Modeling and Decoupling the GPU Power Consumption for Cross-Domain DVFS

João Guerreiro, *Student Member, IEEE,* Aleksandar Ilic, *Member, IEEE,* Nuno Roma, *Senior Member, IEEE,* and Pedro Tomás, *Senior Member, IEEE*

Abstract—Dynamic voltage and frequency scaling (DVFS) is a popular technique to improve the energy-efficiency of high-performance computing systems. It allows placing the devices into lower performance states when the computational demands are lower, opening the possibility for significant power/energy savings. This work presents a GPU power consumption model, used to predict the GPU power consumption of any application at different frequency levels. To obtain this model, an estimation algorithm is proposed, relying on careful benchmarking of the GPU architecture. The model can estimate the contribution of twelve different GPU components (FP32-ADD/MUL/FMA, FP64-ADD/MUL/FMA, INT, SF, CF units, shared memory, L2-cache, and DRAM) to the GPU power consumption. Different model use cases are evaluated (fixed-frequency, DVFS, and scaling-factors), which can obtain both the total or the per-component GPU power consumption. A technique to export models to a distinct GPU from the one it was estimated on is also proposed. These approaches were extensively validated on five different GPUs from the three most recent microarchitectures with a set of 42 standard benchmarks, achieving very accurate predictions. In particular, the scaling-factor power model achieves an average prediction error of 3.5% (Titan Xp), 4.6% (GTX Titan X), 3.1% (GTX 980) and 2.4% (Tesla K40c).

Index Terms—GPGPU, DVFS, Power Modeling, Scaling-Factors.

1 Introduction

As the usage of accelerators, particularly GPUs, has become more predominant in high-performance computing (HPC) systems [1], it is increasingly important to find mechanisms to maximize their energy-efficiency. One of the most widely used techniques is the dynamic voltage and frequency scaling (DVFS), which allows placing the devices into lower performance states. When carefully applied to match the needs of the executing applications, DVFS can lead to significant power and energy savings, often with minimum impact on performance [2], [3], [4].

Nonetheless, to efficiently apply power management techniques (including DVFS), accurate models are required to predict how the performance/power consumption of applications scales with the GPU operating frequencies/voltages. In the past, it has been shown that applications with different GPU resources utilizations have diverse performance and power consumption scaling behaviours when DVFS is applied [5], [6], [7], [8]. Hence, attaining accurate models requires information of how the executing applications are using the different GPU components.

Previous works proposing GPU power consumption models have focused on either fixed frequency [9], [10], [11] or more recently on DVFS prediction [12], [13]. In our previous work [14], a DVFS-aware GPU power consumption model was proposed, which relies on a set of carefully devised microbenchmarks and on a regression-based algorithm to estimate the unknown model parameters. Such devised model allows decoupling the power consumption in seven different components with a high accuracy. This was also one of the first works to consider the non-linear scaling of the GPU voltage with the operating frequency.

This paper significantly expands on our previous work [14] with an extensive focus on the different usage scenarios of power models such as: 1) fixed frequency predictions; 2) DVFS predictions; 3) scaling-factor predictions; and 4) per-component power breakdown in twelve different components. This work also introduces a detailed analysis of the effects of hardware changes in the power consumption model, namely on the portability of an estimated model. Furthermore, the microbenchmark suite is extended with new applications that exercise multiple new GPU components.

Each of these different approaches was extensively validated on five different GPU devices (Titan Xp, GTX Titan X, GTX 980, GTX 960 and Tesla K40c) from the three most recent NVIDIA GPU microarchitectures (Pascal, Maxwell and Kepler) with a set of 42 benchmarks from five commonly used benchmark suites (Parboil [15], Rodinia [16], Polybench [17], SHOC [18] and CUDA SDK [19]). From the conducted experimental evaluation it is shown that the proposed models achieve accurate results, particularly the scaling-factor power model which achieves an average error rate as low as 3.5% (Titan Xp), 4.6% (GTX Titan X), 2.4% (Tesla K40c) and even 3.1% for the GTX 980, where the last one was obtained by adapting the model estimated for the GTX Titan X GPU.

Accordingly, this work makes the following contributions with regards to [14]:

- Introduction of additional microbenchmarks into the microbenchmark suite, allowing the characterization of novel GPU components, as well as the separation

- *The authors are with INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, 1000-029 Lisbon, Portugal (e-mail: {Joao.Guerreiro,Aleksandar.Ilic,Nuno.Roma,Pedro.Tomas}@inesc-id.pt).*

Manuscript received August 2, 2018; revised February 15, 2019.

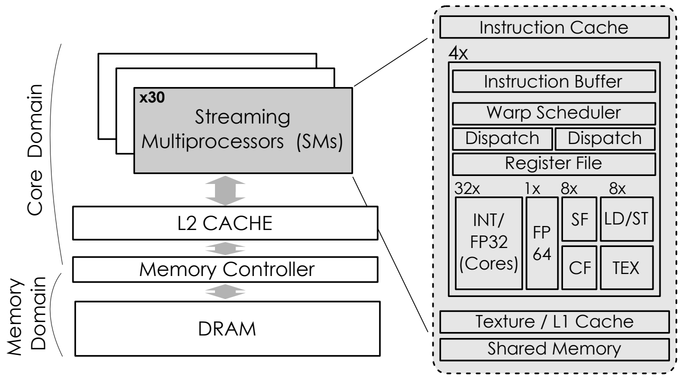


Fig. 1: Block diagram of a Titan Xp GPU (Pascal family).

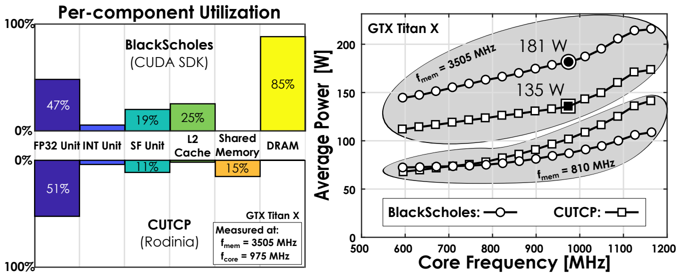


Fig. 2: DVFS impact on the power consumption of two applications on the GTX Titan X. Left: GPU components utilization during the applications execution, when $f_{core} = 975$ MHz and $f_{mem} = 3505$ MHz. Right: power consumption variation with the core and memory frequency scaling.

of some previously aggregated components (*e.g.* by separating FP32 unit into FP32 ADD, FP32 MUL and FP32 FMA compute units).

- New power model and voltage estimation algorithm, allowing to simultaneously estimate the voltage of each frequency domain as well as the unknown model coefficients.
- Extensive validation of the estimated power consumption models in multiple usage scenarios (fixed-frequency, DVFS, scaling-factors, power breakdown), including the evaluation of the results obtained from exporting an estimated model from one GPU to a different one.
- Discussion of the obtained results, including considerations about the general limits of power models based on supervised learning (statistical- or machine-learning).

The complete source code, including the microbenchmark suite and a tool to construct the DVFS-aware GPU power consumption model, is publicly available online (https://github.com/hpc-ulisboa/gpupowermodel).

The rest of this paper is organized as follows. Section 2 motivates the presented work, including a summary of the most relevant *state-of-the-art* works. Section 3 details the proposed DVFS-aware power model. Section 4 presents the different model use cases, validated on real hardware devices. Section 5 overviews the obtained results and Section 6 concludes the manuscript.

2 Background and Motivation

The architecture of GPUs is composed by several distinct components (as an example, Fig. 1 represents a NVIDIA Titan Xp GPU). The main execution components of GPUs are the streaming-multiprocessors (SMs), which include different computational units (INT, FP32, FP64, *etc.*), as well as several elements of private memory (texture/L1-cache, shared memory). GPU devices usually have multiple SMs (30 in the case of the Titan Xp), as well as an L2-cache and the main device memory (DRAM).

2.1 GPU DVFS and Power Consumption

Most GPU devices have two independent frequency domains, which are the *core* (or graphics) *domain*, clocked at f_{core}, and the *memory domain*, clocked at f_{mem}. DVFS can be applied as a way of exploiting the existence of these two independent frequency domains, since it allows adapting the performance of the GPU components to the particular requirements of the executing applications. This can often result in considerable energy savings [2], [3], [20], [21].

However, optimizing the GPU configuration, *i.e.* the voltage-frequency (V-F) levels of both core and memory domains is not a trivial problem [7], [8], [22]. It requires accurate estimations of both the execution time and average power consumption, namely the effects that changing the V-F configuration will have on these two metrics.

Different GPU applications have their unique characteristics (used algorithm, data types, operations, size of the input data, dimensions of the grid of threads, *etc.*), which determine how the different GPU components are stressed during the application execution. Furthermore, depending on how the different GPU components are exercised by applications, DVFS can have vastly different impacts on the performance and on the total GPU power consumption.

Fig. 2 presents an example of such a scenario, where the *BlackScholes* and the *CUTCP* benchmarks are executed on an NVIDIA GTX Titan X GPU across multiple V-F configurations. Fig. 2 also shows the utilization of the main GPU components, represented as the ratio of the achieved and peak theoretical throughputs of the component. As it can be seen, the two applications present very different utilization rates of some GPU components (see L2-cache and DRAM utilizations), which results in the different power consumption levels of 181W and 135W at the default GPU frequency configuration ($f_{core} = 975$ MHz and $f_{mem} = 3505$ MHz). Additionally, it can also be seen that the variation of the power consumption when the memory frequency is decreased is much higher for the *BlackScholes* benchmark, because of its greater DRAM utilization: when the memory frequency decreases from 3505 MHz to 810 MHz, the power consumption decreases by 52% (from 181W to 87W). On the other hand, the power consumption of the *CUTCP* benchmark decreases by only 24% (from 135W to 102W).

Regarding the core frequency scaling, it can be seen that, unlike it is proposed in other previous works [10], [12], the GPU power cannot be represented as a simple linear function of the core frequency. In practice, the power consumption of a GPU device can be decomposed in the sum of the power consumptions of its multiple architectural components [23]. Furthermore, the power of each component ($\mathrm{C_k}$) is associated with its peak power consumption and with how an executing application stresses such component ($\mathrm{Power(C_k) \propto Utilization(C_k)}$).

