Big Data on Heterogeneous Systems with GPUs

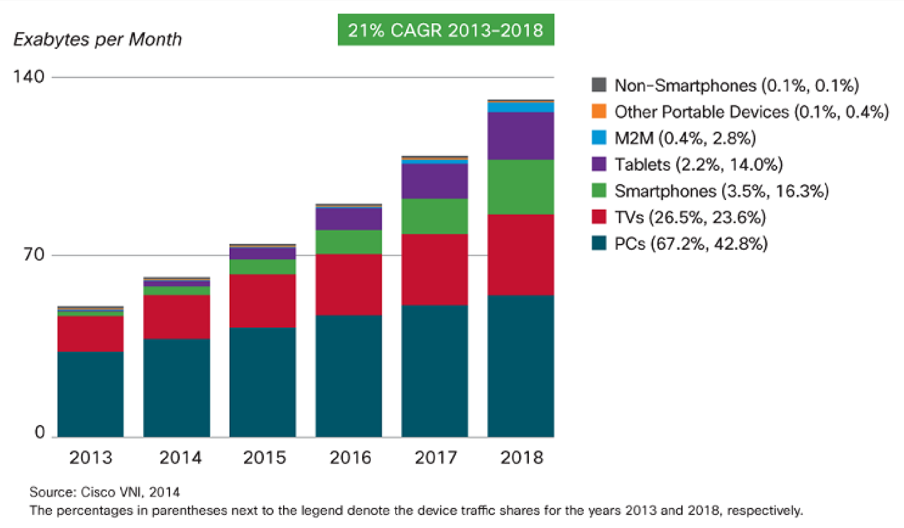
GPU Technology Workshop
South East Asia 2014

Dumitrel Loghin, Lavanya Ramapantulu,
Oana Barbu, Yong Meng Teo
[dumitrel,lavanya,oanabarb,teoym]@comp.nus.edu.sg
Department of Computer Science
National University of Singapore

