CISTER
Research Center in
Real-Time & Embedded
Computing Systems

Conference Paper

Measurement-Based Probabilistic Timing Analysis for Graphics Pro cessor Units

Kostiantyn Berezovskyi
Fabrice Guet
Luca Santinelli
Konstantinos Bletsas
and Eduardo Tovar

CISTER-TR-160302

Measurement-Based Probabilistic Timing Analysis for Graphics Pro cessor Units

Kostiantyn Berezovskyi, Fabrice Guet, Luca Santinelli, Konstantinos Bletsas, and Eduardo Tovar

CISTER Research Center

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail:

http://www.cister.isep.ipp.pt

Abstract

Purely analytical worst-case execution time (WCET) estimation approaches for Graphics Processor Units (GPUs) cannot go farbecause of insufficient public information for the hardware. Thereforemeasurement-based probabilistic timing analysis (MBPTA) seems theway forward. We recently demonstrated MBPTA for GPUs, based onExtreme Value Theory (EVT) of the “Block Maxima” paradigm. In thisnewer work, we formulate and experimentally evaluate a more robustMBPTA approach based on the EVT “Peak over Threshold” paradigmwith a complete set of tests for verifying EVT applicability. It optimallyselects parameters to best-fit the input measurements for more accurateprobabilistic WCET estimates. Different system configuration parameters (cache arrangements, thread block size) and their effect on thepWCET are considered, enhancing models of worst-case GPU behavior.

Measurement-Based Probabilistic Timing Analysis for Graphics Processor Units

Kostiantyn Berezovskyi[+], Fabrice Guet[*], Luca Santinelli[*], Konstantinos Bletsas[+], and Eduardo Tovar[+]

[*]ONERA Toulouse, France, [+]CISTER/INESC-TEC, ISEP, Portugal.

Abstract. Purely analytical worst-case execution time (WCET) estimation approaches for Graphics Processor Units (GPUs) cannot go far because of insufficient public information for the hardware. Therefore measurement-based probabilistic timing analysis (MBPTA) seems the way forward. We recently demonstrated MBPTA for GPUs, based on Extreme Value Theory (EVT) of the "Block Maxima" paradigm. In this newer work, we formulate and experimentally evaluate a more robust MBPTA approach based on the EVT "Peak over Threshold" paradigm with a complete set of tests for verifying EVT applicability. It optimally selects parameters to best-fit the input measurements for more accurate probabilistic WCET estimates. Different system configuration parameters (cache arrangements, thread block size) and their effect on the pWCET are considered, enhancing models of worst-case GPU behavior.

1 Introduction

Programming models such as CUDA (Compute Unified Device Architecture) facilitate harnessing the power of GPUs for general-purpose applications exhibiting inherent parallelism, and even for embedded real-time systems. However, such systems have timeliness constraints and currently no satisfactory worst-case execution time (WCET) analysis technique for parallel applications on GPUs exists. Techniques for CPUs are not portable because GPU applications consist of thousands of identical ultra-lightweight threads (1-cycle context-switch) and we are not interested in the execution time of any one of them; instead we want to bound the time since the first thread starts until the last one completes.

Analytical approaches, relying on detailed GPU models [3,15] have had limited success because the application has *no control* over how intra-GPU thread scheduling, which is also a trade secret; and so is the GPU cache replacement policy. Static measurement-based approaches [6] face the same challenges.

Therefore, a probabilistic measurement-based approach, relying on statistical analysis and Extreme Value Theory (EVT) seems a viable alternative, as it can characterize the WCET even without this information. Many works insist on hardware randomization (e.g., random replacement caches) as a prerequisite for the application of Measurement-Based Probabilistic Timing Analysis (MBPTA) and EVT. Randomization indeed helps with certain properties, but with commercial-of-the-shelf GPUs it is not an option – and, as we will demonstrate, it is not strictly needed either, for WCET characterization via EVT.

GPU architectures and CUDA Modern GPUs contain several "Streaming Multiprocessors" (SMs), which are complex manycores in themselves. For example, the NVIDIA Kepler GK104 [30] has 8 SMs and a shared 1.5 MB L2 cache. Each SM has 192 CUDA cores, 32 load/store units, 32 special units (e.g., for cosines in H/W) and 64 double-precision units. Its 64-kB dedicated memory, with the latency of a register, is split into "shared memory" and L1.

CUDA programs running on GPUs are called "kernels". Under CUDA, at any time, groups of 32 threads (termed *warps*) execute in lockstep, i.e., during the same cycles as each other and also executing the same kernel instruction[1]. At run-time, warps are bundled together in groups termed *thread blocks* and each thread block is sent to one SM for execution. Each SM has a few thread blocks assigned to it at any time. Thread blocks do not migrate among SMs. The CUDA engine tries to keep each SM's processing units busy, but exactly how warps are dispatched is not publicly documented.

GPU timing analysis: state of the art Despite the lack of GPU documentation, effortts are made to analyse GPUs or make them more time-predictable. Many works attempted to make the scheduling on the GPU more predictable [2,18] and provide multitasking [19] among different GPU contexts and efficient resource sharing. In [17], CPU-GPU data transfers are made preemptible, to reduce blocking. The GPU management infrastructure in [31] supports data transfer and computation overlap and multi-GPU systems. The framework in [27] supports LDF and fixed-priority policies and uses the maximum execution time over a number of runs of a task as a WCET estimate. The lock-based management framework for multi-GPU systems in [12] also allocates GPU resources to tasks via a cost predictor that infers computation times and data transfer delay from a few runs. In [25] the adaptive scheduling of anytime algorithms is explored; worst-case scenarios for GPU kernels are empirically derived experimentally.

