On Understanding Time, Energy and Cost Performance of Wimpy Heterogeneous Systems for Edge Computing

Dumitrel Loghin, Lavanya Ramapantulu, Yong Meng Teo
Department of Computer Science
National University of Singapore
{*dumitrel,lavanya,teoym*}*@comp.nus.edu.sg*

***Abstract*—Motivated by the increasing importance of edge computing and improvements in heterogeneous low-power, wimpy systems, we present a novel time-energy-cost analysis of wimpy edge computing in comparison with traditional brawny cloud computing. For this analysis, we use a brawny heterogeneous Amazon EC2 cloud instance with GPU and two wimpy heterogeneous systems represented by Jetson TK1 and Jetson TX1. As the paradigm shift to edge computing is due to the challenges of Big Data processing, we select six representative MapReduce applications with applicability in IoT edge computing. Using our time-energy-cost analysis of both wimpy edge compute nodes and cloud computing systems with GPUs, we present several key insights. Firstly, we advocate the usage of heterogeneous systems with GPU on both edge and cloud since they provide time-energy savings of up to 70% for compute-intensive applications. Secondly, we establish an equivalence ratio between one brawny cloud instance and multiple wimpy edge nodes that achieve the same or better time performance. Based on this equivalence ratio, we show that using wimpy systems as edge computing devices saves cost compared to using traditional cloud computing. Lastly, counter-to-intuition, our analysis shows that the latest Jetson TX1 system exhibits worse time-cost performance compared to the older Jetson TK1 system. This result stems from lower operating core clock frequency and lower instructions-per-cycle of Jetson TX1's GPU on some compute-intensive applications.**

***Keywords*-edge computing; cloud computing; heterogeneous systems; GPU; brawny; wimpy; time performance; energy; cost performance**

I. Introduction

The transformation to "smart cities" is increasing the adoption of Internet of Things (IoT) devices which are predicted to grow beyond 30 billion by 2020 [22]. This phenomenal growth in IoT devices triggers an unprecedented data deluge giving rise to new challenges in terms of real-time data processing technologies. These challenges are causing a paradigm shift towards edge or fog computing to enable data processing closer to the source of data with higher bandwidth and lower latency [23].

As edge computing devices are located closer to IoT devices, they need to be both energy- and space-efficient for mobile application domains such as smart cars, vessel monitoring and so on. The increase in processing capabilities of processors used in mobile devices in the previous decade motivates the case for using such *wimpy* devices in edge computing due to their small form factor and low power consumption. Additionally, these wimpy systems are becoming heterogeneous with the addition of Graphics Processing Units (GPUs) to offload computations resulting in improved time performance and energy efficiency [9, 12, 25].

Nvidia Jetson family is an example of such wimpy heterogeneous systems referred to as mobile supercomputers due to their small size and high performance within the constrained power budget. Jetson TK1 [25] is built using Nvidia's Tegra K1 system-on-chip (SoC) and uses a 32-bit quad-core ARM Cortex-A15 CPU and 192-core Nvidia Kepler GPU. With the introduction of 64-bit ARM architecture, Jetson TX1 [12] based on Tegra X1 SoC employs four 64-bit ARM Cortex-A57 cores along with 256 GPU cores of the latest power-efficient Maxwell architecture.

With the need for new edge computing architectures that are both power- and space-efficient, and the technological advancements in wimpy systems, it is important to study if these systems fit the bill for edge computing. As edge computing is addressing the data deluge from IoTs, it is imperative to study the performance of edge systems with respect to data-intensive applications.

To address this, we perform a detailed measurement-based performance characterization of data analytics applications on Jetson systems to determine their suitability for edge computing. Our study evaluates the time, energy and cost performance of six MapReduce programs representing typical data analytics on two viable edge computing systems, the latest Jetson TX1 and the previous Jetson TK1 wimpy systems. In addition, we compare the performance of these wimpy heterogeneous systems with brawny heterogeneous cloud systems based on Amazon Elastic Compute Cloud (EC2) instances.

The main contributions and the key insights from this paper are:

- We advocate for the usage of heterogeneous systems with GPU for both the edge and the cloud since they provide time-energy savings of up to 70% for compute-intensive applications.
- We establish an equivalence ratio of 12 wimpy edge nodes that achieve the same or better performance compared to a single brawny cloud instance. Using this

978-1-5386-2017-5/17 $31.00 © 2017 IEEE
DOI 10.1109/IEEE.EDGE.2017.10

time-performance equivalence ratio, we show that edge computing using wimpy systems results in cost savings compared to brawny cloud computing servers except for the case where the manpower cost does not get amortized because of small cluster sizes.
- Counter-to-intuition, we observe that recent Jetson TX1 system exhibits lower time-cost performance compared to the older Jetson TK1 system. This is due to lower operating core clock frequency and the unexpectedly low Instructions-per-Cycle (IPC) of the TX1 GPU on some compute-intensive applications.