With regards to this observation, Butts *et al.* [24] and Gonzalez *et al.* [25] proposed the power models presented in Eqs. 1 and 2, which can be used to describe how the dynamic and static components scale with the frequency and voltage of their respective hardware elements:

$$\mathrm{Power_{Static} = V \cdot N \cdot K_{design} \cdot \hat{I}_{leak}}, \tag{1}$$

$$\mathrm{Power_{Dynamic} = a \cdot C \cdot V^2 \cdot f}, \tag{2}$$

where a denotes the average utilization ratio, C the total capacitance, V the supply voltage, f the operating frequency and N the number of transistors in the chip design. K_{design} is a constant factor associated with the technology characteristics and $\hat{I}_{leak}$ is a normalized leakage current for a single transistor, which depends on the threshold voltage.

These models can already provide some reasoning about the non-linear behaviour of the power consumption in Fig. 2, as frequency scaling is usually accompanied by changes of the components voltage. However, even though these models give a valuable insight on the impact of DVFS, it is usually impossible to accurately measure these two components separately, let alone determine the model individual parameters. Consequently, other approaches to model the GPU power consumption are often required [12].

From these observations, it becomes clear the importance of accurate DVFS-aware power models to characterize the relationship between the GPU components utilization, their runtime power consumption and how they change when the frequency/voltage of the GPU domains are scaled.

2.2 Related Work

Initial attempts to model the GPU power consumption were focused on modeling the power at a fixed V-F configuration, not taking into consideration the effect of DVFS [26], [27], [28]. In particular, Nagasaka *et al.* [29] proposed a power consumption model for a Tesla GPU (GTX285) using a statistical approach to find the correlation between hardware performance events and the GPU power consumption, achieving an average prediction error of 4.7%. However, the authors stated that the approach was ineffective for more recent GPUs, namely those from the Fermi generation.

Hong *et al.* also proposed a power model for a Tesla GPU (GTX280) [9] based on an analysis of both the binary PTX and the device pipeline at runtime. The offline PTX analysis allows this model to attain highly accurate GPU power predictions, at the cost of being very GPU-specific. Hence, such an approach lacks the ability to make accurate predictions for different GPU architectures, or even for the same GPU at different core and memory configurations.

Song *et al.* proposed an artificial neural-network based power model for GPU devices [11], achieving better prediction accuracy than previous traditional regression-based models. However, neural network approaches usually lead to highly complex models, where it is often hard to extract its physical/architectural meaning.

Leng *et al.* integrated Hong's power model inside the GPGPU-Sim [30] simulator, resulting in the GPUWattch [10] tool. Hence, it only supports NVIDIA Tesla and Fermi GPU microarchitectures. GPUWattch can estimate the cycle-level GPU power consumption during application execution. However, it assumes that the power consumption of a GPU domain always scales linearly with its frequency [10, eq.6], which previous works (Mei *et al.* [8] and Guerreiro *et al.* [14]) showed to be often incorrect, because of the non-linear behaviour of the voltage scaling in some GPU devices. Nath *et al.* also used GPGPU-Sim to create a performance model for DVFS, which could potentially be expanded to include a power model [31]. However, this type of approaches often requires adding logic to the GPU scoreboard, making it impossible to replicate them on real hardware.

Abe *et al.* deemed the previous approaches to be product-specific and difficult to apply on modern GPUs, and proposed DVFS-aware power regression models for GPUs from the NVIDIA Tesla, Fermi and Kepler generations [12]. The authors separated the GPU power consumption in core and memory domains, each proportional to their corresponding frequency and associated performance events. The models were estimated through linear regression by using measurements taken at three different core and three different memory frequencies. The proposed models achieved average prediction errors of 15% for the Tesla GPU, 14% for the Fermi GPU and 23.5% for the most recent Kepler GPU. However, the work does not disclose the set of performance events used in the model. Additionally, despite performing the power consumption decomposition in the core and memory domains (similar to the one herein presented) the work proposed in [12] also does not consider the non-linear scaling effects of the voltage.

Wu *et al.* studied how the performance and power consumption of an AMD GPU scale with core and memory frequency variations, as well as with different number of cores [13]. The proposed work groups GPU applications into distinct clusters based on their characteristics, each representing a different performance/power scaling-factor. Properly trained neural-network classifiers are then used to characterize new applications, by predicting which scaling-factor better represents an application. They achieve an average prediction error of about 10% on the tested GPU device. However, the model accuracy is highly dependent on a set of fine-tuned parameters, such as the number of clusters, which makes it difficult to replicate on different architectures. More recently, a follow-up technique [32] was proposed that predicts the characteristics of upcoming kernels, based on recent execution history.

The work that is herein presented vastly expands over our previous work [14], where a DVFS-aware power consumption model was proposed. Such model could predict the total or per-component power consumption of GPUs for any voltage-frequency configuration, by using performance counters gathered at a single configuration. To estimate the model of each GPU device, a suite of microbenchmarks was provided and made publicly available, as well as a tool that implemented the devised iterative algorithm relying on statistical regression to model not only the unknown hardware characteristics but also to accurately predict how the core voltage scales with its frequency. The model was validated on three GPU devices from different microarchitectures using a set of 26 standard benchmarks, achieving average errors of 7% (Pascal), 6% (Maxwell) and 12% (Kepler). The herein presented work not only extends on our previous power model (using several more GPU components), but also provides an extensive focus on different usage scenarios (*e.g.*, fixed frequency predictions, DVFS predictions, scaling-factor predictions, per-component power breakdown and the effects of hardware changes), ultimately resulting in more accurate GPU power consumption predictions.

2.3 Applications of a GPU Power Model

As it was referred above, one advantage of this work over previous models (*e.g.* over machine-learning based ones, where it is hard to convey the architectural meaning of the estimated elements) is its versatility over multiple usage scenarios. Hence, the following use cases can be highlighted:

1) **DVFS management**, since the model allows searching for the optimal frequency state without exhaustive execution on all possible configurations (contrasting to [33]). With the proposed model it is possible to estimate the power consumption at different V-F configurations after executing the application at a single configuration (Section 4.4).
2) **Power consumption estimation in GPUs without power sensors**, by using a previously estimated model (*e.g.* using external sensors) to provide an estimate of the total and/or per-component GPU power consumption (similarly to [34] from Intel). Additionally, by using a power model such as the herein proposed one, *i.e.* based on architectural characteristics of the devices, it can also be possible to export a power model from one GPU device to a different one (Section 4.6).
3) **Power-aware optimization of applications**, as the model allows obtaining the per-component power consumption breakdown, which can help developers to assess the power bottlenecks of applications (Section 4.5), as an alternative to the more common performance optimization. This can even be useful in virtualization scenarios (such as the NVIDIA GRID system using Hyper-V execution [35]), where the model — constructed in the *Hypervisor* — could be provided to the guest VMs, allowing them to estimate their corresponding total and/or per-component power consumption (which they currently have no way of measuring).
4) **GPU hardware integration**, by implementing the proposed model in hardware (similar to Intel RAPL [36]), where it would be able to account for fine-grained V-F perturbations and potentially even non-SMU (System Management Unit) V-F adjustments.

3 GPU Power Consumption Model

To effectively apply DVFS techniques to optimize the power consumption of an application execution, it is fundamental to predict how the scaling of each GPU domain frequency/voltage affects its overall power consumption. This section describes the proposed procedure to create a statistical power consumption model of the GPU architecture.

To obtain an accurate model of the GPU power consumption, one must consider the decomposition of the power consumption across the internal components of the GPU. By taking into account that these components operate under different frequency and voltage domains, the following decomposition can be obtained:

$$\mathrm{P_{GPU}} = \sum_{k=1}^{\mathrm{N_{V\text{-}F}}} \mathrm{P}(\mathrm{D}_k), \tag{3}$$

where $\mathrm{N_{V\text{-}F}}$ is the number of independent voltage/frequency (V-F) domains and $\mathrm{P}(\mathrm{D}_k)$ is the power consumption of each domain (D_k), defined as follows:

$$\mathrm{P}(\mathrm{D}_k) = \alpha_0 \bar{\mathrm{v}}_\mathrm{k} + \bar{\mathrm{v}}_\mathrm{k}^2 \mathrm{f_k} (\alpha_1 + \sum_{i=1}^{\mathrm{N_C}} \gamma_i \cdot \mathrm{U}_i) \tag{4}$$

where $\mathrm{f_k}$ represents the frequency of domain $\mathrm{D_k}$, $\bar{\mathrm{v}}_\mathrm{k}$ is the normalized voltage of the domain ($\bar{\mathrm{v}}_\mathrm{k} = \mathrm{v_k}/\mathrm{v_{Ref.}}$), $\mathrm{N_C}$ is the number of GPU components operating in domain $\mathrm{D_k}$ and $\mathrm{U}_i \in [0,1]$ is their respective average utilization rate. The coefficients α_0, α_1, γ_1,...,γ_{N_C} represent a set of hardware-specific parameters, associated to the characteristics of the underlying architecture, such as component total capacitance and leakage resistance. In particular, the proposed power model contains the following distinct elements:

1) $\alpha_0 \bar{\mathrm{v}}_\mathrm{k}$: corresponding to the static power of domain $\mathrm{D_k}$ (see Eq. 1).
2) $\alpha_1 \bar{\mathrm{v}}_\mathrm{k}^2 \mathrm{f_k}$: corresponding to the constant power consumption of that V-F configuration of domain $\mathrm{D_k}$, *i.e.* the dynamic power that is independent of the modeled component utilizations.
3) $\gamma_\mathrm{i} \bar{\mathrm{v}}_\mathrm{k}^2 \mathrm{f_k} \mathrm{U_i}$: corresponding to the dynamic power of component i (see Eq. 2).

Terms $\alpha_0 \bar{\mathrm{v}}_\mathrm{k}$ and $\alpha_1 \bar{\mathrm{v}}_\mathrm{k}^2 \mathrm{f_k}$ correspond to what it is usually denoted by the idle power of that specific V-F level, independent of the utilization rates.