The ILP-based WCET estimation in [3] is intractable for longer kernels (due to control variable explosion) and it relies on an *optimistic* assumption about cache misses. The metaheuristic-based alternative in [4] for soft real-time systems is more tractable but its WCET estimates are not provably safe and the optimistic assumptions about cache misses remain. Since L1 misses take hundreds of cycles, extending [3] or [4] to tractably model caches is hard.

Betts *et al.* [6] employ the simulator GPGPU-sim [1]. Their first technique (*dynamic*) estimates from the respective high-water mark times the maximum "release jitter" (delay in launch, measured from the kernel launch) and WCET (including the effects of contention for cache, GPU main memory, etc) of the GPU warps. A second technique (*hybrid*) assumes a fixed delay for launching each additional warp and uses static analysis based on instrumentation point graphs annotated with execution times obtained from the measurements. This assumes thread blocks arriving in "waves" and processed in round-robin.

Recently [5], we applied Block-Maxima EVT to CUDA kernels, and explored the dependence of probabilistic WCETs (pWCETs) on the size of the problem instance. In this work, we apply the Peak Over Threshold variant of EVT aiming at providing a more complete view to the EVT and highlighting how these

[1] Intra-warp control flow divergence is handled with predicates/masking and NOPs.

techniques can offer clues to the developer about optimizing performance wrt the pWCET. We fix the size of the problem instance, in order to explore how other factors (cache configuration, thread block size) affect the pWCET.

2 Statistical Modeling of Execution Time with GPUs

Whenever there is variability of task execution times, these may be defined as random variables (rvs)[2]. The rv $\mathcal{C}_i$ draws its values from the set of different execution times that task τ_i can experience, with respective observed probability; $\mathcal{C}_i$ is an empirical distribution obtained from actual measurements.

The Cumulative Distribution Function (CDF) $F_{\mathcal{C}_i}(C_{i,x}) \stackrel{def}{=} \sum_{j=0}^{x} f_{\mathcal{C}_i}(C_{i,j}) = P(\mathcal{C}_i \leq C_{i,x})$ and the inverse Cumulative Distribution Function (1-CDF) $F'_{\mathcal{C}_i}(C_{i,x}) \stackrel{def}{=} 1 - \sum_{j=0}^{x} f_{\mathcal{C}_i}(C_{i,j})$ are alternative representations to the pdf. In particular, the 1-CDF outlines the exceedence thresholds as $P\{\mathcal{C}_i \geq C_{i,x}\}$. Each measurement $C_{i,k}$ is an execution time sample, stored in a trace $\mathcal{T}_{C_i}$ such that $\forall k, \mathcal{T}_{C_i}(k) = C_{i,k}$. We call $\mathcal{C}_i$ (calligraphic) the Execution Time Profile (ETP). Together with the traces, it describes the average execution-time behavior of τ_i.

CUDA Measurements. In this work we focus exclusively on the net CUDA kernel execution time, denoted as C^{DEV}. This corresponds to the time since the first warp executes its first instruction until the last one completes:

$$C^{\mathrm{DEV}} = \max_p\{\text{end\_cycle}[p]\} - \min_p\{\text{start\_cycle}[p]\} \tag{1}$$

where $p = 1, 2, \ldots, P$ is the index of the SM and the start_cycle/ end_cycle variables hold the value of special clock-register on each SM, recorded by extra-lighweight instrumentation assembly code injected into the kernel[3]. We collect execution time measurements for a sufficient (see below) number of runs of a given kernel, under the same execution conditions, and apply EVT to those.

2.1 Worst-case profiling

Within a probabilistic paradigm, the pWCET is the worst possible distribution of task execution times. There should exist the exact pWCET $\mathcal{C}_i^*$ as the tightest upper bound distribution to any possible ETP in any possible execution condition or system scenario. Due to the overall complexity or cost of deriving the exact pWCET distribution, MBPTA approaches tend to infer pWCET estimates $\overline{\mathcal{C}}_i$ which are safe in the sense that they are distributions greater than or equal to the exact (and potentially unknown) pWCET[4]. The partial ordering among distributions is defined such that, a distribution $\mathcal{C}_j$ is greater than or equal to a

[2] A random variable is a variable whose value is subject to variations due to chance; it can take on a set of possible different values, each with an associated probability.

[3] Admittedly, then the execution time is that of the *modified* kernel.

[4] The same holds for deterministic approaches, which derive safe WCET estimates from incomplete system models or assumptions about the system behavior.

distribution $\mathcal{C}_k, \mathcal{C}_j \succeq \mathcal{C}_k$, iff $P(\mathcal{C}_j \leq d) \leq P(\mathcal{C}_k \leq d)$ for any d and the two random variables are not identically distributed (two different distributions), [11].

The EVT deals with the extreme deviations from the median of probability distributions. It estimates the tails of distributions, where the worst case should lie, thus pWCET estimates $\overline{\mathcal{C}}_i$. These are continuous worst-case distributions [9]

It is assumed that the safety of the worst-case estimates $\overline{\mathcal{C}}_i$ with EVT relates only to the EVT applicability hypotheses, [10]. Ongoing research is investigating more formally both the safety and the robustness of EVT worst-case estimates.