1 Motivation

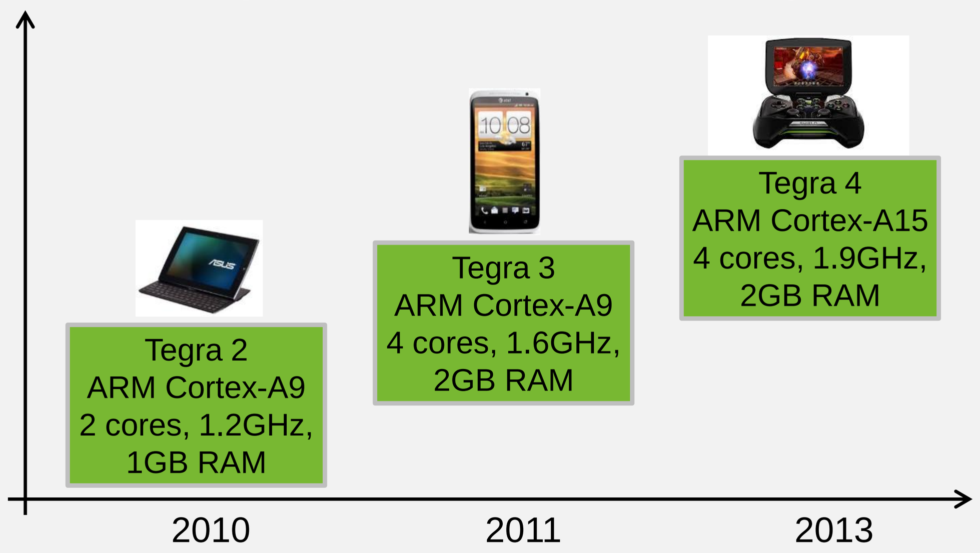
Big Data is everywhere

Source: IBM, Understanding Big Data, 2012

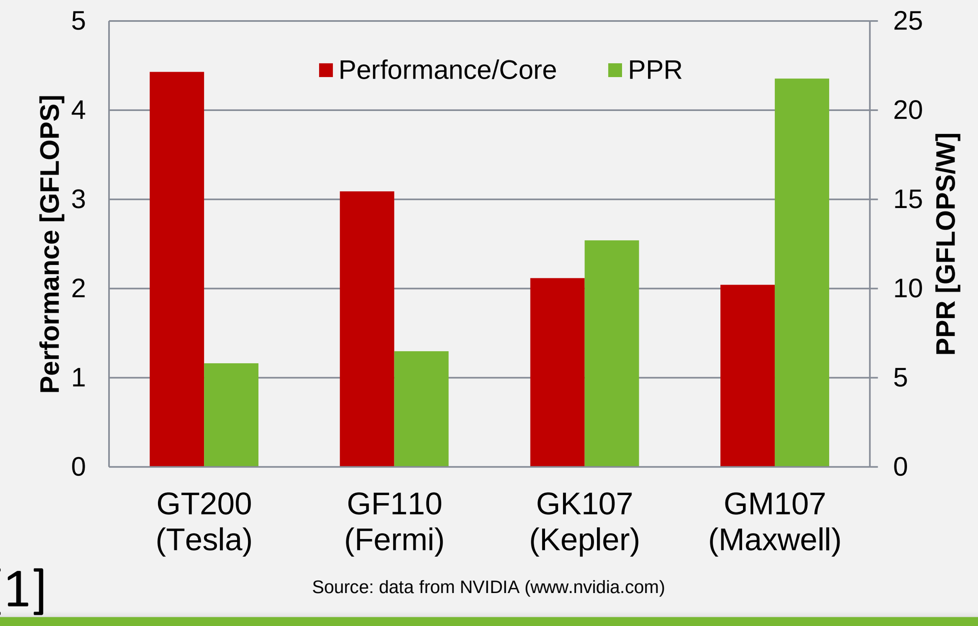


Source: Cisco VNI, 2014

Wimpy systems are evolving



NVIDIA GPUs become energy-efficient



Source: data from NVIDIA (www.nvidia.com)

Challenge: execute scale-out workloads on heterogeneous systems [1]

2 Objective and Approach

Objective

Investigates the **efficiency** of executing Big Data workloads on heterogeneous systems with **NVIDIA GPUs**.

Efficiency
- Execution time
- Energy
  - Performance-to-power Ratio (**PPR**)
  - Energy-delay Product (**EDP**)

Approach

- Big Data MapReduce applications in **CUDA** running on **Hadoop** through pipes mechanism using a *lazy processing* method for <key, value> pairs
- **Measurement**-driven analysis on the following four platform configurations:

1. Brawny node only (Intel Core i7)
2. Wimpy node only (NVIDIA Tegra 3)
3. Brawny node with GPU (i7 + GTX 750 Ti)
4. Wimpy node with GPU (Tegra 3 + GTX 750 Ti)