The rest of the paper is organized as follows. Section II provides the state of the art research on time, energy and cost performance analysis and edge computing architecture systems. Section III details the methodology with respect to applications and hardware systems used for our study. The analysis is presented in Section IV and Section V shows the summary and concluding remarks.

II. Related Work

The related work with respect to evaluation of the Tegra SoC architecture as a viable platform for edge computing can be classified into (i) energy-time and cost performance studies of GPU architecture systems for Big Data analytics and (ii) architecture feasibility studies for edge computing.

A. Time, Energy and Cost Performance of Data-Intensive Applications

While there are many studies involving improvements to existing frameworks for data-intensive applications using MapReduce on systems with GPUs [11, 15, 16, 29, 31], all of these works only focus on time performance aspects. Additionally, while many works do consider Nvidia GPU architectures for data-intensive applications, the system architectures considered belong to the traditional server class systems which are both power hungry and have large spatial volumetric footprints. In contrast, this paper addresses the question whether the latest 64-bit processor systems designed for the mobile market, such as Jetson TX1, are energy- and cost-efficient as edge computing devices.

Other studies analyze the power-performance trade-off between traditional x86 server architectures and processor architectures used in smart mobile devices. While these works provide useful insights on performance bottlenecks [5, 7, 13, 26], they focus on the 32-bit architecture version of the mobile processors without the integrated GPUs. These analysis results are worth revisiting with the improved computational and memory performance on 64-bit processors and with the integration of power-efficient GPUs. Additionally, the above works do not consider the cost of computation which is an important consideration for cloud users moving towards edge computing devices.

There are various cost models proposed to compare performance cost trade-off across different architectures for data-intensive processing and evaluating costs for datacenters [17, 18]. However these models only consider *on-premise* CPU only architecture systems. Due to the recently available power-efficient GPU architectures, this paper considers the cost of wimpy systems with GPUs and compare them with the cost of computing using *cloud* services. On the other hand, there are different cost models for efficient data-intensive processing using cloud computing services [3, 8, 32], but in contrast to this paper they do not address the cost of computations using on-premise wimpy architecture systems.

B. Hybrid and Edge Computing Architectures

With the increasing computational complexities of mobile applications, the CloneCloud framework [6] is a hybrid framework to automatically partition the application and seamlessly off-load portions of the executions on cloud resources in an elastic manner. However, due to the increased computational performance of the latest mobile processors, offloading parts of the mobile application might be unnecessary. In contrast, in this paper we are considering offloading parts of computation from the cloud to the mobile processor to enable cost-efficient edge computing. Nebula [27] is a light-weight architecture that proposes an *edge cloud* for distributed data-intensive processing and storage. However, they use emulated compute nodes as the edge hardware based on Planetlab while our paper proposes the usage of real hardware systems such as Jetson TK1 and Jetson TX1 as edge compute nodes.

III. Approach and Setup

A. Approach

The objective of this paper is to analyze the time, energy and cost performance of batch data analytics on wimpy edge computing devices in comparison with traditional brawny cloud instances, as depicted in Figure 1. For this analysis, we select two representative wimpy systems covering both 32-bit and 64-bit ARM processor architectures, and one type of brawny cloud instance from Amazon EC2. More details about these systems are provided in the next subsection.

For this work, we selected five representative MapReduce applications covering multiple domains and we run them on Hadoop, the most popular MapReduce framework [1]. Since Hadoop supports only CPU processing of input records, we employ our *lazy processing* technique to process the records on heterogeneous CPU+GPU systems [20]. On a GPU running m threads, lazy processing distributes each input record to a GPU thread, in contrast with chunking [29] which splits one record among the m threads.

B. Applications

For this measurement-based analysis of data analytics on edge computing, we select five representative MapReduce

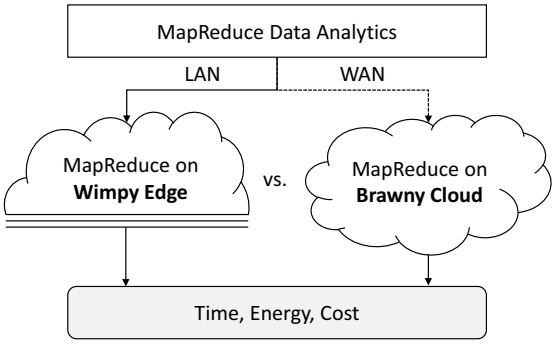


Figure 1: Approach Overview

applications covering different application domains, as summarized in Table I. Pi Estimation (**PI**) represents Monte Carlo simulations used in system optimizations, among other areas. Pi Estimation takes as input the number of samples to generate during the simulation. Due to the increasing importance of financial models, we select BlackScholes (**BS**), a financial model used to compute options prices. Each line of BS input files represents one option. To implement BlackScholes in MapReduce, we used the code from PARSEC benchmarking suite [4]. To cover machine learning, we select Kmeans (**KM**), an algorithm that groups n points from a space with m dimensions into k clusters. Our MapReduce implementation, adapted from Mars [15], computes one iteration of Kmeans algorithm. Representing mathematical computations, Matrix Multiplication (**MM**) is implemented in MapReduce such that Map phase computes each element from the result square matrix of size n. We implement Similarity Score (**SS**) as cosine similarity used in data mining to find entities with similar values. In our implementation, we compute the similarity between m pairs of vectors with n elements. To perform text processing, we select Grep (**GR**), an application that searches for a regular expression in a text file with a certain number of lines. In our experiments, Grep searches for the work *"the"* in Wikipedia's articles dump.