The Fisher-Tippet-Gnedenko theorem [14] presents the EVT Block Maxima (BM) formulation where the tail distribution G_ξ is the possible limit law characterizing the sequence of the maxima $B_n = max\{C_{i,1}, C_{i,2}, \ldots, C_{i,n}\}$ of n independent identically distributed (i.i.d.) measurements $\{C_{i,n}\}$ as $n \to \infty$. In other words, the theorem says that whenever $\mathcal{C}_i$ belongs to the Maximum Domain of Attraction (MDA), $\mathcal{C}_i \in MDA(\mathcal{G}_\xi)$, then $\mathcal{G}_\xi$ is a good approximation of the extreme task behavior. $\mathcal{G}_\xi$ is the Generalized Extreme Value (GEV) distribution which is a family of continuous probability distributions combining the Gumbel, Frechet and Weibull families. The parameter ξ defines the shape of the GEV, such that $\xi = 0$, $\xi > 0$ and $\xi < 0$ correspond respectively to the Gumbel, Frechet and Weibull. The block size *block* plays a central role for the resulting pWCET estimation. In previous works, the pWCET estimates are achieved with the EVT BM approach applying Gumbel distributions, [9].

The second approach to the EVT is the Peaks Over Threshold (POT). It models the law of the execution time peaks in a trace that exceed a threshold.

Definition 1 (Generalized Pareto Distribution: Pickands theorem [13]). *The distribution function $\mathcal{P}_\xi$ of the peaks $\mathcal{C}_u = \mathcal{C} - u$ over a threshold u of the sequence $\mathcal{T}$ of execution time measurements from a distribution function $\mathcal{C}$, $\mathcal{C} \in MDA(\mathcal{G}_\xi)$ whose $\mathcal{G}_\xi$ parameters are ξ, μ, σ, relatively to $\mathcal{C} > u$, is a Generalized Pareto Distribution (GPD) defined as* $\mathcal{P}_\xi(y) = \begin{cases} 1 - (1 + \xi y/\alpha_u)^{-1/\xi} \; if \; \xi \neq 0 \\ 1 - \exp(-y/\alpha_u) \; if \; \xi = 0 \end{cases}$, *with $\alpha_u = \mu - \xi(u - \sigma)$, and defined on $\{y, 1 + \xi y/\alpha_u > 0\}$. The conditional distribution function $\mathcal{C}_u$ of $\mathcal{C}$ above a certain threshold u, the conditional excess distribution function, is defined such as* $\mathcal{C}_u(y) = P(\mathcal{C} - u \leq y | \mathcal{C} > u) = \frac{\mathcal{C}(u+y) - \mathcal{C}(u)}{1 - \mathcal{C}(u)}$.

Hence, P_ξ is the kind of distribution to use for estimating the pWCET distribution i.e. $F_{\overline{\mathcal{C}}_i} = P_\xi$. The threshold u has a key role in the pWCET estimation.

As the threshold is chosen near the worst measured execution time, the law of the peaks tend to a GPD if and only if the measured empirical distribution ($\mathcal{C}_i$) belongs to the maximum domain of attraction of $\mathcal{G}_\xi$, $\mathcal{C}_i \in MDA(\mathcal{G}_\xi)$, i.e. iff the Fisher and Tippet theorem is verified. Formally there exists equivalence between the POT and the BM EVT approaches, as the law of extreme execution times given by G_ξ and the BM is closely linked to the law of peaks above the thresholds P_ξ. This translates into the equivalence of the distribution laws composing both the GEV and GPD distributions G_ξ and P_ξ, as they share the same value of ξ.

The meaning of independence looked for by the EVT is whether individual measurements $C_1, \ldots, C_n$ within the same trace are correlated with each other or not, i.e., the time history relationship. The identical distribution hypothesis assumes that all measurements follow the same distribution $\mathcal{C}_i$.

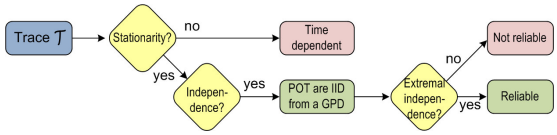


Fig. 1. Decision diagram for DIAGXTRM: Actions and tests for EVT applicability.

Recent works show that independence is not a necessary hypothesis for EVT applicability. Leadbetter et al. [22], Hsing [16] and Northrop [28] developed EVT for stationary weakly dependent time series, extending EVT applicability. In particular, [32,5] demonstrated the *applicability* of the EVT to the worst-case execution time estimation problem in case of some low degree of dependence between measurements (non time-randomized, like the GPUs in our case). Even the identical distribution (i.d.) of random variables does not represent a limiting hypothesis to EVT applicability. Specifically, [26] states the applicability of EVT to non-i.d. random variables, by considering stationary measurements.

3 Measurement-based Probabilistic GPU WCET analysis

MBPTA uses the EVT for estimating pWCETs, [9,32,5]. In this work we apply the newly developed DIAGXTRM MBPTA framework in order to diagnose execution time traces and derive safe pWCET estimates with the EVT.

Figure 1 describes the logic flow with the basic steps that DIAGXTRM follows in order to verify measurements' independence, how to apply the EVT in the more generic and realistic case of extreme independence, and evaluating the reliability/confidence of the resulting worst-case estimates. In this work we make use of the EVT POT approach, for which we make use of the whole GPD distribution comparing the results of the $\xi \neq 0$ case (from the best-fit algorithm to select the ξ value that best-fit the input measurements) with the $\xi = 0$ (the Gumbel case). The Gumbel distribution is kept because it was considered in the past to better fit inferences at low probability levels with regard to measurements and the pessimism of the pWCET estimates, [9,32,5].