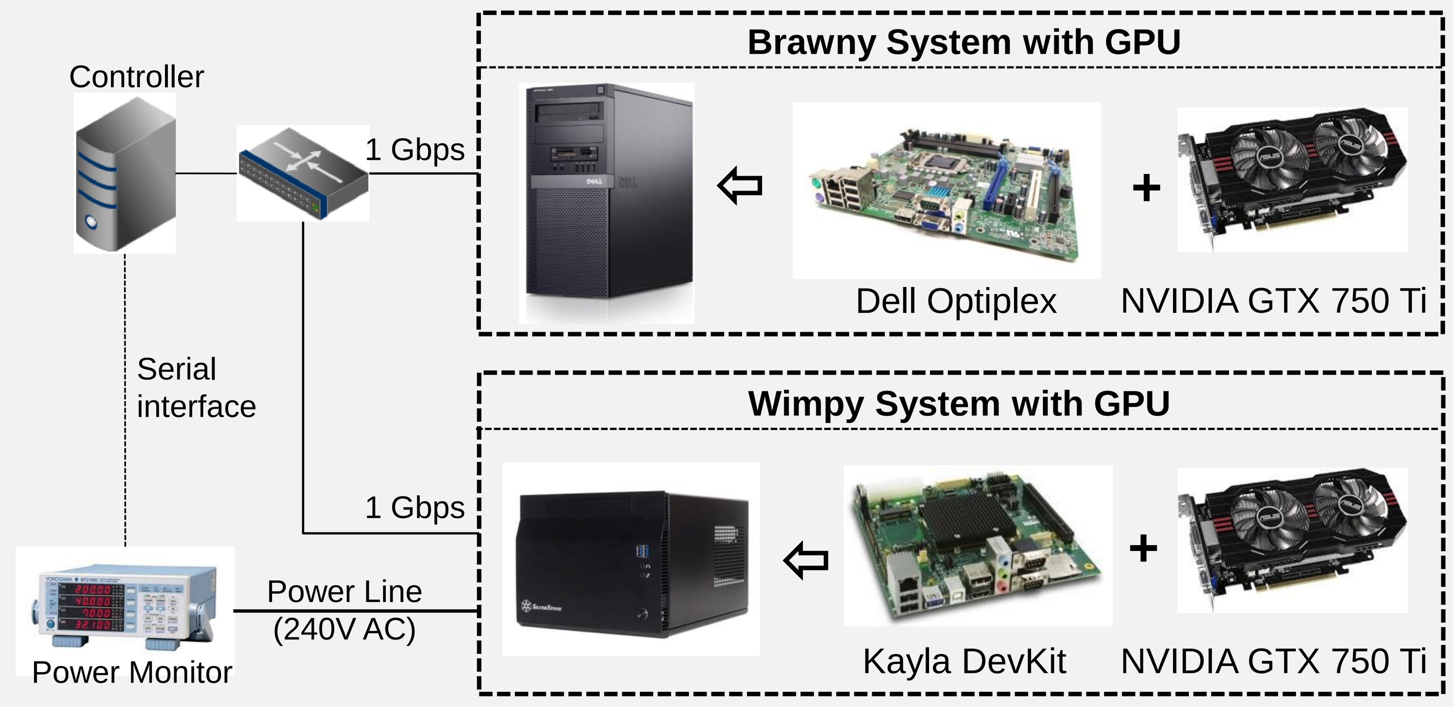
Frameworks and Workloads

Frameworks: Hadoop 1.2.1, CUDA Toolkit 6.0

Workload: K-means clustering of ***n*** points with ***m*** features into ***k*** clusters.

Input size	n	m	k	File size [GB]
Small (S)	3474500	34	5	0.32
Medium (M)	83397420	34	5	8.00
Large (L)	208493550	34	5	20.00

Setup



GPU

NVIDIA GeForce GTX 750 Ti	
Architecture	Maxwell
CUDA cores	640
CUDA compute capability	5.0
Streaming Multiprocessors (SM)	5
Active Blocks / SM	32
GFLOPS/s	1305.6
Total Memory	2GB GDDR5
Register File Size / SM	256 KB
Shared Memory / SM	64 KB
L2 Cache Size	2 MB
Memory bandwidth	86.4 GB/s
Thermal Design Power (TDP)	60 W

Systems

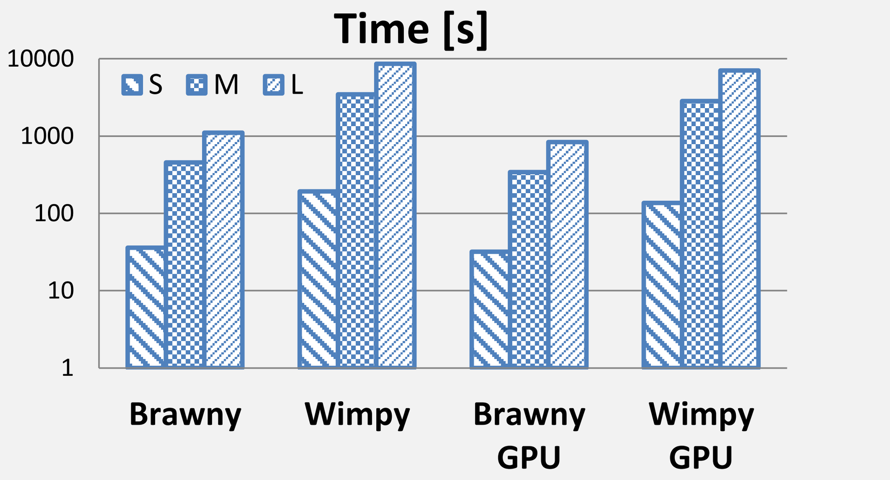
Specs		Dell Optiplex 990 (Brawny)	SECO Kayla DevKit (Wimpy)
CPU	Type	Intel Core i7-2600	NVIDIA Tegra 3 ARM Cortex-A9
	ISA	x86-64	ARMv7
	# of cores	4 (8 threads)	4
	Frequency	1.60 - 3.40 GHz	0.05 - 1.40 GHz
	Cache	32kB L1, 256kB L2, 8MB L3	32kB L1, 1MB L2
Memory		8GB DDR3	2GB LPDDR2
Network		Gigabit Ethernet	
Storage port		SATA 3.0	SATA 2.0
Storage device		512GB SSD (Crucial m4)	
OS		Linux 3.11.0	Linux 3.1.10-carma
C/C++ compiler		gcc 4.8.1	gcc 4.6.3
Java		jdk1.8.0	jdk1.8.0