From IoT perspective, we can find at least one use case for each of the five applications. For example, Monte Carlo simulations can be used in localization applications [2], while mathematical computations represented by MM have many use cases in IoT and Wireless Sensor Networks (WSN) [28]. Financial models have been successfully applied in computer science to optimize power management on wimpy heterogeneous systems [30]. KM can be used to cluster output data and SS can be used to find data points with similar value. Grep is useful for analyzing very large logs produced by a variety of systems.

We evaluate each application for two input sizes, M and L as shown in Table I, for both CPU-only homogeneous systems and CPU+GPU based heterogeneous systems. The experiments are executed three times and the results with the minimum values are used in the paper.

Table I: Applications

Application	Input Size [GB]	Description
Pi Estimation (**PI**)	0.056 (M) 0.186 (L)	300 billion samples 1000 billion samples
BlackScholes (**BS**)	8.0 (M) 24.2 (L)	120 million options 360 million options
Kmeans (**KM**)	7.7 (M) 19.3 (L)	n=83,397,420, m=34, k=5 n=208,493,550, m=34, k=5
Matrix Multiplication (**MM**)	7.5 (M) 26.0 (L)	n=1000 n=1500
Similarity Score (**SS**)	7.5 (M) 26.0 (L)	n=1000, m=1,000,000 n=1500, m=2,250,000
Grep (**GR**)	11.1 (M) 22.3 (L)	166,656,938 lines 368,789,935 lines

C. Systems

We have selected three representative systems covering both brawny cloud and wimpy edge computing, as shown in Table II. For cloud, we have selected *g2.2xlarge* Amazon EC2 (**EC2**) instances equipped with eight virtual CPUs (vCPUs) of Intel Xeon E5-2670 type, and 1536 GPU cores of Nvidia Kepler architecture.

For wimpy edge, we have selected two systems with 32- and 64-bit ARM architectures. Jetson TK1 (**TK1**) is a wimpy heterogeneous system with four 32-bit ARM Cortex-A15 CPU cores and 192 Nvidia Kepler GPU cores integrated in the same chip and sharing 2GB of LPDDR3 memory. Newer generation Jetson TX1 (**TX1**) is a wimpy heterogeneous system with four 64-bit ARM Cortex-A57 CPU cores and 256 Nvidia Maxwell GPU cores integrated in the same chip and sharing 4GB of LPDDR4 memory. For both systems, the Ubuntu OS is installed on the eMMC found on-board. To manipulate large amounts of data, we have equipped each system with a 3TB hard-disk.

Before analyzing the system's performance on running data analytics, we perform benchmarking of sub-systems that play a crucial role in software execution, such as the CPU, GPU, memory, storage and networking. The results are summarized in Table II. Since data analytics frameworks, such as Hadoop, Spark, Cloud Dataflow, run on Java Virtual Machine, we evaluate CPU performance with our custom-designed Java benchmarks[1] that estimate the performance as million iterations per second (MITPS). The *FloatInt* benchmark is performing floating point and integer computations stressing CPU's pipeline, while *Stall* benchmark generates many main memory requests to evaluate the CPU performance for memory-bound programs. As expected, Intel Xeon CPU of cloud instances provides higher performance than ARM CPUs, in part due to higher core count. For compute-intensive *FloatInt*, Xeon performance is three times higher than ARM CPUs, while the 64-bit ARM CPU of TX1 provides 10% less performance than the 32-bit ARM CPU of TK1. We attribute this difference to the lower operating frequency of 1.73GHz on TX1 as compared to