DIAGXTRM is automatic in the sense that it selects the parameters i.e. shape ξ, and threshold u, which best fit the input data $\mathcal{T}$ and reduce the pessimism of the pWCET estimates. Furthermore, DIAGXTRM offers a complete set of tests for verifying EVT applicability hypotheses and it considers confidence metrics for evaluating both the hypotheses and the pWCET estimates. If all the tests are passed we can rely on the pWCETs from the EVT as safe estimation of task worst-case execution times. DIAGXTRM, unlike current measurement-based probabilistic timing analysis [9], may be applied also to non-time-randomized multi-core architectures as it evaluates the degree of dependence in $\mathcal{T}$ and defines the reliability/confidence of the worst-case estimates for specific parameters.

3.1 EVT Applicability for GPUs

With traces, one may study the relationship between measurements to evaluate (i) the distribution that every $C_{i,j}$ follows i.e. the i.d., and (ii) the impact that

previous (in time) measurements would have on future ones, i.e., the degree of dependences between measurements. Such relationships can only be statistically verified. Hereby we describe the 3 main tests applied for EVT hypothesis verification, thus for validating EVT applicability and EVT reliability.

Stationarity. The EVT applicability (in its relaxed form, [32,22]) relates to strictly stationary traces. In a strictly stationary trace (C_1, C_2,...), for any j, k, ℓ, the subtrace $C_j, \ldots, C_{j+k}$ is governed by the same probabilistic law as subtrace $C_{\ell+j}, \ldots, C_{\ell+j+k}$. Statistical tests exist for checking if a trace is strictly stationary or not; one of the most reliable is the Kwiatowski Phillips Schmidt Shin (KPSS) test [20], where results below 0.74 guarantee the trace as stationary. The threshold of 0.74 is achieved for a 1% confidence level: if the KPSS result value is below 0.74, then with a confidence of 0.99 the stationarity is acceptable. The KPSS test indirectly evaluates the i.d. hypothesis. The resulting confidence ρ^{KPSS} on the test translates into a confidence on the i.d. hypothesis.

Patterns and Correlation. The statistical dependence translates into correlated patterns of execution time measurements. One reliable statistical test for identifying correlated patterns is the Brock Dechert Scheinkman (BDS) test based on the correlation integral [7]. The test measures the degree of correlation between patterns of different lengths within a trace. For non-stationary traces the statistic diverges. The BDS results are expressed as the percentage of the independence hypothesis acceptance: the higher the percentage is, the more acceptable is the hypothesis to consider independent measurements. Implicit in the BDS result there is the confidence information on the i. hypothesis. ρ^{BDS} as the result of the BDS test defines the confidence on the independence hypothesis.

Extremal independence. When overall independence does not hold, another way is to look for independence of extreme execution time measurements[5]. Leadbetter [21] introduced two formal conditions for stationary dependent sequences that guarantee the EVT application. Condition $D(u_n)$ means that for execution time measurements that are *distant enough* in the trace of measurements (e.g., $C_{i,j}$ and $C_{i,j+I}$ with the distance I), these measurements can be considered as independent. Condition $D'(u_n)$, if verified, prevents from the clustering of the extreme execution time measurements: if one measurement is over the threshold then the probability that the following measurements are over the threshold too must tend to zero, to not have clustering. Considering an independent measurement sequence, whose limit law is $\mathcal{P}_\xi$ and with the same distribution as the stationary dependent sequence whose limit law is $\mathcal{H}_\xi$, the relationship between the two is such that $\mathcal{H}_\xi(x) = \mathcal{P}_\xi^\theta(x)$.

The Extremal Index (EI) $\theta \in [0, 1]$ is an indicator of the dependence degree of extreme measurements for time series [28]. The worst-case profile produced in case of extreme dependence (ed) $\overline{\mathcal{C}}_i^{ed}$ ($\theta < 1$) is greater than or equal to the one produced in case of extreme independence (ei) $\overline{\mathcal{C}}_i^{ei}$: $\overline{\mathcal{C}}_i^{\theta<1} \succeq \overline{\mathcal{C}}_i^{\theta=1} \equiv \overline{\mathcal{C}}_i^{ei}$. To note that the case $\theta = 1$ is equivalent to the independent case. The ordering of former equation is assured if and only if both extreme independence and independence cases follow the same average distribution. It describes a very im-

[5] By extreme execution time measurements we intend execution time relatively far from the average values or relatively separated in time.

portant relationship between extremal dependence degrees and the independence of the execution times. The effects of extremal dependence are in the direction of adding pessimism to the pWCET estimates: the pWCETs with dependence between measurements are more pessimistic but safer as worst-case estimates. On the other end, papers like [23] that claim to artificially build the independence from dependent execution time should better consider the effects of that, as removing dependences could harm the safety of the pWCET estimates.

In practice, to validate the extremal independence, the EI is enough; with $\theta \simeq 1$ either $D(u_n)$ and/or $D'(u_n)$ are valid, thus the extremal independence is guaranteed. The closer θ is to 1, the greater the confidence. $\rho^{EI} = \theta$ is the confidence measure on the extremal independence hypothesis.