As previously stated, most modern GPU devices comprise two frequency domains, *i.e.* $\mathrm{N_{V\text{-}F}} = 2$, corresponding to the core and memory domains ($\mathrm{P_{GPU}} = \mathrm{P_{core}} + \mathrm{P_{mem}}$). By rewriting Eq. 4 considering these two domains and by denoting with $\mathrm{N_{core}}$ the number of modeled GPU components from the core domain, the following equations are obtained:

$$\mathrm{P_{core}} = \alpha_0 \bar{\mathrm{v}}_\mathrm{core} + \bar{\mathrm{v}}_\mathrm{core}^2 \mathrm{f_{core}} (\alpha_1 + \sum_{i=1}^{\mathrm{N_{core}}} \gamma_i \mathrm{U}_i), \tag{5}$$

$$\mathrm{P_{mem}} = \alpha_2 \bar{\mathrm{v}}_\mathrm{mem} + \bar{\mathrm{v}}_\mathrm{mem}^2 \mathrm{f_{mem}} (\alpha_3 + \gamma_\mathrm{mem} \mathrm{U_{mem}}). \tag{6}$$

3.1 Modeled GPU Components

The proposed power model (Eqs. 5 and 6) is based on the utilization rates of the modeled GPU components. These rates represent a reliable measure of how the considered application exercises the components during its execution. The proposed model focuses on modeling the hardware components that have a significant impact on power consumption during the execution of GPU applications (and which have associated performance counters available), namely: the

TABLE 1: Required metrics to compute the utilization rates used in the proposed power consumption model.

	Name	Domain
1	ACycles	Core
2-5	$\text{AWarps}_{\{\text{INT/FP32,FP64,SF,CF}\}}$	Core
6-7	$\text{Inst}_{\{\text{INT,FP32}\}}$	Core
8-10	$\text{FP32FLOPS}_{\{\text{ADD,MUL,FMA}\}}$	Core
11-13	$\text{FP64FLOPS}_{\{\text{ADD,MUL,FMA}\}}$	Core
14-15	$\text{ABand}_{\{\text{L2,Shared}\}}$	Core
16	$\text{ABand}_{\text{DRAM}}$	Memory

integer (INT), single- and double-precision floating-point (FP32/FP64), special-function (SF) and control-flow units (CF), the shared memory, the L2 cache and the DRAM. Additionally, the model also considers separate units for different FP compute instructions (FMA, ADD and MUL), as it was experimentally observed that their distinct complexities lead to different power consumption levels. Furthermore, the modular structure of the model allows an easy adaptation of it if/when new components (and corresponding performance counters) are added to new GPUs.

The utilization rate of the considered GPU compute units, measured during the application execution, is computed as the ratio between the number of executing warps with the number of warps that would execute if these units were always filled (theoretical peak). Hence, the utilization rates of the SM computational units can be expressed as:

$$\text{U}_\text{x} = \frac{\text{AWarps}_\text{x} \cdot \text{WarpSize}}{\text{ACycles} \cdot \text{UnitsPerSM}_\text{x}}, \qquad \text{x} \in \{\text{INT/FP32}, \text{FP64}, \text{SF}, \text{CF}\}, \quad (7)$$

where AWarps_x is the number of warps executing on unit x during the application execution, ACycles is the number of cycles when there is at least one active warp on the SMs, $\text{UnitsPerSM}_\text{x}$ is the number of units of type x on each SM and WarpSize is the number of threads in a warp, which is a characteristic of the GPU device. Since NVIDIA GPUs aggregate in a single counter the number of warps executing in the INT and FP32 units, the number of instructions of each type (Inst_INT and Inst_FP32) is used to decompose this metric into the utilizations of each separate unit. Similarly, the number of floating-point operations (single or double precision) of each type (ADD, MUL or FMA) are used to separate the utilizations of the FP32 and FP64 units.

On the other hand, the utilization rate of the different memory hierarchy levels is computed as follows:

$$\text{U}_\text{y} = \frac{\text{ABand}_\text{y}}{\text{PeakBand}_\text{y}}, \text{y} \in \{\text{L2}, \text{Shared}, \text{DRAM}\}, \quad (8)$$

where ABand and PeakBand are the achieved and peak bandwidth of each memory subsystem, respectively.

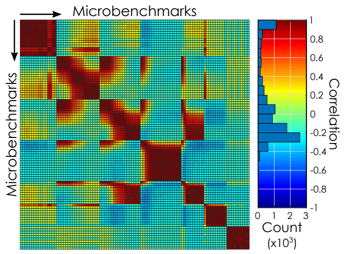
Accordingly, to model all the considered GPU components, the metrics summarized in Table 1 are required.

3.2 Microbenchmarking the GPU

To accurately model the unknown characteristics of the underlying architecture, the proposed methodology relies on microbenchmarking. By creating a set of carefully designed applications covering several GPU components, it is possible to get an accurate prediction on the contribution of each component to the total GPU power consumption.

```
GPU_Kernel {
  initialize registers;
  unrolled for loop {
    stress component Ci;
  }
  copy results to main memory;
}
```

(a) Example of the microbenchmarks code used to stress the GPU components.



(b) Correlation heatmap between the devised microbenchmarks and corresponding histogram (on a Titan Xp).

Fig. 3: Microbenchmark Suite.

TABLE 2: Developed microbenchmark suite to model the power consumption of the considered GPU components.

Name	Components		#	Name	Components		#
INT	Integer units		13	SF	SF units		10
FP32	32-bit FP units	ADD	4	FP64	64-bit FP units	ADD	4
		MUL	4			MUL	4
		FMA	4			FMA	4
L2	L2-cache		10	Shared	Shared memory		9
DRAM	DRAM		12	Mix	Mix of arithmetic Shared, L2 and DRAM		7
CF	Control-flow units		15	Idle	-		1

Fig. 3a presents an example of a skeleton source code of the developed microbenchmark GPU kernels. These microbenchmarks are mainly composed by an unrolled *for* loop which stresses the desired GPU component. By varying its loop boundaries, it is possible to achieve different mixtures of components utilizations. For example, in a given microbenchmark stressing the integer unit, each iteration of the loop executes arithmetic instructions on the registers data. By decreasing the number of loop iterations, the resulting arithmetic intensity (arithmetic instructions per amount of data read from main memory) decreases, which allows creating a range of microbenchmarks, *e.g.* from a more integer-intensive (high INT and low DRAM utilizations) to a more DRAM-intensive (low INT and high DRAM utilizations).

Table 2 presents a summary of the developed collection of 101 microbenchmarks used to estimate the proposed power model. The model considers 12 different GPU components. To better model the interactions between different instructions, the suite also includes microbenchmarks with different mixes of GPU components utilizations (MIX).

Fig. 3b shows a correlation heatmap of the utilization vectors for all microbenchmarks, as well as an histogram of the correlation values. As expected, microbenchmarks from the same group have a higher correlation value, as they are exercising the same component. However, the histogram shows that most microbenchmarks from different groups mostly have a low correlation (absolute value around 0.2).

3.3 Model parameter estimation

Estimating the DVFS-aware power model corresponds to the determination of the unknown coefficients $\mathbf{x} = [\,\alpha_0, \alpha_1, \alpha_2, \alpha_3,\ \gamma_{\text{mem}},\ \gamma_1,\ \ldots,\ \gamma_N\,]$. In order to facilitate

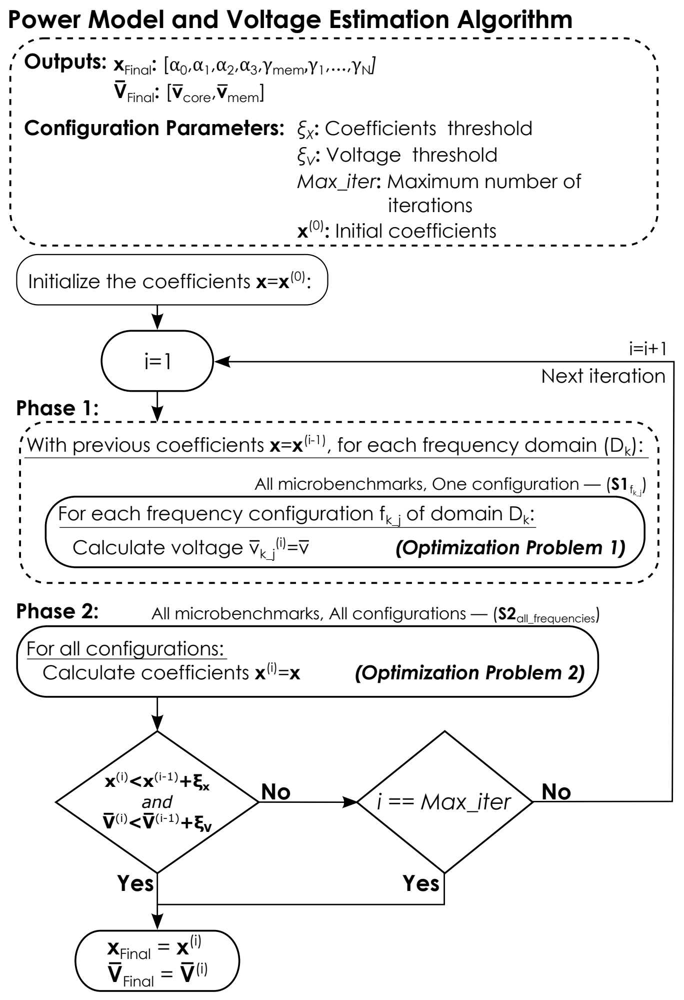

Power Model and Voltage Estimation Algorithm
Outputs: $\mathbf{x}_{Final}$: $[\alpha_0,\alpha_1,\alpha_2,\alpha_3,\gamma_{mem},\gamma_1,\dots,\gamma_N]$
$\bar{\mathbf{v}}_{Final}$: $[\bar{\mathbf{v}}_{core},\bar{\mathbf{v}}_{mem}]$
Configuration Parameters: ξ_X: Coefficients threshold
ξ_V: Voltage threshold
Max_iter: Maximum number of iterations
$\mathbf{x}^{(0)}$: Initial coefficients
Initialize the coefficients $\mathbf{x}=\mathbf{x}^{(0)}$:
i=1
i=i+1
Next iteration
Phase 1:
With previous coefficients $\mathbf{x}=\mathbf{x}^{(i-1)}$, for each frequency domain (D_k):
All microbenchmarks, One configuration — ($\mathbf{S1}_{f_{k\_j}}$)
For each frequency configuration f_{k_j} of domain D_k:
Calculate voltage $\bar{v}_{k\_j}^{(i)}=\bar{v}$
(Optimization Problem 1)
Phase 2:
All microbenchmarks, All configurations — ($\mathbf{S2}_{all\_frequencies}$)
For all configurations:
Calculate coefficients $\mathbf{x}^{(i)}=\mathbf{x}$
(Optimization Problem 2)
$\mathbf{x}^{(i)}<\mathbf{x}^{(i-1)}+\xi_x$ and $\bar{\mathbf{V}}^{(i)}<\bar{\mathbf{V}}^{(i-1)}+\xi_V$
No
i == Max_iter
No
Yes
Yes
$\mathbf{x}_{Final}=\mathbf{x}^{(i)}$
$\bar{\mathbf{V}}_{Final}=\bar{\mathbf{V}}^{(i)}$

Fig. 4: Devised algorithm to estimate the model coefficients ($\mathbf{x}$) and GPU voltage levels ($\bar{\mathbf{v}}_{\text{core}}$ and $\bar{\mathbf{v}}_{\text{mem}}$).

the presentation of the proposed methodology, this section considers a device consisting of 2 frequency domains, with 1 component in the memory domain and N components in the core domain (which is consistent with most GPU devices). Additionally, given that in some devices it is unknown how the voltage of each frequency domain scales with their operating frequency [8], the proposed methodology also estimates the vectors of voltages $\bar{\mathbf{v}}_{\text{core}}$ and $\bar{\mathbf{v}}_{\text{mem}}$ associated with each operating frequency.