[1] *https://github.com/dloghin/ubench*

Table II: Systems

		EC2	TK1	TX1
Specs	System	Amazon EC2 g2.2xlarge	Jetson TK1	Jetson TX1
	CPU	Intel Xeon E5-2670	ARM Cortex-A15	ARM Cortex-A57
	ISA	x86-64	ARMv7l	ARMv8-A
	Cores	8 (virtual)	4	4
	Frequency [GHz]	1.20 - 1.80	0.05 - 2.32	0.10 - 1.73
	LLC Cache [MB]	20	2	2
	Memory Type	-	LPDDR3	LPDDR4
	Memory Size [GB]	15	2	4
	GPU Architecture	Nvidia Kepler	Nvidia Kepler	Nvidia Maxwell
	GPU Cores	1536	192	256
CPU	Java FloatInt Performance [MITPS]	727	260	230
	Java FloatInt PFR [MITPS/GHz]	404	112	133
	Java FloatInt PPR [MITPS/W]	-	17.7	14.8
	System power [W]	-	14.7	15.5
	Java Stall Performance [MITPS]	9.66	0.86	1.60
	Java Stall PFR [MITPS/GHz]	5.37	0.37	0.92
	Java Stall PPR [MITPS/W]	-	0.07	0.13
	System power [W]	-	12.8	12.4
	Idle system power [W]	-	6.5	7.3
GPU	Performance [GFLOPS]	2157	209	477
	Average system power [W]	-	10.1	10.8
	Idle system power [W]	-	6.5	7.3
Memory	Bandwidth [GB/s]	9.1	2.2	3.4
Storage	Write throughput [MB/s]	123	132	96
	Read throughput [MB/s]	138	150	135
	Buffer read throughput [GB/s]	7.3	1.2	2.6
Network	TCP bandwidth [Mbits/s]	1080	715	736
	UDP bandwidth [Mbits/s]	808	669	474

2.32GHz on TK1. While the performance-to-frequency ratio (PFR) is higher for the 64-bit CPU, the overall performance-to-power ratio (PPR) is better on the 32-bit system. For memory-bounded programs represented by *Stall* benchmark, TK1 exposes its limits in terms of memory bandwidth and cache performance by providing half the MITPS of TX1. On the other hand, brawny Xeon-based cloud systems achieve eleven and six times better performance on *Stall* compared to TK1 and TX1, respectively.

At GPU level, Jetson TX1 is two-times better while using only 7% more power. Our measurements support Nvidia's claim that Maxwell architecture is two times more power-efficient than Kepler architecture [14]. Compared to the powerful GPU of EC2 instances, wimpy GPUs provide ten and five times lower performance on TK1 and TX1, respectively. For GPU evaluation, we have used *SHOC MaxFlops* benchmark [10].

We have pointed out that ARM wimpy systems suffer from small memory size and low memory bandwidth in our previous studies [20, 21]. Using *lmbench* [24], we observe improvements on the newer TX1 system. The memory bandwidth of TX1 has improved by 55% compared to TK1, but is more than 2.6 times lower compared to the bandwidth of Intel Xeon systems.

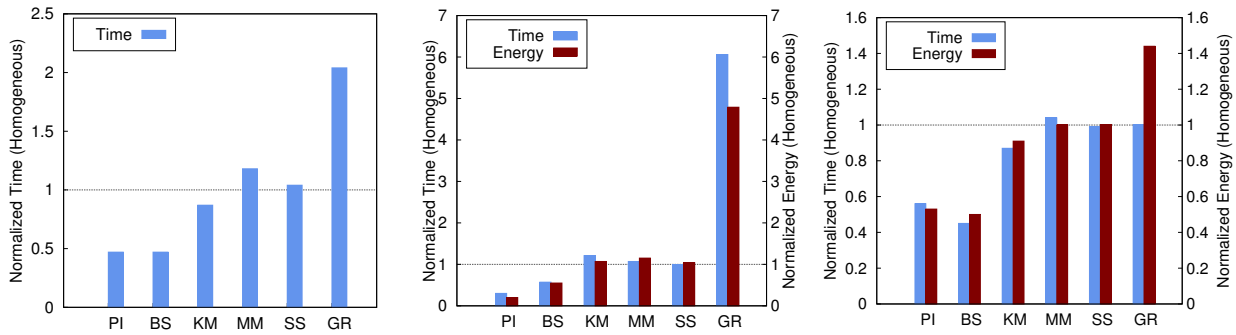
At storage and networking levels, the performance of cloud and edge systems is comparable. The only notable difference is for the read throughput of frequently accessed files that are buffered by the operating system in main memory. In case of buffered read, brawny cloud system based on Xeon architecture uses its larger memory size and higher memory bandwidth to achieve six and three times higher throughput compared to TK1 and TX1, respectively. We use Linux tools such as `dd` for storage benchmarking and `iperf` for networking profiling.

IV. Analysis

In this section, we analyze the time, energy and cost performance of data analytics on edge computing in contrast with traditional cloud computing. First, we investigate if heterogeneous systems with GPU accelerators exhibit time-energy improvements over homogeneous CPU-only systems. Secondly, we derive equivalence ratios between cloud brawny and edge wimpy systems based on their execution time performance. Thirdly, using derived equivalence ratios and a cost model for self-hosted clusters, we compare the cost of edge and cloud computing for data analytics.

A. Heterogeneous Systems Performance

In this section, we compare the time-energy performance of CPU-only, homogeneous systems, with CPU+GPU, heterogeneous systems. We measure the energy consumption only on the edge devices since cloud providers do not allow



(a) EC2 (b) TK1 (c) TX1

Figure 2: Time and Energy Comparison between Homogeneous and Heterogeneous Systems

power and energy measurements. We normalize time and energy to the values obtained on homogeneous systems and plot the results in Figure 2.