EVT Confidence. The BDS test and the EI estimation jointly validate the EVT applicability wrt the independence, as $\max\{\rho^{BDS}, \rho^{EI}\}$. For a metric of confidence in *both* the i. and i.d. hypotheses, hence confidence in the full applicability of the EVT and the pWCET estimates from it, we can define ρ as

$$\rho = min\{\rho^{KPSS}, max\{\rho^{BDS}, \rho^{EI}\}\}. \tag{2}$$

4 Experiments

Our CUDA benchmark is the Voronoi diagram generator [5] inspired by the work of Majdandzic et al [24]. The raster size X by Y determines the number of threads. The per-thread workload scales linearly with K, the number of points (informally "tiles"), used as input. All experiments use $K = 32$ (for constant per-thread workload) and X=Y=256 (for constant overall workload) and we vary independently (i) the thread block size and (ii) the division of on-chip memory into L1 and "shared memory", to see the impact on the pWCET.

The four thread organization scenarios considered were: 64/256/512/1024 thread blocks (respectively, 1024/256/128/64 threads per thread block). Regarding the on-chip memory per SM, it is divided in two parts. The part used as L1 cache is managed by the driver. The other part, called "shared memory", is managed by the developer. The API provides three options for dividing the on-chip memory between these two parts: 75%/25%, 50%/50%, 25%/75%. Thus, we ran 4×3=12 sets of experiments on Kepler GK104 (8 SMs and 64KB of on-chip memory per SM). We label each trace by the number of thread blocks and the fraction of shared memory used for L1, e.g. "512 TB 75%". Our tool repeatedly cold-reboots, launches the kernel and records its timing measurements.

To later safely apply EVT and infer the pWCET estimate $\overline{C}_i$, we need enough measurements per trace. How many, we assess with the desired confidence level *a posteriori*, via the appropriate tests for stationarity, patterns and correlation, and extremal independence. If the tests fail, we add measurements to the trace, until they succeed. In our case, 50000 runs per trace proved sufficient (see below).

4.1 Timing Analysis

The GPU execution time traces show enough variability to be described by random variables. Even if it is a deterministic system (non time-randomized), the

$\mathcal{T}$	ρ^{KPSS}	ρ^{BDS}	ρ^{EI}	u	ρ	ξ	ET-10^{-5}	GPD-10^{-9}	$a(GPD)$	Gumbel-10^{-9}	$a(Gumbel)$
1024 TB 75%	0	0	1	120594	0	0.06, NEG	132301	147056	0.112	177567	0.342
1024 TB 50%	0.645	0.574	0.999	95910	0.99	0.1, NEG	105627	118193	0.119	133150	0.261
1024 TB 25%	0.581	0.056	0.993	94302	0.99	0.1, NEG	104354	117461	0.126	133291	0.277
512 TB 75%	0.764	0.917	1	118650	0.99	0.04, NEG	139878	220798	0.579	187595	0.341
512 TB 50%	0.622	0.889	1	118161	0.99	0.04, POS	141671	193812	0.368	174113	0.229
512 TB 25%	0.876	0.935	0.995	116361	0.99	0.09, POS	137438	230602	0.678	165673	0.205
256 TB 75%	0	0.972	0.983	103168	0	0.02, POS	147773	164985	0.116	154373	0.045
256 TB 50%	0.508	0.972	0.995	104024	0.99	0.14, NEG	120882	129227	0.069	162471	0.344
256 TB 25%	0.891	0.75	0.965	102347	0.99	0.17, NEG	116650	124517	0.067	164717	0.412
64 TB 75%	0.936	1	1	152653	0.99	0.26, NEG	179799	183408	0.02	277551	0.544
64 TB 50%	0.543	0.741	0.989	153905	0.99	0.21, NEG	179426	187952	0.048	265640	0.48
64 TB 25%	0.905	0.667	0.911	152575	0.99	0.12, NEG	178781	197487	0.105	234506	0.312

Table 1. Statistical results on the traces.

interactions between system elements, e.g., concurrent access to shared resources, create unpredictability from one execution to another. The average profiles $\mathcal{C}_i$ can be seen as discrete random variables because the time is measured in cycles.

The variability is quantified by applying KPSS, BDS and EI tests to the traces. From the results (Table 1), the variability is enough to have θ very close to 1, if not 1: the extremal independence of the execution times is guaranteed for all traces investigated. Moreover, the resulting pWCET estimates from the EVT would be the same as those with independent traces, since $\rho^{EI} \approx 1$. The confidence metric of Equation (2) outlines the large confidence we would have on the EVT applicability, thus on the EVT pWCET estimates. In statistical hypothesis testing, a confidence of 0.99 for accepting a hypothesis is very large.

For two limit cases, *256 TB 75%* and *1024 TB 75%*, the stationarity and so the i.d. hypothesis are not guaranteed. The independence is still guaranteed by θ. Looking at their traces, we spot no patterns among the execution times, neither trends characterizing the task execution evolution. It is clear how they represent two false negatives from the KPSS test. Notably, the extremal independence of *256 TB 75%* and *1024 TB 75%* is very strong.

This first statistical analysis shows that EVT can also be applied with high confidence ($\rho \approx 1$) even to some non-time-randomized systems (in this case, GPUs).

