A macroscopic analysis of GPU power consumption

Dorra Boughzala , Laurent Lefevre, Anne-Cécile Orgerie

LIP, ENS Lyon, INRIA

dorra.boughzala@ens-lyon.fr

COMPAS 2019 - Anglet

June 27, 2019

The Exascale, where are we ?

- Summit, the fastest supercomputer in the world in the Top 500 [1] list of June 2019.
- Summit has 4,608 nodes. Each node is composed of 2 IBM Power9 CPUs and 6 NVIDIA Volta GPUs.
- HPL performance (11/18 vs. 06/19) = **143.500 Pflops/s** vs. **148.600 Pflops/s**.
- Power consumption (11/18 vs. 06/19) = **9.783 MW** vs. **10.096 MW**.
- Energy Efficiency (11/18 vs. 06/19) = **14.668 Gflops/watt** vs. **14.719 Gflops/watt**.

[1] https://www.top500.org

Energy efficiency, an HPC concern ?

Fact 1

To achieve sustainable exascale performance, we should provide approximately 3-fold increase in energy efficiency.

Fact 2

Heterogeneous computing has become the most dominant paradigm in HPC systems. Among accelerators, GPUs have emerged as ideal co-processors.

Fact 3

GPU-based systems still consume a lot of energy.

Before modeling, we need analysis !

- Few works focus on analyzing the power profiling and sharing it within their power modeling methodology.

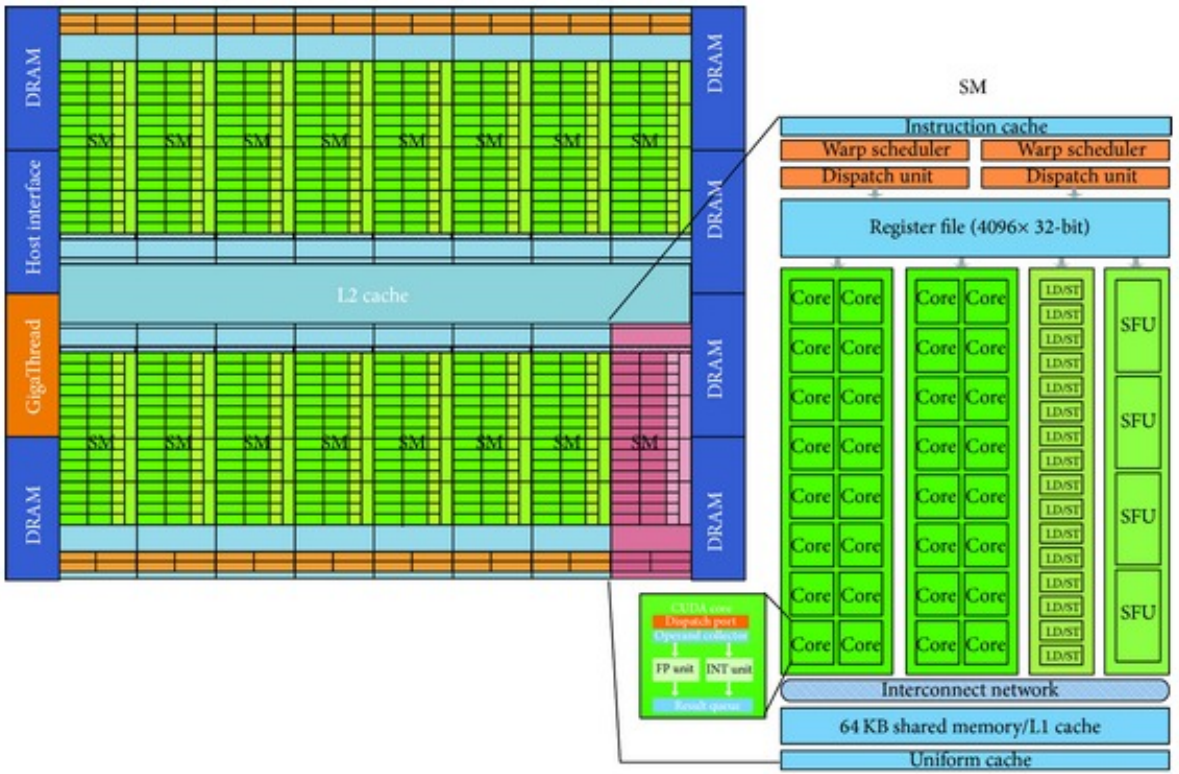
Related works	**Main approach**
Rofouei et al. (08)	GPU-CPU versus a CPU-only system
S.Collange et al. (09)	Power characterization of GPU functional blocks
J.M Cebri'n et al (12)	Discovery of resource under utilization
M.Burtscheret al. (14)	Analysis of K20 power samples