We show that for three out of six applications, namely PI, BS, and KM, heterogeneous systems with GPU save both time and energy. On EC2, the GPU improves execution time with 53% for PI and BS, and 13% for KM. On TK1, the GPU improves the time with 70% and 43% for PI and BS, respectively. On TX1, the GPU improves the time with 44%, 55% and 13% for PI, BS and KM, respectively. For SS, homogeneous and heterogeneous systems exhibit the same time performance but the system with GPU uses slightly more energy by activating the accelerator's hardware. For MM, the heterogeneous system is between 4% and 18% slower compared to the homogeneous system. We attribute this to the fact that MM computational kernel is small in terms of instructions. Only GR is much slower on EC2 and TK1 heterogeneous systems and uses more energy on TK1 and TX1 heterogeneous systems compared to homogeneous execution. We attribute this to the fact that GPU kernel is not suitable for GPU execution since threads processing different input lines may finish their execution at different moments in time depending on the position of the regular expression. A detailed analysis of GR behaviour on GPU is presented in our previous work [20].

In summary, we advocate for the usage of heterogeneous systems with GPU for MapReduce data analytics where some applications can exhibit significant time and energy improvements. For other applications, CPU-only execution can be enabled to obtained the best performance. While selecting the best of CPU-only or CPU+GPU execution can be done automatically, it is out of the scope of this paper. Rather, application developers can select during development phase which execution unit is more suitable and fix it for further executions. In the remainder of this paper, we use the best results between homogeneous and heterogeneous executions for each application on each system.

B. Edge-Cloud Equivalence Ratio

In this section, we compare the execution time performance of cloud and edge systems and derive an equivalence ratio between a single brawny cloud instance and multiple wimpy edge cluster nodes. As shown in Figure 3, the execution time ratio between one wimpy node and one brawny EC2 node is between two and six. The ratio of two is for MM and SS on TK1 and TX1. The ratio of six is exhibited by BS.M, BS.L and KM.M running on TK1, and by PI.L running on TX1. However, Hadoop does not scale linearly [20, 21], thus, we over-provision and approximate that 12 wimpy nodes of both TK1 and TX1 type are needed to do the job of a single brawny EC2 instance.

Comparing TK1 and TX1, we are surprised to observe that TK1 has better time performance for four out of six applications, namely PI, MM, SS and GR. While for MM, SS and GR this is due to the lower performance of TX1 CPU as shown in Section III-C, for PI executing on GPU this is surprising. We have profiled GPU kernel execution using `nvprof` from CUDA toolkit and found that the instruction per cycle on TX1 is only 0.44, while on TK1 is 1.95. This difference can be attributed to different GPU architectures but it shows a performance downgrade from Kepler to Maxwell.

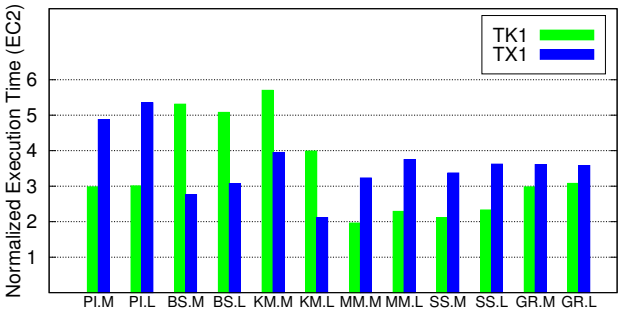


Figure 3: Execution Time Comparison with One Cloud Instance

Table III: Cost Model Notations

Notation	Description	Value	
		TK1	TX1
T	cluster lifetime *	3 years	
U	cluster utilization [%] *	10 - 100	
N	number of cluster nodes +	12	12
P_i	idle power per node [W] +	6.5	7.3
P_a	average power per node [W] +	16.2	14.2
C_s	cost of acquisition per node [USD] +	360	760
C_m	cost of manpower per month [USD] *	0 - 7000	
C_{ph}	cost of electricity per hour [USD] *	0.10	
C_p	total cost of electricity [USD] #	-	
C	total cost [USD] #	-	
C_h^c	cost of one EC2 instance [USD/h] *	0.65	

C. Cost Performance

In this section, we compare the cost of cloud and edge computing based on the equivalence ratio derived in Section IV-B. We define a cost model using the symbols in Table III[2] and we analyze the effect of different parameters, such as number of cluster nodes, cluster utilization and manpower salary on the total cost.

We define the cost of edge computing as the sum of equipment cost, system administration cost and electricity cost, similar to the marginal cost model for self-hosted clusters defined in our previous work [21],

$$C = N \cdot C_s + T \cdot C_m + C_p \quad (1)$$

where electricity cost depends on system's power and utilization,

$$C_p = T \cdot C_{ph} \cdot N \cdot (U \cdot P_a + (1 - U) \cdot P_i) \quad (2)$$

Equipment cost includes base-system kit, hard-disk and networking switch costs. The cost of a Jetson TK1 and Jetson TX1 kit is $200 and $600, respectively. We add $150 for the 3TB hard disk on each node, and $120 for a networking switch at cluster level. The cost of setting-up the cluster in included in equipment and manpower costs. Manpower cost is based on the average salary data found on recruiting websites[3] and ranges between $3000 and $7000 per month. We also consider the case where no additional system administrator is needed for small clusters, thus, manpower cost reduces to zero. Lifetime span of the cluster is three years [21], which translates to 36 months for manpower salary and 26280 hours for energy usage. For system utilization, we use the maximum value of 100% representing fully-utilized clusters. In practice, cloud servers have a typical utilization of only 10% [19].