In order to better understand the characteristics of each GPU component, the previously described set of developed microbenchmarks (Section 3.2) is used to stress these components. The microbenchmarks are executed and their power consumption is measured at each supported V-F configuration. Additionally, the value of the metrics (see Table 1) required to compute the component utilization rates is also measured for each microbenchmark. This set of measurements can then be used to estimate the unknown parameters of the proposed model.

Finally, Eqs. 5 and 6 show a relation between both the unknown voltages $\bar{\mathbf{v}}_{\text{k}}$ and coefficients α_i and γ_i (with $\text{k}\in\{\text{core},\text{mem}\}$). Therefore, a simple least squares regression cannot be used, as it leads to a non-full-rank optimization problem. Hence, an alternative iterative optimization

Optimization Problem 1. *Voltage of Domain* D_k ($\bar{\text{v}}$).
For each frequency level of domain D_k ($\text{f}_{\text{k\_j}}$, with $\text{k}\in\{\text{core},\text{mem}\}$, $\text{j}\in[\text{k\_min},\text{k\_max}]$), the voltage $\text{V}_{\text{k\_j}}$ is calculated by solving:

$$\begin{aligned}\underset{\bar{\text{v}}}{\arg\min}\quad & \sum_{\text{Microbench.}\in\mathbf{S}}\left(\text{P}_{\text{meas.}}-\hat{\text{P}}\right)^2\\ \text{subject to}\quad & \hat{\text{P}}=\text{P}_{\text{core}}+\text{P}_{\text{mem}},\\ & \mathbf{S}=\mathbf{S1}_{\text{f}_{\text{k\_j}}},\\ & \forall_{\text{f}_{\text{k\_1}}>\text{f}_{\text{k\_2}}}\ \bar{\text{v}}_{\text{k\_1}}\geq\bar{\text{v}}_{\text{k\_2}},\ \text{k}\in\{\text{core},\text{mem}\}\end{aligned}$$

Optimization Problem 2. *Model Coefficients* ($\mathbf{x}$).

$$\begin{aligned}\underset{\mathbf{x}}{\arg\min}\quad & \sum_{\text{Microbench.}\in\mathbf{S}}\left(\text{P}_{\text{meas.}}-\hat{\text{P}}\right)^2\\ \text{subject to}\quad & \hat{\text{P}}=\text{P}_{\text{core}}+\text{P}_{\text{mem}},\\ & \mathbf{S}=\mathbf{S2}_{\text{all\_frequencies}}\end{aligned}$$

Fig. 5: Optimization problems for the algorithm from Fig. 4.

algorithm was devised to estimate such parameters. Its operation is summarized in Fig. 4. The referred Optimization Problems 1 and 2 are described in Fig. 5.

The algorithm is composed of two main phases, corresponding to the estimation of the voltage levels and model coefficients. In phase 1, the model coefficients ($\mathbf{x}$) are fixed to the previously found values in order to compute the voltage levels of each frequency domain. Furthermore, the algorithm focuses on each domain separately, for example, by first fixing the memory voltages to a constant value and estimating the core voltages for each frequency level (*i.e.*, by solving Problem 1 for each core operating frequency). Afterwards the estimated core voltages are used to estimate the memory voltages associated with each memory configuration (*i.e.*, by solving Problem 1 for each memory operating frequency). In phase 2, all the estimated voltages are simultaneously used to estimate the model coefficients (*i.e.*, by solving Problem 2).

It is important to note that, unlike previous works [10], [12], the proposed methodology does not make any assumption on the scaling of the voltage with the frequency of each domain. Given the importance of the voltage in the power consumption of these types of devices (in both the static — Eq. 1 — and dynamic — Eq. 2 — power consumptions), it is important to have an informed knowledge of these values, in order to avoid a low prediction accuracy. However, in situations where the device voltage levels are known a priori, the proposed methodology can be simplified into a single execution of phase 2 (*i.e.*, by only solving Problem 2), utilizing the real voltage values.

3.4 Fixed Frequency Estimation

The proposed model can also be applied when the goal is to simply predict the power consumption for a fixed frequency configuration, *i.e.* in the same configuration where

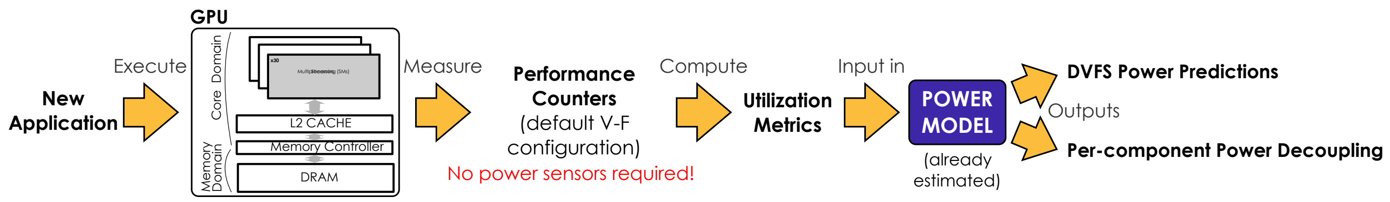


Fig. 6: Usage of a (previously estimated) power model to predict the power consumption of a new application.

the utilization levels are measured. In this simpler case, the model parameters are estimated with a fixed frequency (and voltage), and Eq. 4 can be rewritten as:

$$\mathrm{P}(\mathrm{D}_k) = \beta_0 + \sum_{i=1}^{\mathrm{N_C}} \omega_i \cdot \mathrm{U}_i, \tag{9}$$

where the values of the voltages and frequencies are now integrated inside the coefficients to be determined. Since there are no products between unknown parameters in this simplified model (in contrast with the previous formulation), parameter estimation can be done in a single step, by using least squares regression. This is done by executing phase 2 of the previous algorithm, *i.e.*, by solving Problem 2 using the measurements taken at the target frequency.

3.5 Power Consumption Prediction

Once the model coefficients are known, the newly determined GPU power model can be used to predict the power consumption of any (previously unseen) application, as it is presented in Fig. 6. The model allows predicting how the application power consumption changes over the whole range of the device V-F configurations, by simply measuring its performance events on a single configuration (without requiring any power measure). The model also allows decoupling the partial power consumption of the multiple modeled GPU components from the total power consumption (more on this in Section 4.5).

4 Power Model Use Cases

One feature of the proposed power model is its potential usefulness in multiple and diverse scenarios. This section presents five of these use cases, providing also a validation of the proposed model in real and modern hardware devices and with a set of commonly used standard benchmarks (not used during model estimation).

4.1 Experimental setup

To validate the proposed model, a collection of five GPUs from different NVIDIA microarchitectures were used as testing platforms (summarized in Table 3). All experiments were performed on a Linux CentOS 7.4 server, with CUDA 9.0 and NVIDIA driver v384.98.

The NVML [37] library was used for monitoring and for changing the operating frequencies of the GPU domains (the voltage is automatically set). Additionally, real power measurements are also obtained using NVML. To guarantee a proper model validation, a reasonable set of power samples are required during the kernel execution. Since the GPU power sensors have a low sampling frequency, the kernels were repeatedly executed whenever necessary, to ensure an execution time of at least 1 second at the fastest GPU configuration (highest core and memory frequencies). Finally, the power consumption of each kernel was computed as the average of all gathered samples. To guarantee the accuracy of the presented results, all applications were repeatedly executed, with the presented values corresponding to the median value.

To estimate the devised power model on each GPU device, the conceived microbenchmark suite (see Section 3.2) was executed on a wide range of different V-F configurations. By using the corresponding power consumption values (measured at all tested V-F configurations) and the performance events used to compute the metrics presented in Table 1 (measured only at the reference frequency configuration), it is possible to estimate all the model coefficients. For all GPU devices, the estimation algorithm (see Fig. 4)

TABLE 3: Summarized description of the used GPUs.

		Titan Xp	GTX Titan X	GTX 980	GTX 960	Tesla K40c
Base architecture		Pascal	Maxwell			Kepler
Compute capability		6.1	5.2			3.5
Memory frequencies (MHz)		{5705, 4705}*	{4005, 3505, 3300, 810}		3505	3004
Core freq. range (MHz)		[1911:582]	[1164:595]		1226	[875:666]
Default Mem. Frequency		5705	3505		3505	3004
Default Core Frequency		1404	975		1226	875
Number of SMs		30	24	16	8	15
Per SM	**SP/INT Units**	128	128	128	128	192
	DP Units	4	4	4	4	64
	SF Units	32	32	32	32	32
	Shared Memory (bytes)	96K	96K	96K	96K	{16,32 48}K
L2-Cache Size (bytes)		3M	3M	2M	1M	1.5M
L2-Cache Banks		12	12	8	4	6
Global Memory Size (bytes)		12G	12G	4G	4G	12G
Memory Bus Width (bits)		384	384	256	128	384
TDP (W)		250	250	165	120	235

* NVIDIA driver does not allow setting the memory frequency to lower levels.

