petters, "Worst Case Execution Time Estimation for Advanced Processor Architectures," Ph.D. dissertation, Technische Universität München, 2002.

[25] D. B. Oliveira and R. S. Oliveira, "Timing Analysis of the PREEMPT RT Linux Kernel," *Software: Practice and Experience*, vol. 46, no. 6, pp. 789–819, Jun. 2016.

[26] A. Suyyagh and Z. Zilic, "Real-Time Benchmark set Synthesis Based on pWCET Estimation and Bounded Hyper-Periods," in *International Conference on Circuits System and Simulation (ICCSS)*. IEEE, 2017.

[27] I. Fedotova, B. Krause, and E. Siemens, "Applicability of Extreme Value Theory to the Execution Time Prediction of Programs on SoCs," in *Proceedings of the 5th International Conference on Applied Innovations in IT, (ICAIIT)*, 2017.

[28] J. Gustafsson, A. Betts, A. Ermedahl, and B. Lisper, "The Mälardalen WCET Benchmarks: Past, Present And Future," in *International Workshop on Worst-Case Execution Time Analysis 2010 (WCET'10)*, vol. 15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010.

[29] Perf, "Linux Perf tool," 2015. [Online]. Available: https://perf.wiki.kernel.org/

[30] D. B. Oliveira and R. S. Oliveira, "Comparative Analysis of Trace Tools for Real-Time Linux," *IEEE Latin America Transactions*, vol. 12, no. 6, pp. 1134–1140, Sept 2014.

[31] ——, "Automata-based modeling of interrupts in the Linux PREEMPT RT kernel," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sept 2017, pp. 1–8.

[32] G. Marsaglia and W. W. Tsang, "Some Difficult-to-pass Tests of Randomness," *Journal of Statistical Software*, vol. 7, pp. 1–9, 2002.

[33] G. M. Ljung and G. E. P. Box, "On a Measure of Lack of Fit in Time Series Models," *Biometrika*, vol. 65, pp. 297–303, 1978.

[34] M. Slijepcevic, L. Kosmidis, J. Abella, E. Quiñones, and F. J. Cazorla, "Time-Analysable Non-Partitioned Shared Caches for Real-Time Multicore Systems," in *Design Automation Conference 2014 (DAC'14)*. ACM, 2014, pp. 198:1–198:6.

[35] F. W. Scholz and M. A. Stephens, "K-Sample Anderson-Darling Tests," *Journal of the American Statistical Association*, vol. 82, 1987.

[36] R. L. Smith, "Maximum Likelihood Estimation in a Class of Nonregular Cases," *Biometrika*, vol. 72, pp. 67–90, 1985.

[37] E. S. Martins and J. R. Stedinger, "Generalized Maximum-Likelihood Generalized Extreme-Value Quantile Estimators for Hydrologic Data," *Water Resources Research*, vol. 36, pp. 737–744, 2000.

[38] J. R. M. Hosking, "L-Moments: Analysis and Estimation of Distributions Using Linear Combinations of Order Statistics," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 52, 1990.

[39] S. Edgar and A. Burns, "Statistical Analysis of WCET for Scheduling," in *Real-Time Systems Symposium 2001 (RTSS'01)*. IEEE, 2001.

[40] J. Hansen, S. Hissam, and G. A. Moreno, "Statistical-Based WCET Estimation and Validation," in *International Workshop on Worst-Case Execution Time Analysis 2009 (WCET'09)*, vol. 10. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2009, pp. 1–11.