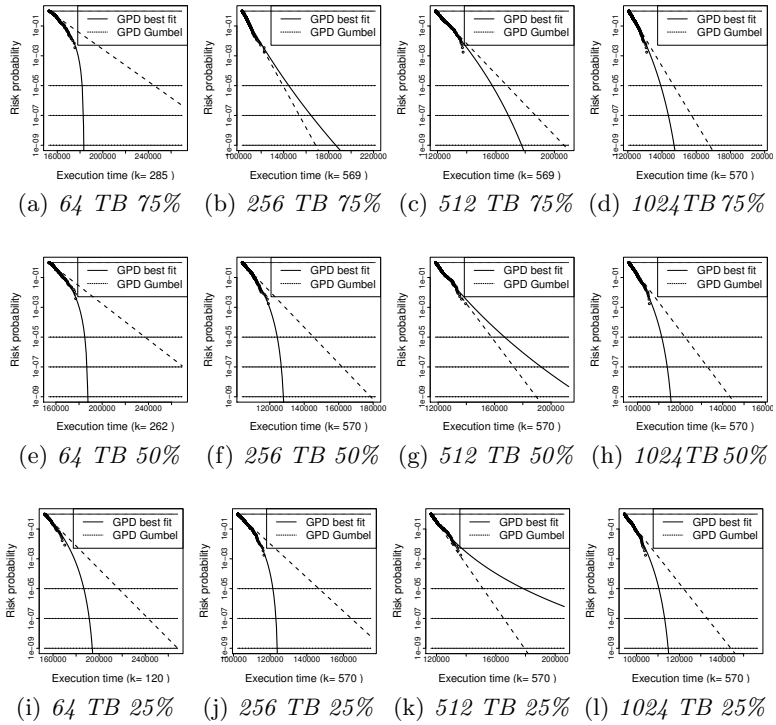
4.2 pWCET with the EVT

Equation (3) defines the accuracy metric a through which we evaluate pWCET estimates with respect to the execution time measurements in $\mathcal{T}$:

$$a \stackrel{def}{=} \frac{\text{WCET thresholds at } 10^{-9} - \text{ maximum observed value}}{\text{WCET thresholds at } 10^{-9}}, \qquad (3)$$

which translates into $a = (\overline{C}_i^\sharp - C_i^\sharp)/\overline{C}_i^\sharp$, where $\overline{C}_i^\sharp$ is such that $P(\overline{\mathcal{C}}_i > \overline{C}_i^\sharp) = 10^{-9}$, and $C_i^\sharp$ such that $P(\mathcal{C}_i > C_i^\sharp) = 2{\cdot}10^{-5}$; $2{\cdot}10^{-5}$ is the minimum observable probability, as $\frac{1}{50000}$, from the size of the traces.

Figure 2 compares all the traces of measurements with their EVT pWCET estimates with both $\xi = 0$ (the Gumbel pWCET estimate) and ξ resulting from the best-fit procedure implemented within DIAGXTRM. It is worthy to note the difference that exists between the two estimates. This is motivated by the fact that the best-fit procedure best-fits the input traces, thus the known information, which do not necessarily follow a Gumbel distribution at the extremes. While



(a) *64 TB 75%* (b) *256 TB 75%* (c) *512 TB 75%* (d) *1024TB 75%*

(e) *64 TB 50%* (f) *256 TB 50%* (g) *512 TB 50%* (h) *1024TB 50%*

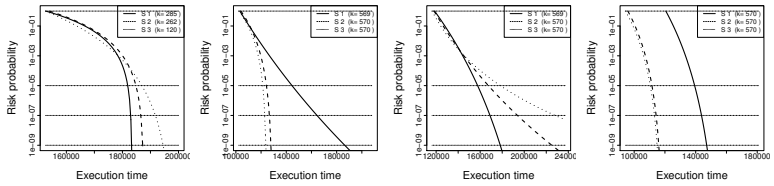
(i) *64 TB 25%* (j) *256 TB 25%* (k) *512 TB 25%* (l) *1024 TB 25%*

Fig. 2. Direct comparisons (# of thread blocks; % of on-chip memory used for L1).

with the best-fit procedure, the input measurements are best modeled, the tail of the distribution is not necessarily accurate; this is the case of $\xi > 0$. As we can see, there are also cases where $\xi < 0$ and the pWCET estimate is a Weibull distribution, which is more accurate than the Gumbel at the tail too.

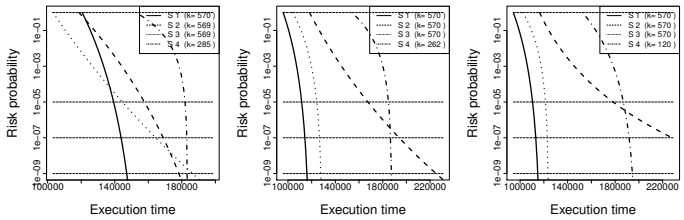
With DIAGXTRM we are able to achieve a pWCET estimate accuracy of at worst 68% with respect to the maximum measured value $C_i^\sharp$, see Table 1, with both Gumbel or GPD with $\xi \neq 0$. Most of the ξ resulting from the best-fit algorithm are negatives, thus the pWCET estimates take the form of Weibull distributions and a better accuracy. In a few cases, $\xi > 0$ (see Table 1). For those, the resulting pWCET is more conservative (potentially less accurate) by comparison with the other traces. In those cases the pWCET has a finite support and has a better accuracy than the Gumbel distribution. The positive values we obtain are close to 0, thus the GPD in those cases has a shape very close to the shape of Gumbel distributions: under a certain probability range they can be considered equivalent to the Gumbel pWCET estimates.