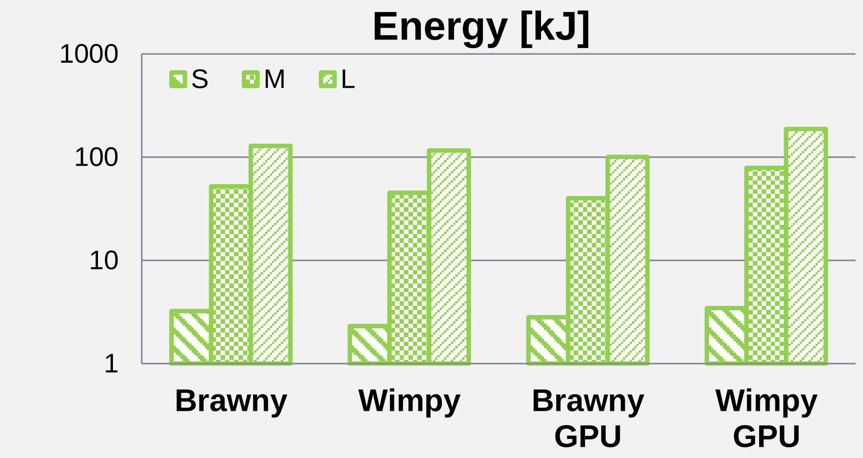
4 Preliminary Results

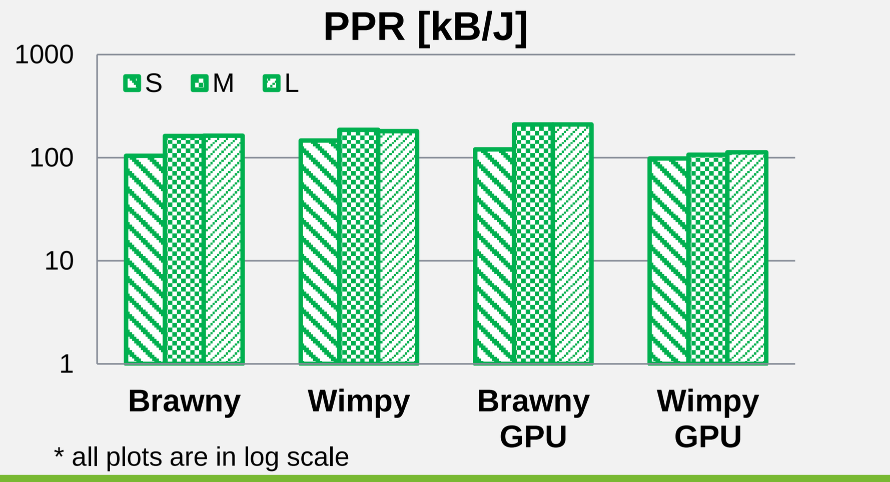
Hadoop-CUDA Kmeans(S) execution time [s]