Table: Related works on GPU power anaylsis

- In this paper, we propose an analysis of the GPU performance and power consumption through the benchmarking of a representative kernel under different case studies.

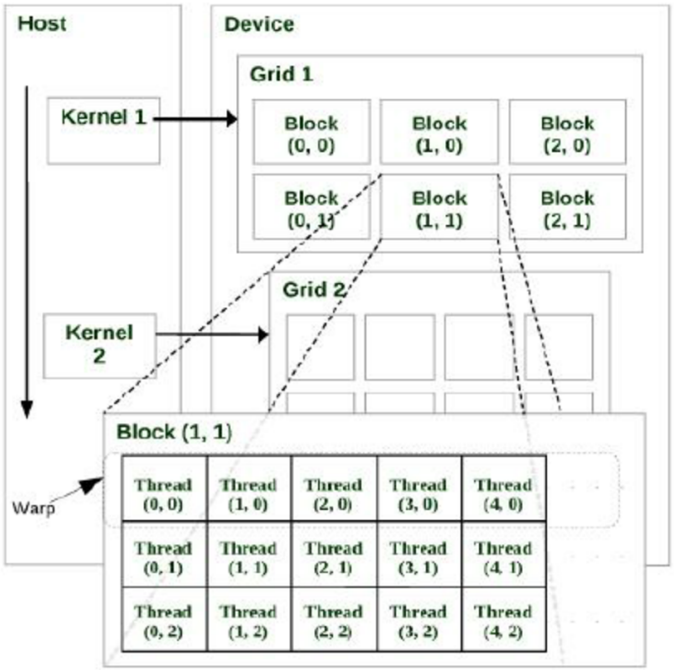
What about the GPU architecture ?

- (NVIDIA) GPU architecture = Streaming Multiprocessors (SMs) + Memory
- Fermi architecture:



All thanks go to CUDA !

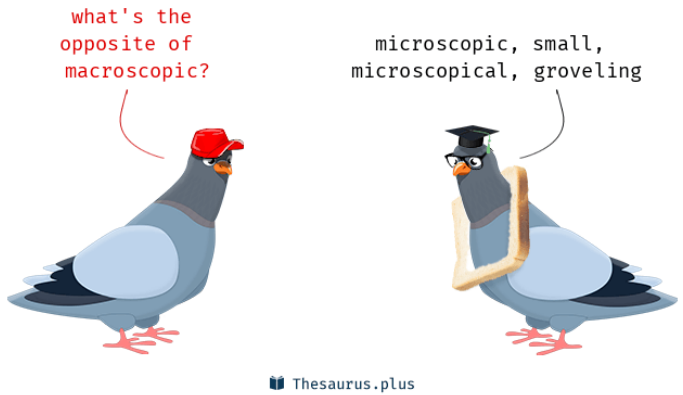
- CUDA (Compute Unified Device Architecture) is **both** the platform and the programming model built by NVIDIA for developing applications on NVIDIA GPUs cards.
- CUDA exposes an abstract view of the GPU parallel architecture: for mapping the parallelism within an application to the hardware.



Outline

Outline

What does "macroscopic" mean ? and Why ?