Figures 3 and 4 compare pWCET estimates in different execution scenarios. For the comparison, applied the EVT POT with the best threshold u selection was applied, in order to increase the accuracy of the pWCET estimates; k is the number of the measurements over the threshold u used to infer the pWCET estimation. It depends on u as a direct result of the best-fit approach. The



(a) *64 TB* (b) *256 TB* (c) *512 TB* (d) *1024 TB*

Fig. 3. Comparison between on-chip memory splitting.



(a) *75% L1* (b) *50% L1* (c) *25% L1*

Fig. 4. Comparison between threads per thread block.

Gumbel distribution is chosen to compare with the best-fit resulting GPDs and to comply with previous works. It allows us also to outline that DIAGXTRM can face any resulting GPD shape. In most of the time, the resulting GPD is a Weibull distribution ($\xi < 0$), thus the pWCET estimates will have finite support and will be less pessimistic than the Gumbel distribution; this is the direct result of the best-fit approach which best-fits measurements and not necessarily concludes that Gumbel is the best approximation to the worst-case behaviors.

In Figure 3, due to shortage of space, labels S1, S2 and S3 correspond to 75%, 50% and 25% of the per-SM on-chip memory used as L1 cache (the rest being "shared memory"). In the case of Figure 3(a) (which corresponds to 64 TBs), interestingly, the thread thresholds which maximizes the accuracy for S1, S2, S3 differ from each other; unlike what holds for the other three cases. The reason for that comes from the best-fit parameter selection algorithm we have implemented and the shape of the input traces: to best-fit the input traces the threshold could vary, and in this case it does so more than in others.

Figure 3 shows that, depending on the number of thread blocks in which the kernel is configured, a bigger L1 may have either a positive or a *negative* effect on both average-performance and the pWCET. For 64 TBs or 512 TBs, the pWCET is smaller with smaller L1, which is anomalous. But for "interleaved" cases of 256 TBs and 1024 TBs, a bigger L1 helps. Strikingly, there is no monotonic trade-off with the number of thread blocks. We attribute this to a strange interplay of various micro-architectural effects, most likely including the hit rate on the shared L2 (especially, since the cache hierarchy is not strictly inclusive but not exclusive either [8]). As for the number of thread blocks, NVIDIA ac-

knowledges [6] that the thread block size is not a simple tradeoff. Our experiments demonstrate that this reality also extends to the pWCETs. Figure 4 organizes the same information as Figure 3 differently, to highlight the effect of thread block organization. Here, again due to shortage of space, the labels S1 to S4 now correspond to the number of thread blocks (1024/512/256/64, respectively).

DIAGXTRM captures even such counter-intuitive performance dependencies and allows the designer to optimize according to the pWCET by choosing the best configuration. For example, if the exceedance probability of interest is 10^{-9}, the best-performing configuration is 1024 TBs and 25% L1.

To conclude, apart from the two case limits, with non time-randomized architectures such as the GPUs considered, it is still possible to verify EVT applicability with extremely high confidence. Such confidence propagates to the pWCET estimates achieved with the EVT. Finally, with Equation (2) we are able to relate test confidence to the confidence in the whole EVT approach.

5 Conclusions

This work applied the DIAGXTRM MBPTA approach to GPUs. The results show that hardware time-randomization is not strictly necessary for the applicability of EVT. Indeed the execution time traces, even when dependent, are all independent at the extremes, resulting in pWCET estimates as accurate as those from fully independent traces. Using generic GPDs or GEVs, not limiting the pWCET estimates to Gumbel distributions, allows for accurate pWCET estimates. The best-fit of the input measurements usually led to better extreme event estimation than the Gumbel assumption. We also compared GPU execution scenarios using DIAGXTRM to study system behavior with probabilistic models.

In the future, we will investigate other system configurations and /or other system elements and apply the sensitivity analysis to evaluate their effect on the pWCET estimates. Our goal is to develop an aided-design probabilistic framework for more deterministic GPU development. Concerning DIAGXTRM, we will enhance its tests to reduce both false positives and false negatives and increase the confidence in its tests and EVT estimates.

Acknowledgements: Work partially supported by National Funds through FCT/MEC (Portuguese Foundation for Science and Technology) and co-financed by ERDF (European Regional Development Fund) under the PT2020 Partnership, within project UID/CEC/04234/2013 (CISTER); also by FCT/MEC and the EU ARTEMIS JU within projects ARTEMIS/0003/2012 - JU grant 333053 (CONCERTO) and ARTEMIS/0001/2013 - JU grant 621429 (EMC2); by FCT/MEC and ESF (European Social Fund) through POPH (Portuguese Human Potential Operational Program), under PhD grant SFRH/BD/82069/2011.

References

1. A. Bakhoda, G. L. Yuan, W. W. Fung, H. Wong, and T. M. Aamodt. Analyzing CUDA workloads using a detailed GPU simulator. In *Proc. IEEE ISPASS*, 2009.
2. M. Bautin, A. Dwarakinath, and T. Chiueh. Graphics Engine Resource Management. In *Proc. 15th ACM/SPIE MMCN*, 2008.
3. K. Berezovskyi, K. Bletsas, and B. Andersson. Makespan computation for GPU threads running on a single streaming multiprocessor. In *Proc. 24th ECRTS*, 2012.
4. K. Berezovskyi, K. Bletsas, and S. M. Petters. Faster makespan estimation for GPU threads on a single streaming multiprocessor. In *Proc. ETFA*, 2013.