TABLE 4: Standard benchmarks used for model validation.

Suite	Application Name
Parboil [15]	*CUTCP, LBM, MRI-Gridding*
Rodinia [16]	*Backprop, DWT2D, Gaussian, Hotspot, Hotspot3D, LUD, K-Means, K-Means_2, ParticleFilter_naive, ParticleFilter_float, SRAD_v1, SRAD_v2, Streamcluster*
Polybench [17]	*2MM, 3MM, 3DCONV, ATAX, BICG, CORR, COVAR, FDTD-2D, GEMM, GESUMMV, GRAMSCHM, MVT, SYRK, SYRK_DOUBLE*
SHOC [18]	*BFS, FFT, MD5Hash, Reduction, S3D, S3D_double, Sort, Stencil2D, Stencil2D_double, QTClustering*
CUDA SDK [19]	*Blackscholes, matrixMulCUBLAS*

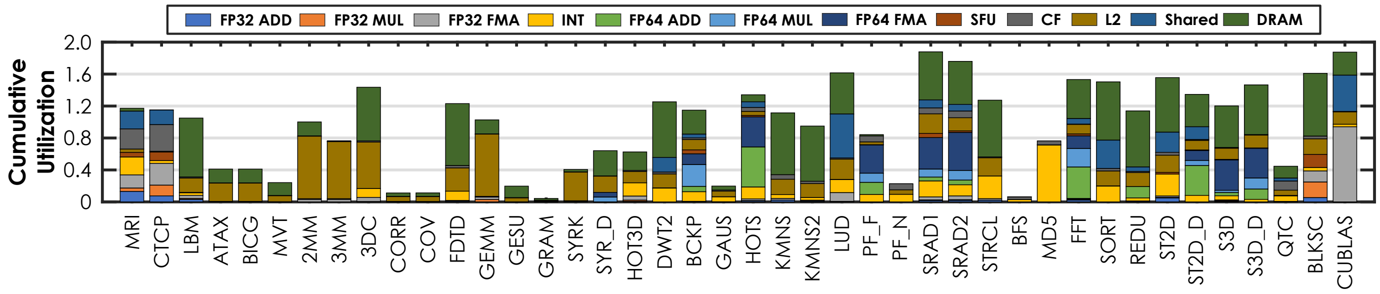


Fig. 7: Per-component utilizations for the set of 42 standard benchmarks on the Titan Xp.

converged in less than 100 iterations, corresponding to less than 5 minutes execution on an Intel i7 8550U processor.

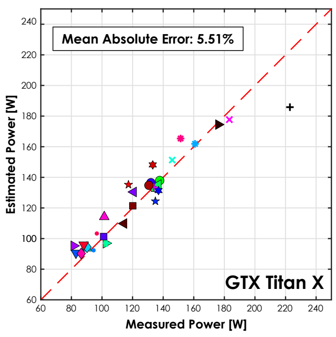
Model validation was performed using an independent collection of 42 applications from 5 benchmark suites (see Table 4), *i.e.* these benchmarks were not used during model estimation. Each application was executed at the reference frequency configuration to measure the hardware events required by the model. Fig. 7 presents the utilization rate of the GPU components for each benchmark, where the utilization of *each* component is in the $[0, 1]$ range. The obtained results show a wide diversity of different utilizations. In order to evaluate the accuracy of the power predictions provided by the devised model, the power consumption of each application at the different V-F configurations was also measured. Since these applications were not used to estimate the model parameters, they show the model robustness for new (unseen) applications.

The rest of this section will analyze the model by looking at the different scenarios in which it can be used: i) at fixed V-F configuration, therefore disregarding the influence of both voltage and frequency levels on the power consumption (Section 4.2); ii) for DVFS management, *i.e.* by providing predictions over the whole frequency range, which (in these cases) also include providing the voltage levels — as such values are unknown (Section 4.3); iii) using a single power sample to improve the DVFS model accuracy over the whole frequency range (using scaling-factors — Section 4.4); iv) providing a per-component breakdown of the GPU power consumption (Section 4.5); and v) performing power predictions on a different GPU than the one the model was estimated on (Section 4.6).

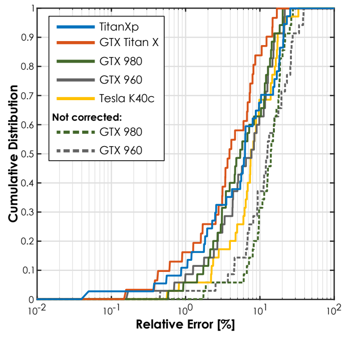
4.2 Power Prediction at a Fixed Frequency

As previously discussed, the proposed power consumption model can be used to predict the power consumption of applications at the same frequency configuration that the performance counters are measured. Fig. 8a presents such a scenario, with the values of the predicted and measured power consumptions for each benchmark executed on the GTX Titan X GPU. The obtained results show the accuracy of the proposed power model for the set of standard benchmarks (used only to validate the model).

Fig. 8b presents the obtained results for the five considered GPUs at their respective reference (default) frequency configuration, namely the cumulative prediction errors on each GPU. For the GTX 980 and GTX 960 GPUs, two curves are presented corresponding to the obtained results



(a) Per-benchmark accuracy on GTX Titan X.



(b) Cumulative prediction errors on the evaluated GPU devices.

Fig. 8: Power prediction at a fixed V-F configuration on the standard benchmarks (not used in the model estimation).

with and without the correction factor after exporting an estimated model to a different GPU (more details on this in Section 4.6). The results show that the power model is able to accurately predict the GPUs power consumption, where a mean absolute error of 5.5% was achieved on the GTX Titan X, while on the other GPUs the errors were 8.8% (Titan Xp), 7.7% (GTX 980), 8.5% (GTX 960) and 7.1% (Tesla K40c).

4.3 DVFS-Aware Power Prediction

Unlike previous ones, the proposed model assumes that both V_{core} and V_{mem} can scale with the frequency changes of the two GPU domains. However, while the estimated core voltages were possible to be confirmed, by using the measured voltages obtained using the NVIDIA Inspector and MSI Afterburner (third-party Windows tools), the voltage of the memory domain cannot be measured using these tools.

From the obtained measurements, it was observed that the voltage scaling behaviour depends on the method used to scale the domains frequency. When using the NVML library (on the Titan Xp, GTX Titan X and Tesla K40c GPUs), the voltage variation presents two different regions [14]: for higher frequencies the voltage scales linearly, while for lower frequencies it stays constant. In some GPUs NVML does not allow changing the frequency (*e.g.*, for non-Titan or non-Tesla GPUs). In these cases, the domains frequencies were changed by varying the graphics clock and memory transfer rate offsets of the *Powermizer* inside the *nvidia-settings* tool. However, experimental results showed that this alternate method does not result in the same behaviour as

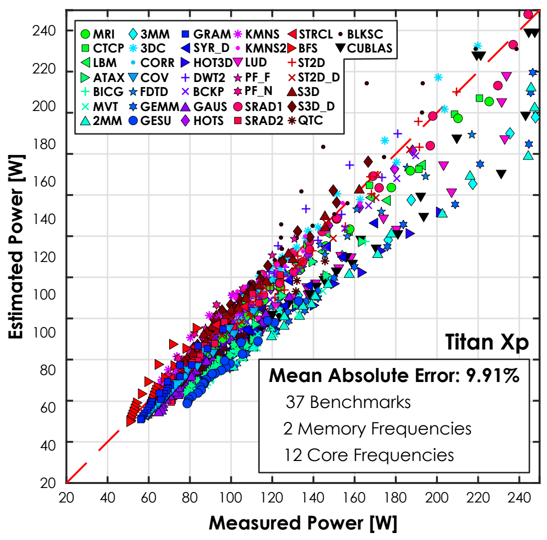


Fig. 9: DVFS power predictions, across all V-F configurations, of the standard benchmarks (not used in the model estimation), on the Titan Xp.

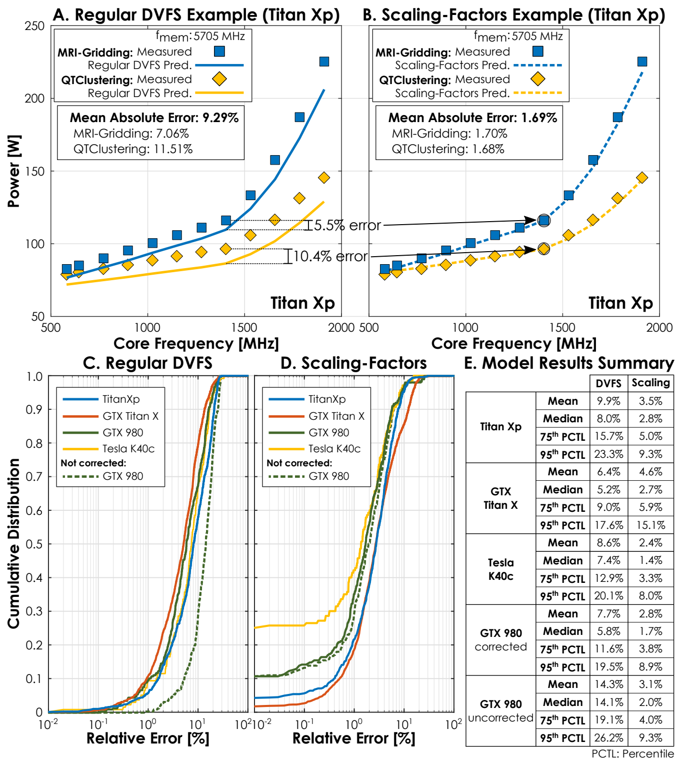
in this case the voltage stays constant across all frequencies.

Fig. 9 presents the accuracy of the proposed DVFS power model when considering the validation set of standard benchmarks, for multiple core and memory V-F configurations on the GTX Titan X GPU. This figure presents the power estimated by the model versus the measured power consumptions for the considered benchmarks (not used to estimate the model), across 2 memory and 12 core frequencies. The power predictions are based on the performance counters measured, for each application, at a single (reference) V-F setting. The diverse set of applications and frequency configurations results in a wide range of observed power consumptions, going from 50W up to 250W. Hence, when covering a frequency range up to 4× for the core frequencies and 1.2× for the memory frequencies, the model showed to be still able to make accurate power predictions, with a mean absolute error of 9.9%.