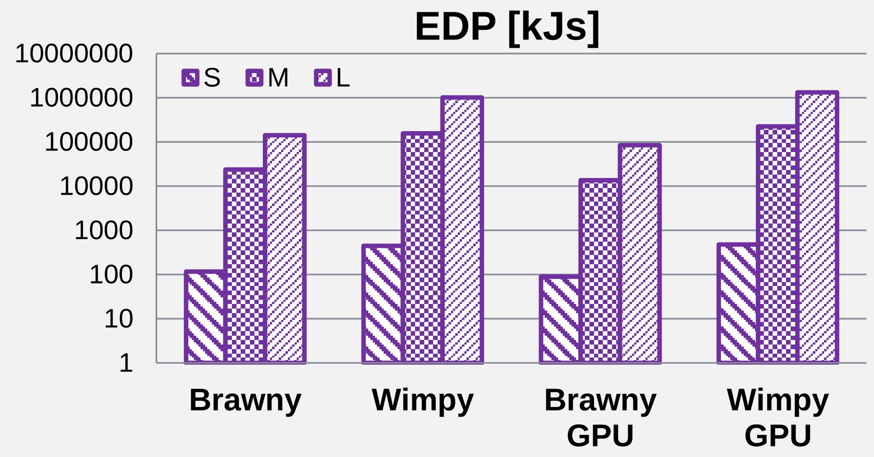
Brawny + GPU			Wimpy + GPU		
Naïve approach	Lazy processing	Speedup	Naïve approach	Lazy processing	Speedup
311	32	9.7	934	137	6.8