[6] As stated in [29], p .47: "There are many factors involved in selecting block size, and inevitably some experimentation is required."

5. K. Berezovskyi, L. Santinelli, K. Bletsas, and E. Tovar. WCET Measurement-based and EVT Characterisation of CUDA Kernels. In *Proc. RTNS*, 2014.
6. A. Betts and A. F. Donaldson. Estimating the WCET of GPU-accelerated applications using hybrid analysis. In *Proc. 25th ECRTS*, pages 193–202, 2013.
7. W. Brock, J. Scheinkman, W. Dechert, and B. LeBaron. A test for independence based on the correlation dimension. *Econometric Reviews*, 15(3), 1996.
8. X. Chen, L.-W. Chang, C. I. Rodrigues, J. Lv, Z. Wang, and W.-M. Hwu. Adaptive cache management for energy-efficient gpu computing. In *Proc. 47th IEEE/ACM Int. Symp. on Microarchitecture*, pages 343–355, 2014.
9. 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 *Proc. 23nd ECRTS*, 2012.
10. R. I. Davis, L. Santinelli, S. Altmeyer, C. Maiza, and L. Cucu-Grosjean. Analysis of probabilistic cache related pre-emption delays. *Proceedings of the 25th IEEE Euromicro Conference on Real-Time Systems (ECRTS)*, 2013.
11. J. Díaz, D. Garcia, K. Kim, C. Lee, L. Bello, L. J.M., and O. Mirabella. Stochastic analysis of periodic real-time systems. In *23rd RTSS*, pages 289–300, 2002.
12. G. Elliott, B. Ward, and J. Anderson. GPUSync: Architecture-aware management of GPUs for predictable multi-GPU real-time systems. Proc. RTSS 2013.
13. P. Embrechts, C. Klüppelberg, and T. Mikosch. *Modelling extremal events for insurance and finance*. Applications of mathematics. Springer, 1997.
14. E. Gumbel. *Statistics of Extremes*. Columbia University Press, 1958.
15. V. Hirvisalo. On static timing analysis of GPU kernels. In *Proc. WCET*, 2014.
16. T. Hsing. On tail index estimation using dependent data. *Ann. Stat.*, 1991.
17. S. Kato, K. Lakshmanan, A. Kumar, M. Kelkar, Y. Ishikawa, and R. Rajkumar. RGEM: A responsive GPGPU execution model for runtime engines. RTSS 2011.
18. S. Kato, K. Lakshmanan, R. Rajkumar, and Y. Ishikawa. Timegraph: GPU scheduling for real-time multi-tasking environments. In *USENIX ATC*, 2011.
19. S. Kato, M. McThrow, C. Maltzahn, and S. Brandt. Gdev: First-class GPU resource management in the operating system. In *Proc. USENIX ATC*, 2012.
20. D. Kwiatkowski, P. C. B. Phillips, P. Schmidt, and Y. Shin. Testing the null hypothesis of stationarity against the alternative of a unit root : How sure are we that economic time series have a unit root? *J. of Econometrics*, 54(1-3), 1992.
21. M. R. Leadbetter, G. Lindgren, and H. Rootzén. Conditions for the convergence in distribution of maxima of stationary normal processes. *Stochastic Processes and their Applications*, 8(2):131–139, 1978.
22. M. R. Leadbetter, G. Lindgren, and H. Rootzén. *Extremes and Related Properties of Random Sequences and Processes*. Springer-Verlag, 1983.
23. Y. Lu, T. Nolte, I. Bate, and L. Cucu-Grosjean. A statistical response-time analysis of real-time embedded systems. In *Proc. RTSS*, pages 351–362, 2012.
24. I. Majdandzic, C. Trefftz, and G. Wolffe. Computation of Voronoi diagrams using a graphics processing unit. In *IEEE Int. Conf. Electro/Inf. Tech. (EIT)*, 2008.
25. R. Mangharam and A. A. Saba. Anytime algorithms for GPU architectures. In *Proceedings of the 32nd IEEE Real-Time Systems Symposium (RTSS)*, 2011.
26. D. Mejzler. On the problem of the limit distribution for the maximal term of a variational series. *Lvov. Politehn. Inst. Naucn Zap. Ser. Fiz.-Mat.*, 1956.
27. R. Membarth, J.-H. Lupp, F. Hannig, J. Teich, M. Körner, and W. Eckert. Dynamic task-scheduling and resource management for gpu accelerators in medical imaging. In *Architecture of Computing Systems–ARCS 2012*, pages 147–159. Springer, 2012.
28. P. Northrop. Semiparametric estimation of the extremal index using block maxima. Technical report, Dept of Statistical Science, UCL, 2005.
29. NVIDIA Corp. CUDA C Best Practices Guide. DG-05603-001_v5.5.
30. NVIDIA Corp. Whitepaper: Kepler GK110. www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf, 2012.
31. C. J. Rossbach, J. Currey, M. Silberstein, B. Ray, and E. Witchel. Ptask: Operating system abstractions to manage GPUs as compute devices. ACM SOSP 2011.
32. L. Santinelli, J. Morio, G. Dufour, and D. Jacquemart. On the sustainability of the extreme value theory for WCET estimation. In *Int. WCET Workshop*, 2014.