Fig. 10C presents the obtained results across the four considered GPU devices (GTX960 was not considered in this section, because it does not allow to change frequency configurations). Overall, the devised DVFS-aware power model results in mean absolute errors of 9.9% (Titan Xp), 6.4% (GTX Titan X), 7.7% (GTX 980) and 8.6% (Tesla K40c). The approach presented in [12] proposed a DVFS power consumption model for NVIDIA GPUs, achieving a mean error of 23.5% for the Kepler GPU (same as the Tesla K40c).

4.4 DVFS Predictions with Scaling-Factors

To produce power estimations, the proposed power model always requires at least one execution of the application under evaluation to measure the components utilizations. Leveraging from this fact, whenever the device has power sensors available, the GPU power consumption can also be measured during the application execution. Once the power consumption at a certain V-F configuration is known, the power model can be used to determine how the consump-



		DVFS	Scaling
Titan Xp	Mean	9.9%	3.5%
	Median	8.0%	2.8%
	75th PCTL	15.7%	5.0%
	95th PCTL	23.3%	9.3%
GTX Titan X	Mean	6.4%	4.6%
	Median	5.2%	2.7%
	75th PCTL	9.0%	5.9%
	95th PCTL	17.6%	15.1%
Tesla K40c	Mean	8.6%	2.4%
	Median	7.4%	1.4%
	75th PCTL	12.9%	3.3%
	95th PCTL	20.1%	8.0%
GTX 980 corrected	Mean	7.7%	2.8%
	Median	5.8%	1.7%
	75th PCTL	11.6%	3.8%
	95th PCTL	19.5%	8.9%
GTX 980 uncorrected	Mean	14.3%	3.1%
	Median	14.1%	2.0%
	75th PCTL	19.1%	4.0%
	95th PCTL	26.2%	9.3%

PCTL: Percentile

Fig. 10: Top: Example of frequency scaling model benefits on two benchmarks, on Titan Xp. Bottom: Results on the standard benchmarks, across the different V-F configurations: Cumulative distribution of the prediction errors of the regular DVFS (C.) and scaling-factors (D.); and summary of the obtained results (E.).

tion will scale for different V-F variations (*i.e.*, the different *scaling-factors*, as referred in [13]), in the following way:

$$\hat{\mathrm{P}}^{(\mathrm{scaling})}(\mathbf{f}_2, \mathbf{v}_2) = \frac{\mathrm{P}_{\mathrm{GPU}}(\mathbf{f}_2, \mathbf{v}_2)}{\mathrm{P}_{\mathrm{GPU}}(\mathbf{f}_1, \mathbf{v}_1)} \cdot \mathrm{P}_{\mathrm{meas.}}(\mathbf{f}_1, \mathbf{v}_1), \qquad (10)$$

where $\mathbf{f}_1$ and $\mathbf{v}_1$ are the frequencies and voltages at the reference configuration, and $\mathrm{P}_{\mathrm{GPU}}(\mathbf{f}_1, \mathbf{v}_1)$ and $\mathrm{P}_{\mathrm{GPU}}(\mathbf{f}_2, \mathbf{v}_2)$ are the power consumptions given by the regular DVFS model (Eqs. 5 and 6) at configurations $(\mathbf{f}_1, \mathbf{v}_1)$ and $(\mathbf{f}_2, \mathbf{v}_2)$, respectively. Finally, $\mathrm{P}_{\mathrm{meas.}}(\mathbf{f}_1, \mathbf{v}_1)$ is the measured power consumption at the reference configuration and $\hat{\mathrm{P}}^{(\mathrm{scaling})}(\mathbf{f}_2, \mathbf{v}_2)$ is the new estimate for the power consumption at configuration $(\mathbf{f}_2, \mathbf{v}_2)$.

Since this method is already using a measured value of the power consumption (offsetting the curve to a known value), it will result in a much higher accuracy of the DVFS power predictions. Figs. 10A and 10B present an example of the benefits of using the scaling-factors approach on two distinct applications on the Titan Xp GPU. Fig. 10A presents the measured and estimated power consumptions using the regular DVFS approach (presented in Section 4.3), resulting in a mean absolute prediction error of 9.3%. However, in both cases, the error is almost constant across the different V-F configurations, *i.e.* the model accurately predicts how the power scales with the frequency and voltage of the two domains. This is confirmed in the results achieved using the scaling-factors method, presented in Fig. 10B. In this case, the measured power sample allows sliding the

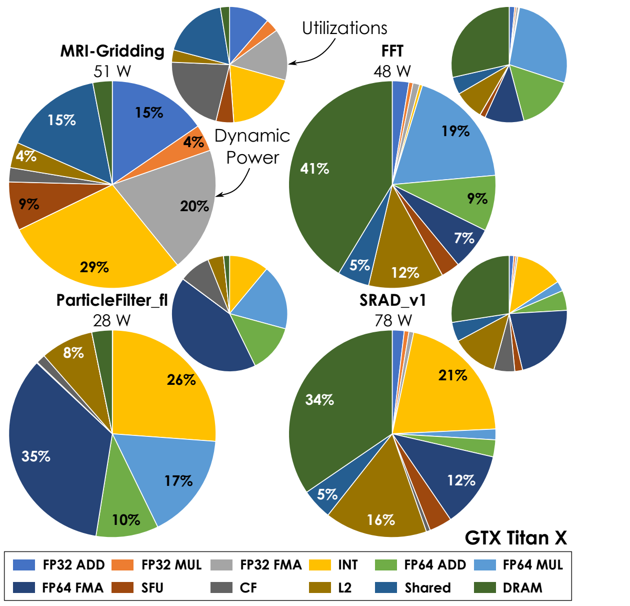


Fig. 11: Dynamic power consumption breakdown for each modeled GPU component on GTX Titan X. The values below benchmark names correspond to the dynamic power at the reference V-F, where constant power is predicted to be 87 W.

estimated curve closer to the measure values, resulting in much smaller prediction errors (1.7%).

The overall results of the scaling-factors approach on the considered GPU devices are presented in Fig. 10D. Since the model is using the actual samples for the reference configuration, it will have an error of 0% at those configurations (hence the curves are not asymptomatically tending to 0, when the relative error goes to $-\infty$). From these results it can be seen that around 20% (Titan Xp), 18% (GTX Titan X), 30% (GTX 980) and 40% (Tesla K40c) of the estimated values have an error below 1%. On the other hand, 95% (Titan Xp), 85% (GTX Titan X), 95% (GTX 980) and 97% (Tesla K40c) of the estimated values have an error below 10%, while using the regular DVFS approach these values would be 60% (Titan Xp), 80% (GTX Titan X), 70% (GTX 980) and 70% (Tesla K40c). The results are summarised in Fig. 10E. Comparing with previous state-of-the-art works, the approach presented in [13] proposed a scaling-factor power consumption model (using a neural network classifier) and achieved a mean error of 10%.

4.5 GPU Power Decoupling

Once the power model is fully determined, it can be used to estimate not only the total GPU power consumption, but also the power of each component during the execution of any application. This power breakdown can be very useful for the application optimization, as it provides crucial information to the developers regarding which components represent the main power consumption bottleneck.

Fig. 11 presents a breakdown of the dynamic power consumption at the reference V-F configuration for each modeled component of the GTX Titan X GPU, for four distinct benchmarks (large pies). The relative utilizations of the GPU components for each of these applications are also

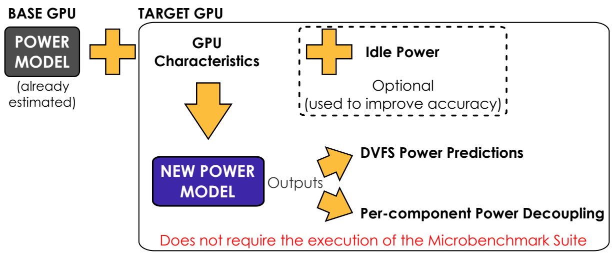


Fig. 12: Diagram displaying how a model estimated for a GPU can be used on a different GPU.

presented in the figure (small pie). As one would expect, these two representations produce different results. This is because of the different estimated weights of each GPU component in the power model (γ_i in Eqs. 5 and 6), which make some components to have a more dominant contribution to the power consumption than others. For example, in the *MRI-Gridding* application, the SFU has a utilization of only 5% of the total application execution. However, as it is shown in Fig. 11, the SFU contributes to 9% of the GPU dynamic power consumption. Its contribution is actually higher than the contribution of the CF units, which have a much larger utilization. Naturally, this result is coherent with the complexity of each unit, as the SF performs much more complex operations than the CF unit.

The difference between the two representations of Fig. 11 confirms the usefulness of the proposed model: providing the means for alternative power optimization strategies regarding the conventional performance only approach.

4.6 Exporting the Model to Different GPUs

Given the modular design of NVIDIA GPUs (as it can be observed from Table 3), a power model estimated for a specific GPU could potentially be adapted and applied to a different GPU. To attain this objective, one must take into account the effects of the architectural changes in the power consumption of GPU applications. This can be particularly useful in providing power estimations for GPU devices without power sensors. Additionally, this approach (summarized in Fig. 12) would also avoid the need for the execution of the whole microbenchmark suite on the target GPU device. For now, this work focused on extrapolating the model only within GPUs of the same microarchitecture, namely, for the three Maxwell GPUs (see Table 3). Despite that, a similar approach could be used to provide power predictions for GPUs of different microarchitectures, which could be even useful in the design stages of future microarchitectures.

As it is summarized in Table 3, the GTX Titan X, GTX 980 and GTX 960 have 30, 24 and 16 SMs, respectively. However, the internal architecture of each SM is the same. Therefore, it is reasonable to assume that the peak power consumption associated with each modeled internal component (FP32, FP64, INT, *etc.*) will scale by the same factor. Hence, for the target GPUs, the model coefficients associated with each modeled GPU component are obtained by scaling the estimated coefficients (from the base GPU) in the same way that the corresponding components are scaling between the two GPU devices. For example, between the GTX Titan X and GTX 980 GPUs, the number of SMs is decreased by

TABLE 5: Summary of proposed model results.