- Previous works rely on "highly detailed-architecture" approach = our opposite of "macroscopic".
- The macroscopic approach does not consider the study of GPU functional blocks behavior.
- Instead, we seek to understand by characterizing the performance and power cost of the GPU usage under different case studies.

So how ?

We propose 3 case studies in order to characterize the impact of varying some parameters on the execution time and the power consumption of our GPU:

- Case study 1: we study the impact of varying the data size.
- Case study 2: we study the impact of varying number of threads per block.
- Case study 3: we study the impact of varying number of blocks and active SMs.

Outline

What we used !

- In our work, we rely on the Grid'5000 infrastructure [2].
- Orion node: 2 CPUs + NVIDIA Tesla M2075 + wattmeter.
- Our GPU is based on the Fermi architecture (CUDA Driver v8.0).

Resources	Constraints
Number of SMs	14
Number of cores/SM	32
Number of cores	448
Global memory size	5301 Mbytes
Shared memory/Block	49152 bytes
Memory bus width	384-bit
Registers/Block	32768
Max Number of threads/SM	1536
Max number of threads/block	1024
Compute capability	2.0

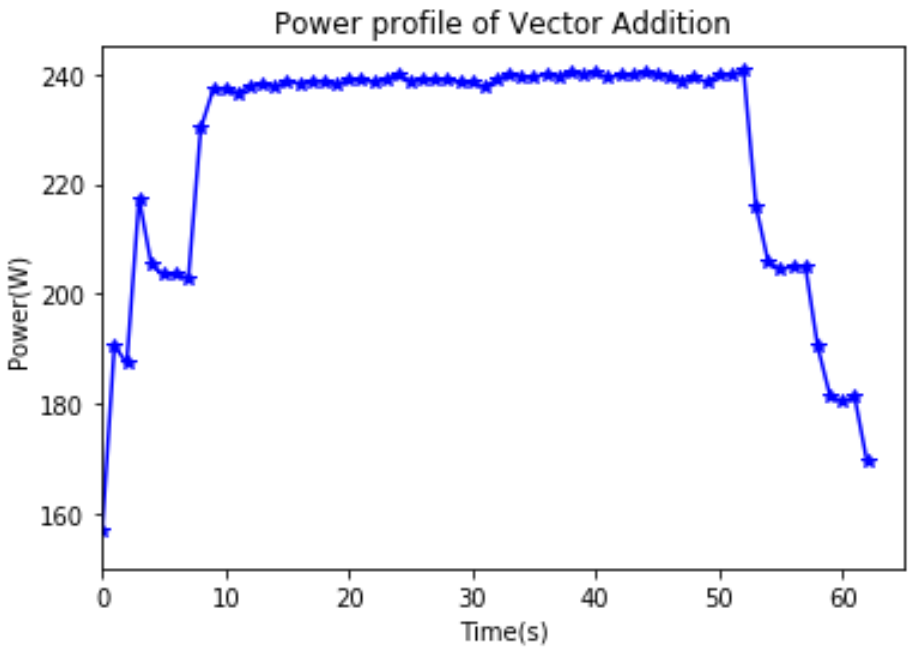
Table: NVIDIA Tesla M2075 resources description

[2] https://www.grid5000.fr

Outline

Results and analysis

- Idle power of the node (CPU + GPU) ~ 156 W.
- Idle GPU power ~ 57W
- Vector Addition (CUDA SDK): memory-bound kernel.



Vector Addition: Execution flow

- Allocates arrays in the CPU memory **(Malloc)**
- Initiates them with random floats.
- Allocates arrays in the GPU memory **(cudaMalloc)**
- Copies those arrays from the CPU memory to the GPU memory **(CopyC2G)**
- Launches the kernel by the CPU to be executed on the GPU **(VectAdd)**
- Copies the result from the GPU memory to the CPU memory **(CopyG2C)**
- Frees arrays from the GPU memory **(CudaFree)**
- Frees arrays from the CPU memory **(Free)**

Case study 1: Data size impact

- We use 1024 threads/block.
- We vary the data size from 500,000 to 50,000,000.