First, we analyze the effect of varying the number of edge cluster nodes on total cost and compare it to Amazon EC2 price of $0.65 for one *g2.2xlarge* instance, as retrieved in March 2017. We assume fully-utilized clusters and no manpower cost, and vary the number of edge cluster nodes from one to eighteen as shown in Figure 4. We observe that the cost of edge computing is always smaller than the cost of cloud. However, the cost of TX1 is higher and is increasing faster than the cost of TK1, showing that the new Jetson TX-1-based heterogeneous wimpy system is not cost-efficient. In the remainder of this section, we use the equivalence ratio of 12, as derived in Section IV-B.

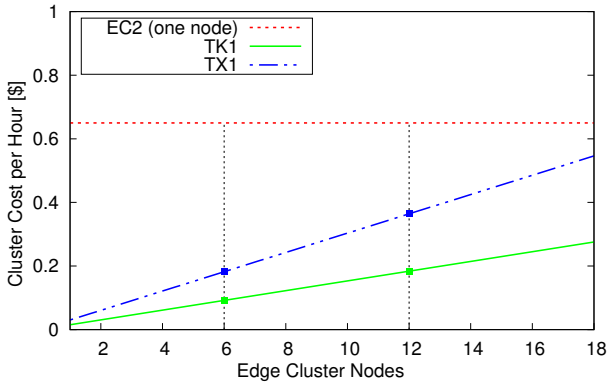


Figure 4: Effect of Cluster Node Count on Cost

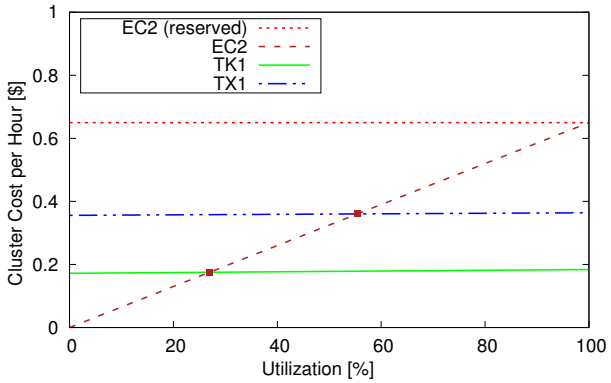


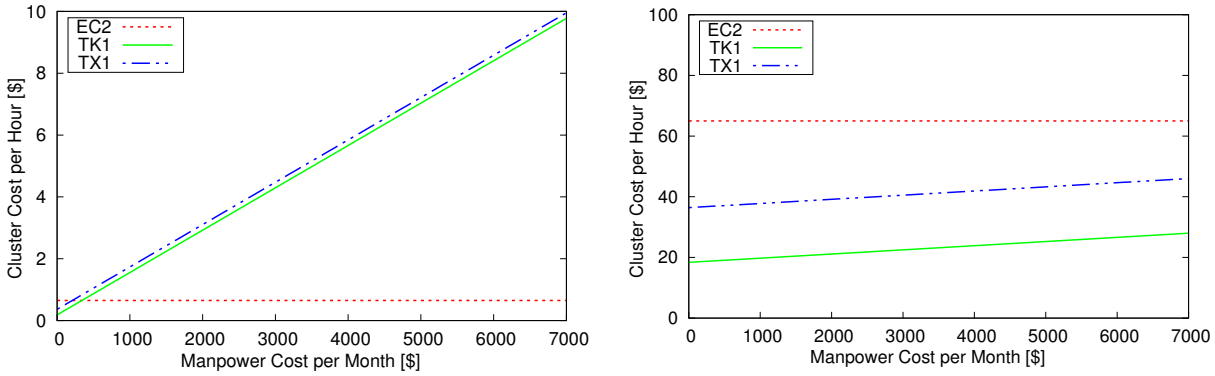
Figure 5: Effect of Utilization on Cost

Next, we analyze the impact of cluster utilization on total cost by comparing it to cloud cost when (i) cloud is reserved for the entire lifetime (or (ii) cloud is reserved and setup based on-demand depending on the utilization. We assume that there is no cost associated with manpower and that edge clusters have 12 nodes. As shown in Figure 5, high utilization does not lead to higher cost for edge computing. Rather, the cost per hour is roughly constant. On the other hand, the cost of on-demand utilization of cloud is lower when edge utilization is less than 27% and 55% for TK1 and TX1, respectively. However, for highly-utilized systems, the cost of edge is always smaller compared to cloud.

