On Characterization of Performance and Energy Efficiency in Heterogeneous HPC Cloud Data Centers

Amer Qouneh, Nilanjan Goswami, Ruijin Zhou, Tao Li
Department of Electrical and Computer Engineering
University of Florida
Gainesville, Florida, USA
{aqouneh, zhourj}@ufl.edu, nil@ieee.org, taoli@ece.ufl.edu

***Abstract*—The relocation of high performance computing systems (HPC) to the cloud poses new challenges for data center architects and IT managers. These challenges are due to heterogeneity injected into data centers by cutting-edge virtualization technologies and hardware accelerators used to support emerging cloud applications and services. Although hardware accelerators like General Purpose Graphics Processing Units (GPGPUs) and virtualization technologies have been well studied and evaluated individually, a detailed analysis of their combined architectures and collective behavior from the data center point of view is lacking.**

Using real platforms and high performance computing workloads, we study the power performance tradeoffs due to various granularities of heterogeneity across hardware and software layers and expose hidden opportunities for optimizing overall data center efficiency. Our approach is to evaluate server power and performance from a data center point of view as opposed to evaluating hardware accelerators and virtualization technologies themselves.

Our results show that performance on cloud is affected by virtualization overhead and fraction of serial code. Moreover, GPU workloads achieve 25% and 30% savings in power and energy consumption when executed on low power platforms; and only 50% of our GPU workloads are more energy efficient than their corresponding CPU implementations. The results also show that it is much more power efficient to collocate GPU virtual machines with non-GPU virtual machines.

Keywords—heterogeneous; high performance computing; data center; efficiency; cloud

I. Introduction

High performance computing (HPC) has been associated with homogeneous high-performance systems supported by high-speed networks. However, due to frequent hardware refreshing and replacements, the underlying architecture is increasingly becoming heterogeneous [1]. Currently, GPUs are becoming a staple in HPC computing and thus adding another level of heterogeneity to the data center [2]. Significant performance inefficiencies can result if heterogeneity is not taken into consideration during scheduling [3]. Not only does heterogeneity affect performance, but it also affects power consumption. For example, GPUs consume substantially greater power than multi-core architectures [4] in return for performance. Figure 1 shows power profiles of eleven HPC benchmarks implemented in both CUDA (a programming model for GPUs) and OpenMP (parallel programming model that runs on multi-core CPUs). The benchmarks are executed sequentially on two identical platforms one with GPU and the other without. It is evident that the CUDA batch executes about 2.25 times faster than OpenMP batch. The GPU profile is strongly pulse-shaped having a peak to idle power ratio of 3.4 while the OpenMP profile is rather flat having a peak to idle power ratio of 2.7. CUDA consumes double the average power of OpenMP and has a peak power draw of 217W compared to 70W for OpenMP. Such disparity in power draw imposes greater pressure on overall power consumption. Recent studies have shown that world-wide consumption for 2010 exceeded 250 billion kWh, almost 1.5% of the world's total electricity consumption [5], and the US consumed 100 billion kWh in 2011 [6].

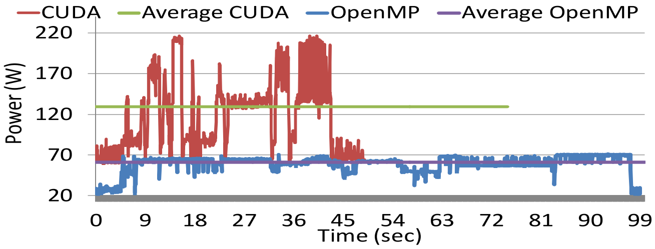


Figure 1. Power profiles of CUDA and OpenMP benchmarks

In order to expand services and support new applications like remote workstations and cloud gaming, considerable efforts have been exerted to move HPC computing to the cloud [1,7,8]. However, these efforts have been met with some reluctance, mainly due to performance and management issues [9].