	Malloc(C)	Malloc(G)	CopyC2G	AddVec(G)	CopyG2C	Free(G)	Free(C)
N1=500000	0.01	2218.67	1.66	23880.14	1.73	0.28	0.38
N2=5000000	0.01	2216.51	12.76	235425.28	14.85	0.34	2.59
N3=50000000	0.01	2216.95	123.75	2368287.0	144.21	0.61	22.92

Table: Execution Time characterization in milliseconds.

	Malloc(C)	Malloc(G)	CopyC2G	AddVec(G)	CopyG2C	Free(G)	Free(C)
N1=500000	-	37.9	-	146.9	-	-	-
N2=5000000	-	37.2	-	146.5	-	-	-
N3=50000000	-	41.2	-	146,7	-	-	-

Table: Dynamic Power Consumption characterization in Average Watt.

	N1	N2	N3
Energy(kJ)	3.50	34.48	347.42

Table: Dynamic energy consumption in kJ

Threads per block impact

- We use N=5,000,000.
- We vary the number of threads per block from 128 to 1024.

	Malloc(C)	Malloc(G)	CopyC2G	AddVec(G)	CopyG2C	Free(G)	Free(C)
T/B= 128	0.01	2217.81	12.80	234380.4	14.60	0.35	2.43
T/B= 256	0.01	2214.74	12.76	233362.03	14.82	0.36	2.76
T/B= 512	0.01	2214.60	12.77	242002.4	14.64	0.32	2.53
T/B= 1024	0.01	2214.65	12.92	235320.53	14.28	0.34	2.59

Table: Execution Time characterization in milliseconds.

	Malloc(C)	Malloc(G)	CopyC2G	AddVec(G)	CopyG2C	Free(G)	Free(C)
T/B= 128	-	44.3	-	141.6	-	-	
T/B= 256	-	31.5	-	150	-	-	-
T/B= 512	-	39.8	-	153.3	-	-	-
T/B= 1024	-	39.4	-	142	-	-	-

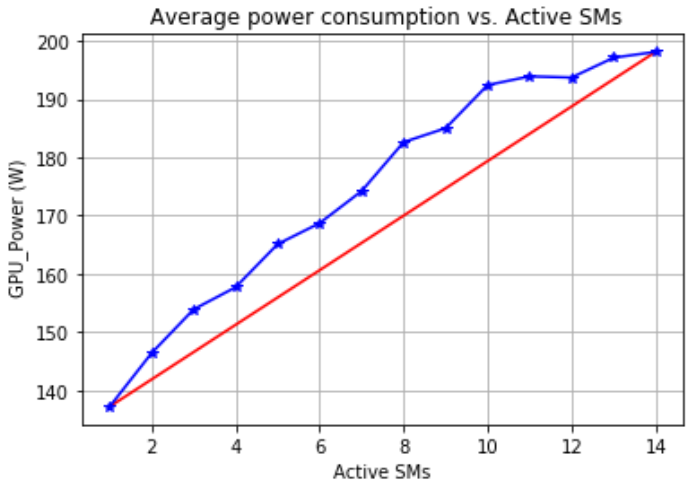
Table: Dynamic Power consumption characterization in Average Watt.

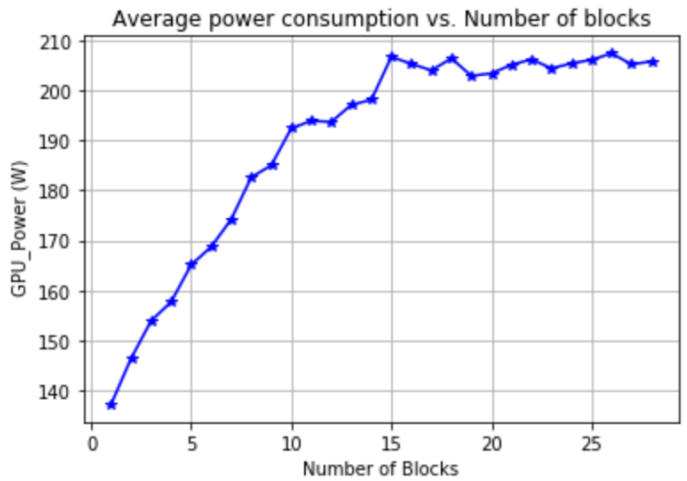
	T/B=128	T/B=256	T/B=512	T/B=1024
Energy(kJ)	33.18	35.00	37.09	33.41