Source	Model Type	Titan Xp	GTX Titan X	GTX 980	GTX 960	Tesla K40c
HPCA [14]	DVFS	11.7%	6.75%	-	-	9.07%
Proposed	Fixed	8.75%	5.51%	14.2% (7.66%)†	15.4% (8.47%)†	7.09%
	DVFS	9.91%	6.43%	14.3% (7.67%)†	-‡	8.60%
	Scaling	3.54%	4.55%	3.07% (2.80%)†	-‡	2.39%

† Porting model trained on GTX Titan X. Errors without (with) constant power correction.
‡ The GTX 960 only allows 1 frequency configuration.

2/3. Therefore, the coefficients associated with the SMs units (γ_i in Eq. 5) will be 2/3 of the coefficient value from the GTX Titan X. Regarding the memory hierarchy, it can be seen that from Table 3 L2-Cache and DRAM components are actually scaling in the same proportion as the number of SMs. Therefore, their coefficients (γ_{L2} and γ_{mem}) will also scale by the same factor (naturally, on other GPUs the scalings for the memory and SM components may differ).

Regarding the estimation of the new coefficients associated with the static and constant power of the V-F configurations ($\alpha_0, \ldots, \alpha_3$ in Eqs. 5 and 6) two possible approaches were tested. The first one assumes the coefficients also scale with the same factors as the components of each domain (*e.g.*, for GTX 980 α_0 and α_1 would be 2/3 of the values in GTX Titan X, *etc.*). However, this approach ignores the possibility that between the two GPUs there are probably multiple (not modeled) components that do not change (*e.g.*, PCIe interface, GigaThread Engine, *etc.*) and therefore their power consumption remains the same. Therefore, this straightforward approach will generally under-predict the power consumption on the target GPUs, resulting in prediction errors greater than the ones obtained on the source GPU. In particular, for the fixed-frequency model, the mean absolute errors obtained for the two GPUs were 14.2% (GTX 980) and 15.4% (GTX 960). For the regular DVFS model on the GTX 980, a mean absolute error of 14.3% was obtained.

The second approach requires the measurement of the idle power at the reference configuration and therefore requires the existence of GPU power sensors. Using the measured idle power consumptions, the values of the new coefficients can be obtained in the following way:

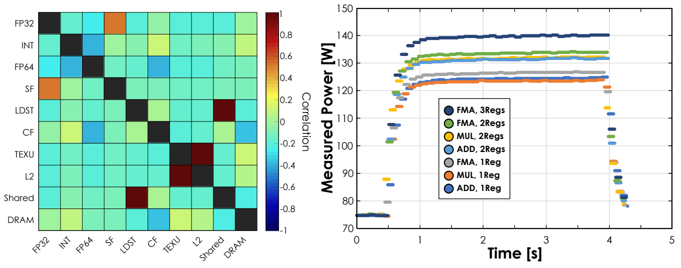
$$\alpha_i^{(\text{target})} = \alpha_i^{(\text{base})} \cdot \frac{\mathrm{P}_{\text{idle}}^{(\text{target})}}{\mathrm{P}_{\text{idle}}^{(\text{base})}}, i \in \{0,1,2,3\}. \tag{11}$$

The measured values for the idle power consumption at the reference V-F configuration of each GPU device are 75W (GTX Titan X), 65W (GTX 980) and 36W (GTX 960). As it was previously mentioned, it can be seen that the idle power values at the target GPUs are greater than those obtained by assuming it simply scales with the number of SMs (*e.g.*, $75\text{W} \times 2/3 = 50\text{W} < 65\text{W}$).

By using this approach to determine the corrected coefficients, a much higher prediction accuracy is obtained, where the fixed frequency prediction model has a mean absolute errors of 7.7% (GTX 980) and 8.5% (GTX 960). The DVFS model also has an error of 7.7% on the GTX 980 GPU.

5 Discussion

The presented research represents an improvement and a substantial extension over our previous work presented in [14], where a DVFS power model was proposed, together with a microbenchmark suite that allowed the modeling of seven GPU components. The herein presented work proposes three different variations of a GPU power model: 1) fixed frequency, 2) DVFS and 3) DVFS with scaling-factors. All three include the modeling of twelve GPU components, allowing the models to provide either the total or per-component GPU power consumption. Furthermore, this extended work also presented a successful technique to export power models estimated on a GPU device to GPUs with different hardware configurations.



(a) Correlation heatmap of multiple GPU components utilization rates on Titan Xp.

(b) Different power consumption of different variations of 32-bit FP instructions on GTX 980.

Fig. 13: Model limitations.

The newly proposed model was validated on five different GPU devices (from three different NVIDIA microarchitectures) with a set of 42 benchmarks. The results of the different model usage scenarios are summarized in Table 5, where the results obtained from the models trained in [14] are also provided (applied to the new set of benchmarks). It is important to recall that some of the model features herein proposed were not supported in [14] (*e.g.*, scaling-factors model or portability of an estimated model). It can be seen that even in a fair comparison scenario, *i.e.* comparing just the regular DVFS power model between the two works, the herein proposed model outperforms the former one on all GPU devices. Furthermore, these predictions can even be further improved on all devices by using the scaling-factors model, which was not considered in the previous work.

5.1 Model Limitations

Despite achieving considerable improvements over the work presented in [14], a few roadblocks were reached while trying to develop the herein presented work, some of which would equally limit the quality of any supervised regression/machine-learning based power model. In particular, one factor that can limit the quality of the proposed model is the accuracy of the GPU performance counters (or power samples). Notwithstanding, as it was presented the proposed model is still able to achieve very accurate power predictions on a diverse range of GPU devices.

On the other hand, by allowing to directly choose which GPU components can be modeled, the proposed model arises as in a more hardware-centric approach. However, it becomes very important to consider the possibility for multicollinearity between utilization rates of the chosen components [38]. Fig. 13a presents an example of the observed correlation between the utilization rates of 10 GPU components during the execution of the microbenchmark suite (Section 3.2) on the Titan Xp GPU (for simplicity, this figure aggregates the ADD, MUL and FMA components in the FP32 and FP64 units). It can be seen that there is a very high correlation (≈ 0.99) between the utilizations of the Load/Store (LDST) unit and the shared memory. Similarly, there is also a high correlation between the utilizations of the texture units and L2 cache. A similar behaviour was observed on the GTX Titan X and Tesla K40c devices. For this reason, the LDST units and Texture units were not considered as separate units in the herein presented model, since their inclusion would result in a decrease of the model prediction accuracy. In fact, the high correlation between their utilizations and the utilization of an already modeled unit, means the proposed regression-based model is already partially accounting for their power consumptions.

Another limitation arises from the fact that the same PTX instruction can have different power consumptions depending on the used operands. For example, a FP32 ADD instruction of the form *Rc=Ra+Ra* has a different power consumption than one of the form *Rc=Ra+Rb*. This difference can be seen in Fig. 13b, where it is presented the GPU power consumption over the time for seven different 32-bit FP instructions (using different number of register operands) on the GTX 980 GPU. The encountered problem is the fact that these different power consumptions are impossible to take into account in the proposed power model, as there are no performance counters that give insights on these differences during the applications execution (*i.e.*, how to distinguish between FMA-3Regs, FMA-2Regs and FMA-1Reg). Similarly, there are also no counters that allow distinguishing between different types of integer instructions (ADD, MUL or MAD), which were observed to also have different power consumptions. Without being able to identify these differences, they are impossible to model and therefore there will always be a baseline prediction in accuracy which cannot be reduced. However, it is important to note that these different power consumptions could be easily included in the model if NVIDIA introduced new performance counters that allowed identifying these different invocations of similar instructions types.

6 Conclusions

This work presented a GPU power consumption model that can be used to predict the GPU power consumption during the execution of any application (and at any voltage and frequency configuration). To model the GPU, a novel estimation algorithm is presented, which relies on careful benchmarking of the GPU architecture. This algorithm not only is able to estimate the contribution of twelve different GPU components (FP32-ADD/MUL/FMA, FP64-ADD/MUL/FMA, INT, SF, CF units, shared memory, L2-cache and DRAM) to the total power consumption, but it also allows to determine how the voltage of each separate GPU domain scales with its corresponding frequency.

Three different model use cases were proposed (fixed frequency, DVFS and scaling-factors) to obtain the total or per-component GPU power consumption, as well as a way to export models to a distinct GPU device than the one it was estimated on. Each of these approaches was extensively validated on five different GPU devices from the three most recent GPU microarchitectures (Pascal, Maxwell and Kepler) with a set of 42 benchmarks from five commonly used benchmark suites. In particular, the scaling-factor power model provided very accurate power predictions, with an average error of 3.5%, 4.6%, 3.1% and 2.4% for the Titan Xp, GTX Titan X, GTX 980 and Tesla K40c GPUs, respectively.

Acknowledgments

This work was partially supported by national funds through Fundação para a Ciência e a Tecnologia (FCT), under grant SFRH/BD/101457/2014, and projects UID/CEC/50021/2019, PTDC/EEI-HAC/30485/2017 and PTDC/CCI-COM/31901/2017.

References

[1] Top500, “TOP500, Supercomputer Site.” [Online]. Available: https://www.top500.org/lists/2018/11/

[2] X. Mei, L. S. Yung, K. Zhao, and X. Chu, “A measurement study of GPU DVFS on energy conservation,” in *Proc. Workshop Power-Aware Comput. Syst. (HotPower)*, 2013.

[3] R. Ge, R. Vogt, J. Majumder, A. Alam, M. Burtscher, and Z. Zong, “Effects of Dynamic Voltage and Frequency Scaling on a K20 GPU,” in *Proc. Int. Conf. Parallel Process. (ICPP)*, 2013.

[4] J. Guerreiro, A. Ilic, N. Roma, and P. Tomas, “DVFS-aware application classification to improve GPGPUs energy efficiency,” *Parallel Computing*, 2018.

[5] S. Zhuravlev, S. Blagodurov, and A. Fedorova, “Addressing shared resource contention in multicore processors via scheduling,” in *Proc. Int. Conf. Archit. Support Program. Lang. Oper. Syst. (ASPLOS)*, 2010.

[6] S. Che, J. W. Sheaffer, M. Boyer, L. G. Szafaryn, Liang Wang, and K. Skadron, “A characterization of the Rodinia benchmark suite with comparison to contemporary CMP workloads,” in *Proc. Int. Symp. Workload Characterization (IISWC)*, 2010.