Results for Kmeans with lazy processing



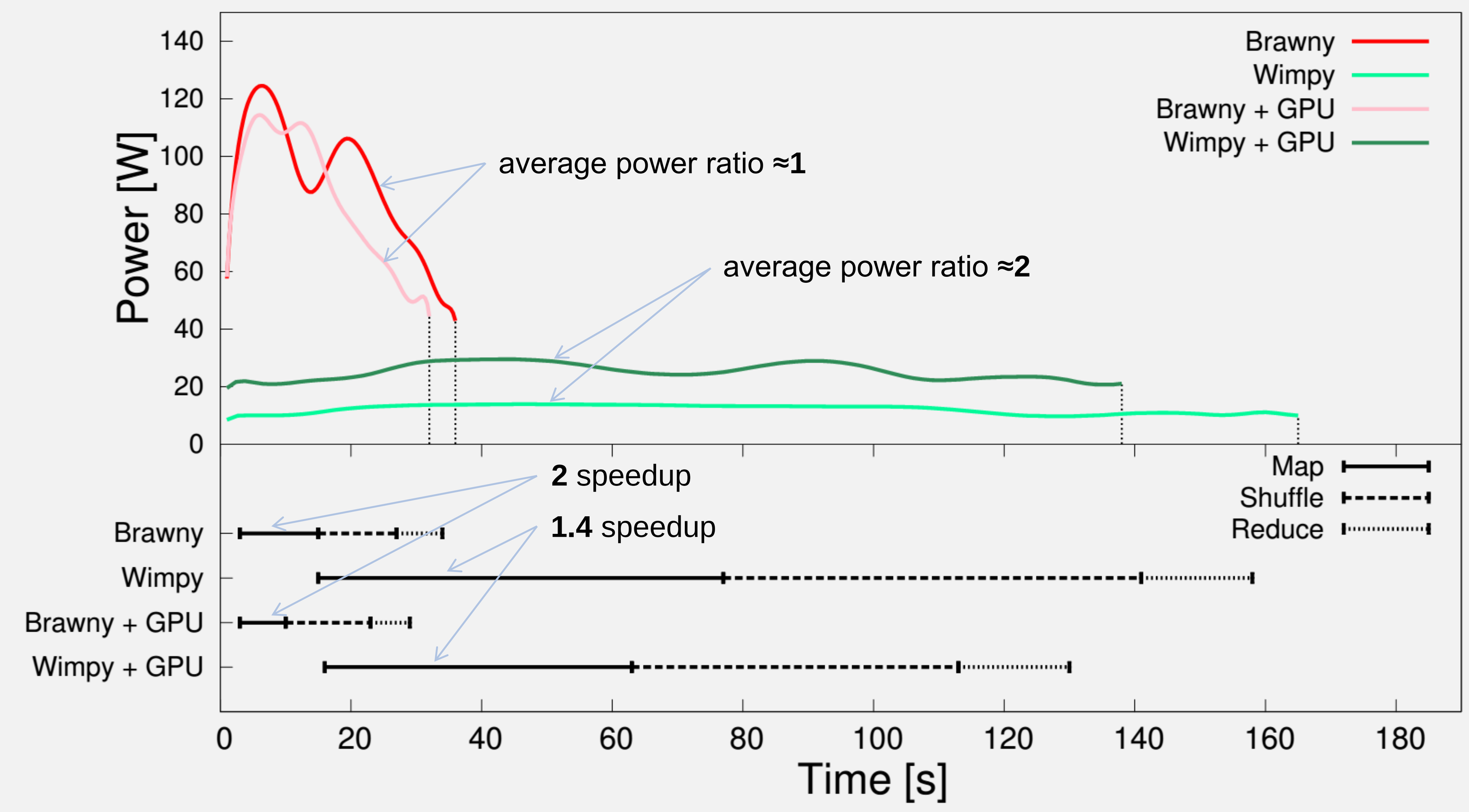






* all plots are in log scale

In-depth view of Kmeans(S) execution



Time

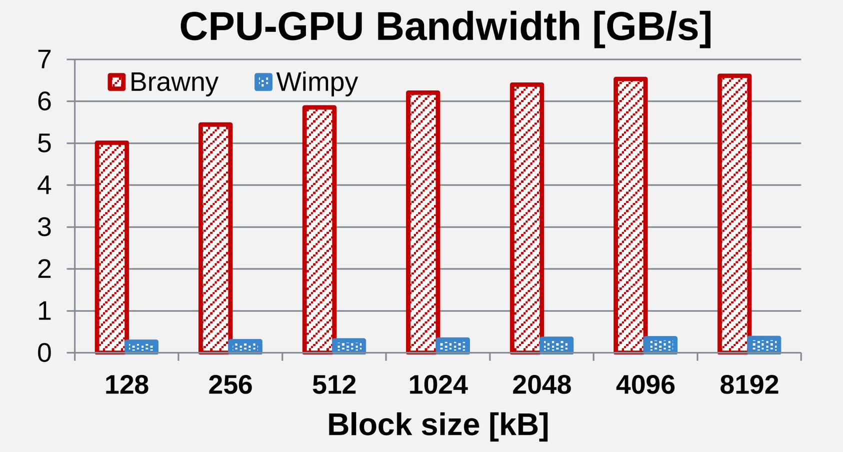
CPU-only execution time for a map task :

$$T_m = T_s(\text{setup time}) + T_p(\text{processing time})$$

CPU + GPU map task time:

$$T_m = T_s + T_t(\text{transfer time}) + T_k(\text{kernel time})$$

Assuming same kernel time on both brawny and wimpy systems, speedup difference is due to longer setup and transfer times on wimpy systems (see below).



19x higher CPU-GPU transfer bandwidth on brawny system

Energy

Given total execution time (*T*) and average power consumption (*P*), energy usage is:

$$E = T \cdot P$$

To reduce the energy of wimpy node with GPU (WG) to at least the energy of wimpy node only (W):

$$\frac{T_W}{T_{WG}} \geq \frac{P_{WG}}{P_W}$$

This leads to two options:

1. *improve execution time* so that speedup is at least 2 for the given average power ratio of 2 (see below)
2. *reduce GPU power* by 60% given an execution time speedup of 1.4

System	Average idle power [W]	Average active power [W]
Brawny only	42.9	104.9
Wimpy only	8.1	12.8
Brawny + GPU	43.5	107.1
Wimpy + GPU	16.6	25.9

5 Summary

- Analysis of time-energy performance of Big Data applications on heterogeneous systems with NVIDIA GPUs
- GPU with wimpy node is viable in future generations of wimpy systems with better computational performance and memory bandwidth
- Hadoop and CUDA on heterogeneous CPU-GPU nodes is not new [2], but we introduce an efficient method for lazy processing of <key, value> pairs

References

[1] L. Ramapantulu, B. M. Tudor, D. Loghin, T. Vu, Y. M. Teo, Modeling the Energy Efficiency of Heterogeneous Clusters, ICPP 2014

[2] K. Shirahata, H. Sato, S. Matsuoka, Hybrid Map Task Scheduling for GPU-based Heterogeneous Clusters, CloudCom 2010