Previous work [4,10,11] have evaluated GPUs themselves in non-virtualized systems, but evaluation of heterogeneous servers that contain GPUs within virtualized environments is lacking. In this work, we study the effects of heterogeneity on performance, power, and energy profiles for emerging platforms in virtualized data centers. Our results show that CUDA performance on cloud is affected by virtualization overhead and size of serial code. They also show that CUDA implementations are not always more energy efficient, and that it is better to collocate OpenMP VMs with CUDA VMs than to collocate VMs of the same type.

This paper is organized as follows: Section II provides a background on heterogeneity in HPC data centers on the cloud. Section III presents our experimental setup and test-bed. Section IV analyses experimental results. We present related work in Section V and our conclusion in Section VI.

II. Background

The advent of virtualization and low cost GPUs has accentuated the effects of heterogeneity on data center performance and energy consumption. In this work, we consider four levels of heterogeneity: execution paradigm, virtualization, micro-architectural variations, and platform heterogeneity.

A. Execution Paradigm

We consider the application layer as a form of heterogeneity because the execution paradigm determines the architecture to be used. Traditionally, HPC workloads have been implemented in one of several parallel programming languages like MPI and OpenMP. OpenMP runs on a single symmetrical multi-processor (SMP) server under a single operating system domain.

GPUs are parallel architectures optimized for high throughput by executing many concurrent threads. GPUs are installed in the Peripheral Component Interconnect (PCI) slots of a server. In this model, an application is written in a special programming interface like CUDA or OpenCL, to launch kernels. A kernel is a function callable from the host and executes on the GPU. Due to availability of massive parallelism, applications could experience significant speedups.

B. Virtualization

Virtualization has been employed in data centers to improve utilization through consolidation and is a corner stone in the cloud model due to its benefits in management, security, and live migration [12]. Virtual Machines (VMs) are software abstractions of hardware architectures [9] where a hypervisor (also called a Virtual Machine Monitor VMM) acts as an interface between applications and the underlying hardware.

Virtualization allows multiple guest virtual machines to be collocated on the same physical server. Despite the advantages, performance issues due to virtualization overhead and management efficiency hinder the adoption of virtualization in HPC data centers [8,9]. Virtualization overhead results from the delay caused by the interface layer (VMM) while management difficulties are caused by the lack of management framework to map and dynamically distribute VMs and OS images to physical machines [8]. These issues have been addressed by providing PCI pass-through, OS-bypass, and bypass capabilities to I/O and peripherals with hardware support from CPU and chipsets [8,9,13,14]. Bypassing the hypervisor provides VMs with direct access to I/O and hardware, and improves performance significantly. This architecture allows a GPU to be attached or passed-through to a VM with minimum penalty on performance. In our experiments, we explore the effect of varying the number of VCPUs allocated to VMs running CUDA and OpenMP benchmarks.

C. Micro-architectural Variations

Unlike enterprise data centers, HPC data centers do not enjoy low utilization periods where typical power management techniques can be applied. Often, HPC data centers run batch-type workloads for days with very little idle time. The main power management technique has been dynamic voltage and frequency scaling (DVFS) in which the voltage or frequency of the CPU clock are varied. The dynamic power of the CPU is a quadratic function of its operating voltage. Hence, reducing voltage/clock speed results in considerable power savings at the cost of longer execution times. Depending on platform architecture and configuration, the effectiveness of CPU frequency scaling varies widely. In particular, number of supported nodes, memory, storage, and GPUs can significantly shrink the savings fraction compared to total server power.

D. Platform Heterogeneity

New generations of low power architectures and processors are becoming popular in data center design. For example, Google data centers rely on commodity servers and architectures for low cost and high performance [2]. These architectures excel at running non-critical applications with little loading [15]. However, their use in HPC data centers may not be considered due to their low throughput. But if combined with a high-end GPU to shoulder the bulk of the computation, then the CPU's role reduces to executing I/O and housekeeping serial code which does not require a high-end processor. Under this arrangement, overall power savings are possible because no intensive computation is performed on the low power CPU. This has direct effect on power consumption and cooling [16,17].