Lastly, we analyze the effect of manpower cost on the final cost per hour for three years of cluster lifetime. As shown in Figure 6a, varying manpower salary from $0 to $7000

[2]Descriptions are marked with * if the associated values are taken from the literature, with + if the values are measured by us, and with # for model output values.

[3]Salary data are taken from *www.glassdoor.com*



(a) 1-node cloud, 12-node edge

(b) 100-node cloud, 1200-node edge

Figure 6: Effect of Manpower on Cost

per month, we observe that a small edge cluster incurs a higher cost per hour compared to the cloud when salaries are higher than $340 and $210 for TK1 and TX1, respectively. Increasing the cluster size by 100 times, edge computing becomes cheaper than cloud even with a dedicated system administrator. We believe that organizations that want to adopt edge computing already have IT system administrators that can handle small and medium size cluster. Thus, we show that using the wimpy heterogeneous systems is not only energy-efficient but also incurs lower cost compared to cloud resources.

V. Conclusion

With the need for new cost-efficient edge computing architectures and the improvements of low-power wimpy systems based on ARM processors, we have presented a novel time, energy and cost performance analysis of wimpy edge systems in comparison with brawny cloud systems. For this comparison, we have selected an Amazon EC2 cloud instance with GPU and two heterogeneous wimpy systems representing edge computing devices. These two wimpy systems are Jetson TK1 and Jetson TX1 equipped with multicore ARM CPUs and Nvidia GPUs of different generations. Representing contemporary Big Data analytics workloads, we use six MapReduce applications covering a wide spectrum of domains related to IoT, and run then on Hadoop, the most popular data analytics framework.

We first show that compared to homogeneous CPU-only systems, heterogeneous systems with GPU provide time-energy savings of up to 70% for compute-intensive applications on both the edge and the cloud. Secondly, we compare the time performance of brawny cloud instances with wimpy edge nodes and establish an equivalence ratio of 12 wimpy edge nodes that achieve the same or better performance compared to a single brawny cloud instance. Based on this ratio, we show that wimpy edge computing incurs lower cost than brawny cloud computing except for the case of small wimpy edge clusters as they do not amortize the manpower costs of the system administrator. Additionally, counter-to-intuition, we observe that recent Jetson TX1 systems exhibits lower time-cost performance compared to older Jetson TK1 systems. This is in due to lower operating core clock frequency and the unexpectedly low IPC of the Jetson TX1 GPU on some compute-intensive applications.

Acknowledgement

This work was supported in part by Singapore Ministry of Education through the Academic Research Fund Tier 1. The authors thank Nvidia for providing hardware grants consisting of Jetson TK1 and Jetson TX1 systems, and Amazon for providing Elastic Compute Cloud credits.

References

[1] Apache, Hadoop, http://hadoop.apache.org, 2017.

[2] A. Baggio, K. Langendoen, Monte Carlo Localization for Mobile Wireless Sensor Networks, *Ad Hoc Networks*, 6(5):718–733, 2008.

[3] T. Bicer, D. Chiu, G. Agrawal, Time and Cost Sensitive Data-intensive Computing on Hybrid Clouds, *Proc. of 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 636–643, 2012.

[4] C. Bienia, S. Kumar, J. P. Singh, K. Li, The PARSEC Benchmark Suite: Characterization and Architectural Implications, *Proc. of 17th International Conference on Parallel Architectures and Compilation Techniques*, pages 72–81, 2008.

[5] E. Blem, J. Menon, K. Sankaralingam, Power Struggles: Revisiting the RISC vs. CISC Debate on Contemporary ARM and x86 Architectures, *Proc. of 19th IEEE International Symposium on High Performance Computer Architecture*, pages 1–12, 2013.

[6] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, A. Patti, Clonecloud: Elastic Execution between Mobile Device and Cloud, *Proc. of 6th Conference on Computer Systems*, pages 301–314, 2011.

[7] M. F. Cloutier, C. Paradis, V. M. Weaver, Design and Analysis of a 32-bit Embedded High-performance Cluster Optimized for Energy and Performance, *Proc. of 1st International Workshop on Hardware-Software Co-Design for High Performance Computing*, pages 1–8, 2014.

[8] M. Conley, A. Vahdat, G. Porter, Achieving Cost-efficient, Data-intensive Computing in the Cloud, *Proc. of 6th ACM Symposium on Cloud Computing*, pages 302–314, 2015.

[9] A. Cunningham, From Smartphone to Server Room: Nvidia's "Kayla" Shows the Future of Tegra, http://www.webcitation.org/6VcpwYsD5, 2013.

[10] A. Danalis, G. Marin, C. McCurdy, J. S. Meredith, P. C. Roth, K. Spafford, V. Tipparaju, J. S. Vetter, The Scalable Heterogeneous Computing (SHOC) Benchmark Suite, *Proc. of 3rd Workshop on General-Purpose Computation on Graphics Processing Units*, pages 63–74, 2010.