[7] J. Guerreiro, A. Ilic, N. Roma, and P. Tomás, “Performance and Power-Aware Classification for Frequency Scaling of GPGPU Applications,” in *Proc. Workshop Algorithms, Model. Tools Parallel Comput. Heterog. Platforms (HeteroPar)*, 2016.

[8] X. Mei, Q. Wang, and X. Chu, “A survey and measurement study of GPU DVFS on energy conservation,” *Digital Communications and Networks*, vol. 3, no. 2, pp. 89–100, may 2017.

[9] S. Hong and H. Kim, “An integrated GPU power and performance model,” in *Proc. Int. Symp. Comput. Archit. (ISCA)*, 2010.

[10] J. Leng, T. Hetherington, A. ElTantawy, S. Gilani, N. S. Kim, T. M. Aamodt, and V. J. Reddi, “GPUWattch: enabling energy optimizations in GPGPUs,” in *Proc. Int. Symp. Comput. Archit. (ISCA)*, 2013.

[11] S. Song, C. Su, B. Rountree, and K. W. Cameron, “A Simplified and Accurate Model of Power-Performance Efficiency on Emergent GPU Architectures,” in *Proc. Int. Symp. Parallel Distrib. Process. (IPDPS)*, 2013.

[12] Y. Abe, H. Sasaki, S. Kato, K. Inoue, M. Edahiro, and M. Peres, “Power and Performance Characterization and Modeling of GPU-Accelerated Systems,” in *Proc. Int. Parallel Distrib. Process. Symp. (IPDPS)*, 2014.

[13] G. Wu, J. L. Greathouse, A. Lyashevsky, N. Jayasena, and D. Chiou, “GPGPU performance and power estimation using machine learning,” in *Proc. Int. Symp. High Perform. Comput. Archit. (HPCA)*, 2015.

[14] J. Guerreiro, A. Ilic, N. Roma, and P. Tomas, “GPGPU Power Modeling for Multi-domain Voltage-Frequency Scaling,” in *Proc. Int. Symp. High Perform. Comput. Archit. (HPCA)*, 2018.

[15] J. A. Stratton, C. Rodrigues, I.-J. Sung, N. Obeid, L.-W. Chang, N. Anssari, D. Geng, W.-M. Liu, and W. Hwu, “Parboil: A Revised Benchmark Suite for Scientific and Commercial Throughput Computing,” *Center for Reliable and High-Performance Computing*, vol. 127, 2012.

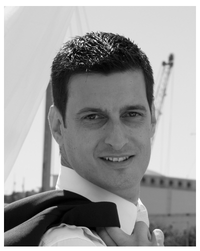
[16] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron, “Rodinia: A benchmark suite for heterogeneous computing,” in *Proc. Int. Symp. Workload Characterization (IISWC)*, 2009.

[17] L.-N. Pouchet, "Polybench: The polyhedral benchmark suite," 2012. [Online]. Available: http://www.cs.ucla.edu/{~}pouchet/software/polybench/
[18] A. Danalis, G. Marin, C. McCurdy, J. S. Meredith, P. C. Roth, K. Spafford, V. Tipparaju, and J. S. Vetter, "The scalable heterogeneous computing (SHOC) benchmark suite," in *Proc. Workshop General-Purpose Computation Graph. Process. Units (GPGPU)*, 2010.
[19] NVIDIA, "GPU Computing SDK," 2017. [Online]. Available: https://developer.nvidia.com/cuda-code-samples
[20] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Proc. Int. Symp. Low Power Electron. Des. (ISLPED)*, 2007.
[21] K. Ma, X. Li, W. Chen, C. Zhang, and X. Wang, "GreenGPU: A Holistic Approach to Energy Efficiency in GPU-CPU Heterogeneous Architectures," in *Proc. Int. Conf. Parallel Process. (ICPP)*, 2012.
[22] R. A. Bridges, N. Imam, and T. M. Mintz, "Understanding GPU Power: A Survey of Profiling, Modeling, and Simulation Methods," *ACM Comput. Surv. (CSUR)*, vol. 49, no. 3, p. 41, 2016.
[23] C. Isci and M. Martonosi, "Runtime power monitoring in high-end processors: methodology and empirical data," in *Proc. Int. Symp. Microarchitecture (MICRO)*, 2003.
[24] J. Butts and G. Sohi, "A static power model for architects," in *Proc. Int. Symp. Microarchitecture (MICRO)*, 2000.
[25] R. Gonzalez, B. M. Gordon, and M. A. Horowitz, "Supply and threshold voltage scaling for low power CMOS," *IEEE J. Solid-State Circuits (SSC)*, vol. 32, no. 8, pp. 1210–1216, aug 1997.
[26] X. Ma, M. Dong, L. Zhong, and Z. Deng, "Statistical power consumption analysis and modeling for GPU-based computing," in *Proc. Workshop Power-Aware Comput. Syst. (HotPower)*, 2009.
[27] J. Chen, Bin Li, Ying Zhang, L. Peng, and J.-k. Peir, "Statistical GPU power analysis using tree-based methods," in *Proc. Int. Green Comput. Conf. and Workshops (IGCC)*, 2011.
[28] V. Adhinarayanan, B. Subramaniam, and W.-C. Feng, "Online Power Estimation of Graphics Processing Units," in *Proc. Int. Symp. Clust. Cloud Grid Comput. (CCGrid)*, 2016.
[29] H. Nagasaka, N. Maruyama, A. Nukada, T. Endo, and S. Matsuoka, "Statistical power modeling of GPU kernels using performance counters," in *Proc. Int. Green Comput. Conf. and Workshops (IGCC)*, 2010.
[30] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt, "Analyzing CUDA workloads using a detailed GPU simulator," in *Proc. Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, 2009.
[31] R. Nath and D. Tullsen, "The CRISP performance model for dynamic voltage and frequency scaling in a GPGPU," in *Proc. Int. Symp. Microarchitecture (MICRO)*, 2015.
[32] A. Majumdar, L. Piga, I. Paul, J. L. Greathouse, W. Huang, and D. H. Albonesi, "Dynamic GPGPU Power Management Using Adaptive Model Predictive Control," in *Proc. Int. Symp. High Perform. Comput. Archit. (HPCA)*, 2017.
[33] J. Guerreiro, A. Ilic, N. Roma, and P. Tomas, "Multi-kernel Auto-Tuning on GPUs: Performance and Energy-Aware Optimization," in *Proc. Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. (PDP)*, 2015.
[34] J. Haj-Yihia, A. Yasin, Y. B. Asher, and A. Mendelson, "Fine-Grain Power Breakdown of Modern Out-of-Order Cores and Its Implications on Skylake-Based Systems," *ACM Trans. Archit. Code Optim. (TACO)*, vol. 13, no. 4, pp. 1–25, dec 2016.
[35] NVIDIA and Microsoft, "Graphics Accelerated Productivity For Every User, Any Application," 2016. [Online]. Available: http://images.nvidia.com/content/grid/pdf/microsoft-server-solution.pdf
[36] D. Hackenberg, R. Schone, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer, "An Energy Efficiency Feature Survey of the Intel Haswell Processor," in *Proc. Int. Parallel Distrib. Process. Symp. Workshop (IPDPSW)*, 2015.
[37] NVIDIA, "NVML API Reference Guide vR384," 2017. [Online]. Available: http://docs.nvidia.com/deploy/nvml-api
[38] M. J. Walker, S. Diestelhorst, A. Hansson, A. K. Das, S. Yang, B. M. Al-Hashimi, and G. V. Merrett, "Accurate and Stable Run-Time Power Modeling for Mobile and Embedded CPUs," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. (CAD)*, vol. 36, no. 1, pp. 106–119, jan 2017.

João Guerreiro received the MSc degree in Electrical and Computer Engineering from the Instituto Superior Técnico (IST), Universidade de Lisboa, Lisbon, Portugal in 2014. He is a Junior Researcher at the Instituto de Engenharia de Sistemas e Computadores R&D (INESC-ID) where he is working toward the PhD degree. His research interests include parallel computing, graphics processors and energy-efficiency. He is a student member of IEEE.

Aleksandar Ilic received his Ph.D. degree in electrical and computer engineering from Instituto Superior Tecnico (IST), Universidade de Lisboa, Portugal, in 2014. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering of IST and a Senior Researcher of the Signal Processing Systems Group (SiPS), Instituto de Engenharia de Sistemas e Computadores R&D (INESC-ID). His research interests include high-performance and energy-efficient computing and modeling on parallel heterogeneous systems. He contributed to more than 40 papers to international journals and conferences and served in the organization of several international scientific events.

Nuno Roma received the Ph.D. degree in electrical and computer engineering from Instituto Superior Técnico (IST), Universidade Técnica de Lisboa, Lisbon, Portugal, in 2008. Currently, he is an Assistant Professor with the Department of Electrical and Computer Engineering of IST and a Senior Researcher of the Signal Processing Systems Group (SiPS) of Instituto de Engenharia de Sistemas e Computadores R&D (INESC-ID). His research interests include computer architectures, specialized and dedicated structures for digital signal processing, energy-aware computing, parallel processing and high-performance computing systems. He contributed to more than 100 manuscripts to journals and international conferences and served as a Guest Editor of Springer Journal of Real-Time Image Processing (JRTIP) and of EURASIP Journal on Embedded Systems (JES). He has also acted as the organizing chair of several workshops and special sessions. He has a consolidated experience on funded research projects leadership (principal investigator) and he is member of several research Networks of Excellence (NoE), including HiPEAC (European Network of Excellence on High Performance and Embedded Architecture and Compilation). Dr. Roma is a Senior Member of the IEEE Circuits and Systems Society and a member of ACM.



Pedro Tomás (S'04, M'09, SM'18) received the five-year licentiate, MSc and Ph.D. degrees in electrical and computer engineering from Instituto Superior Tecnico (IST), Technical University of Lisbon, Portugal, in 2003, 2006, and 2009, respectively. He is currently an assistant professor in the Department of Electrical and Computer Engineering (DEEC), IST, and a senior researcher at Instituto de Engenharia de Sistemas e Computadores R&D (INESC-ID). His research activities include computer microarchitectures, specialized computational structures, and high-performance computing. He is also interested in artificial intelligence models and algorithms. He is a member of the IEEE Computer Society and has contributed to more than 60 papers to international peer-reviewed journals and conferences.