Table: Dynamic energy consumption in kJ

Active SMs and number of blocks impact

- We vary the number of blocks, such that one block can be executed in each SM = we vary the data size.





Highlights

#1

Varying data size does not have an impact on the power consumption of the GPU (kernel execution).

#2

From the energy side, it's important to choose properly the number of threads/block.

#3

The power consumption increases linearly with the number of active SMs. But once we have more blocks than SMs, the power consumption becomes constant.

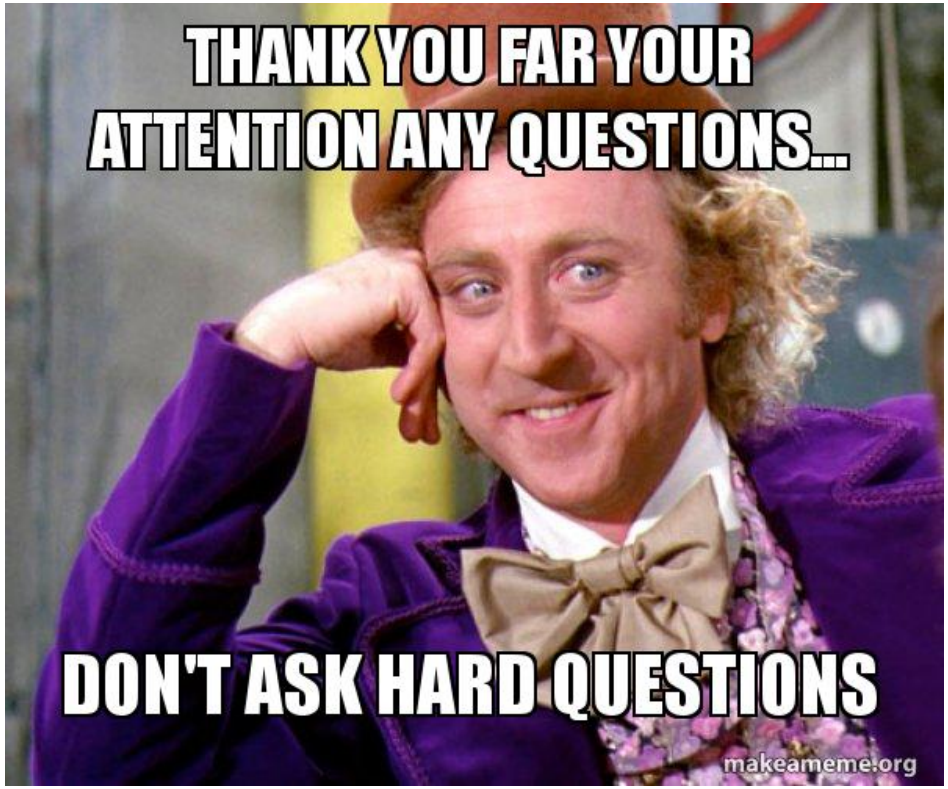
Outline

Conclusion and discussions

- Exploring those parameters helps us to understand more the behavior of GPU in terms of execution time and power consumption.
- Highlight #3 is very insightful and helpful for our modeling process.
- **What I am doing now :** implementing the GPU performance and energy consumption models in SimGrid [3].

[3] https://simgrid.org/

Questions ?


THANK YOU FAR YOUR
ATTENTION ANY QUESTIONS...
DON'T ASK HARD QUESTIONS
makeameme.org