[11] R. Farivar, A. Verma, E. Chan, R. Campbell, MITHRA: Multiple Data Independent Tasks on a Heterogeneous Resource Architecture, *Proc. of 2009 IEEE International Conference on Cluster Computing and Workshops*, pages 1–10, Aug 2009.

[12] D. Franklin, Nvidia Jetson TX1 Supercomputer-on-Module Drives Next Wave of Autonomous Machines, http://www.webcitation.org/6qK3fusRx, 2015.

[13] M. Halpern, Y. Zhu, V. J. Reddi, Mobile CPU's Rise to Power: Quantifying the Impact of Generational Mobile CPU Design Trends on Performance, Energy, and User Satisfaction, *Proc. of 22nd IEEE International Symposium on High Performance Computer Architecture*, pages 64–76, 2016.

[14] M. Harris, 5 Things You Should Know About the New Maxwell GPU Architecture, http://www.webcitation.org/6VcZH7xTv, 2014.

[15] B. He, W. Fang, Q. Luo, N. K. Govindaraju, T. Wang, Mars: A MapReduce Framework on Graphics Processors, *Proc. of 17th International Conference on Parallel Architectures and Compilation Techniques*, pages 260–269, 2008.

[16] C. Hong, D. Chen, W. Chen, W. Zheng, H. Lin, MapCG: Writing Parallel Program Portable Between CPU and GPU, *Proc. of 19th International Conference on Parallel Architectures and Compilation Techniques*, pages 217–226, 2010.

[17] W. Lang, J. M. Patel, S. Shankar, Wimpy Node Clusters: What About Non-wimpy Workloads?, *Proc. of Sixth International Workshop on Data Management on New Hardware*, pages 47–55, 2010.

[18] K. Lim, P. Ranganathan, J. Chang, C. Patel, T. Mudge, S. Reinhardt, Understanding and Designing New Server Architectures for Emerging Warehouse-computing Environments, *ACM SIGARCH Computer Architecture News*, 36(3):315–326, 2008.

[19] H. Liu, A Measurement Study of Server Utilization in Public Clouds, *Proc. of 9th IEEE International Conference on Dependable, Autonomic and Secure Computing*, pages 435–442, 2011.

[20] D. Loghin, L. Ramapantulu, O. Barbu, Y. M. Teo, A TimeEnergy Performance Analysis of MapReduce on Heterogeneous Systems with GPUs, *Performance Evaluation*, 91:255–269, 2015.

[21] D. Loghin, B. M. Tudor, H. Zhang, B. C. Ooi, Y. M. Teo, A Performance Study of Big Data on Small Nodes, *Proc. of VLDB Endowment*, 8(7):762–773, 2014.

[22] C. Louis, Roundup Of Internet Of Things Forecasts And Market Estimates, 2016, http://www.webcitation.org/6omvoFAoD, 2016.

[23] A. Maher, IoT, from Cloud to Fog Computing, http://www.webcitation.org/6qK46SIKb, 2015.

[24] L. McVoy, C. Staelin, Lmbench: Portable Tools for Performance Analysis, *Proc. of 1996 Annual Conference on USENIX Annual Technical Conference*, pages 23–23, 1996.

[25] Nvidia, Nvidia Unveils First Mobile Supercomputer for Embedded Systems, http://www.webcitation.org/6VdkUISQn, 2014.

[26] N. Rajovic, L. Vilanova, C. Villavieja, N. Puzovic, A. Ramirez, The Low Power Architecture Approach Towards Exascale Computing, *Journal of Computational Science*, 4(6):439–443, 2013.

[27] M. Ryden, K. Oh, A. Chandra, J. Weissman, Nebula: Distributed Edge Cloud for Data Intensive Computing, *Proc. of 2014 IEEE International Conference on Cloud Engineering*, pages 57–66, 2014.

[28] H. Shen, J. Chen, Efficient Matrix Multiplication on Wireless Sensor Networks, *Proc of. 7th International Conference on Grid and Cooperative Computing*, pages 331–341, 2008.

[29] K. Shirahata, H. Sato, S. Matsuoka, Hybrid Map Task Scheduling for GPU-Based Heterogeneous Clusters, *Proc. of 2nd International Conference on Cloud Computing Technology and Science*, pages 733–740, 2010.

[30] T. Somu Muthukaruppan, A. Pathania, T. Mitra, Price Theory Based Power Management for Heterogeneous Multi-cores, *Proc. of 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 161–176, 2014.

[31] J. A. Stuart, J. D. Owens, Multi-GPU MapReduce on GPU Clusters, *Proc. of 25th IEEE International Parallel and Distributed Processing Symposium*, pages 1068–1079, 2011.

[32] C. Szabo, Q. Z. Sheng, T. Kroeger, Y. Zhang, J. Yu, Science in the Cloud: Allocation and Execution of Data-intensive Scientific Workflows, *Journal of Grid Computing*, 12(2):245–264, 2014.