III. Experimental Setup

Our test-bed consists of two platforms summarized in Table I. Both platforms are identical except for the CPU. Each platform is configured with a GPU installed on the PCI slot through a riser card for easy access of power supply lines. A LabView virtual instrument controlled a data acquisition card NI PCI-6221 to collect current profile data from current sensors. To execute CUDA benchmarks, VMs are configured with PCI pass-through capability for attaching a GPU. To setup pass-through, the hardware must support hardware-assisted instruction set virtualization capability and I/O DMA remapping.

We characterize benchmarks from Rodinia suite [10], which is designed for architectural studies on GPUs, and includes applications and kernels which target multi-core CPU and GPU platforms. The profiled benchmarks are listed in Table II. All benchmarks have implementations in CUDA and OpenMP respectively. We use Xen [14] as our virtual machine monitor.

IV. Characterization of Workloads

In this section, we characterize and compare profiles of CUDA and OpenMP implementations across various levels of heterogeneity. Based on our results, we propose guidelines to improve data center efficiency by leveraging heterogeneity.

TABLE I. HARDWARE TEST-BED

CPU	Intel i7 2600S, 4-core, TDP (65 W)
CPU - Low Power	Intel i5 3470T, 2-core, TDP (35 W)
Motherboard	GigaByte GA-H77M-D3H
Server Memory	32 GB
VM Memory	2 GB
Storage	1 TB
GPU	Nvidia Tesla M2050
Operating Systems	Fedora 17, Xen 4.1.3, Ubuntu 12.04
Data Acquisition System	LabView, NI PCI-6221
Current Sensors	HCS-20-10-AP

TABLE II. WORKLOAD SYNOPSIS

Benchmark	**Description**
Back Propagation (BP)	Machine learning algorithm
Breadth First Search (BF)	Graph algorithms
Heart Wall (HW)	Tracks movements of hearts
Hotspot (HS)	Estimates processor temperature
LavaMD (LV)	Calculates particle potential, relocation
LU Decomposition (LD)	Algorithm to solve linear equations
Nearest Neighbor (NN)	Finds the k-nearest neighbors
Needleman-Wunsch (NW)	Nonlinear optimization for DNA
PathFinder (PF)	Dynamic programming to find paths
SRAD (SR)	Diffusion method for imaging
Streamcluster (SC)	Finds predetermined no. of medians

TABLE III. FRACTION OF SERIAL TIME

	BP	BF	HW	HS	LV	LD	NN	NW	PF	SR	SC
OpenMP	0.70	0.98	0.001	0.48	0.0002	0.15	0.7	0.23	0.70	0.009	0.06
CUDA	0.94	0.99	0.3	0.87	0.34	0.9	0.9	0.75	0.92	0.28	0.67

A. Execution Paradigm

Although virtualization has been well studied, no evaluation exists for virtual machines equipped with GPUs. In this set of experiments, we evaluate performance, power, and energy profiles for servers running VMs with pass-through GPUs. Due to virtualization overhead, execution times can be greater than those on bare metal systems. Figure 2(a) shows percent change in execution time for both implementations CUDA and OpenMP compared to their respective bare metal times. CUDA implementations show greater degradation than OpenMP due to virtualization. *BF* and CUDA *SC* incur the greatest degradation with 87% and 50% respectively. It is possible that *BF* has difficulties with memory access patterns and I/O delays in reading input data.

Because of their high throughput and many cores, GPUs are capable of significant speedup of parallel code compared to multi-core CPUs. Figure 2(b) shows speedup of CUDA implementations relative to OpenMP using total execution time not just parallel code time. Four CUDA benchmarks indicate at least 4× performance improvement over OpenMP; for example, *SR* achieves a speedup of 5.8×. However, benchmarks *BP*, *BF*, *HS*, and *NN* actually run slower on GPUs due to CUDA overhead not due to virtualization. Unlike [10], which reports speedups only for the parallel code running on GPU, we show overall program speedup. Data centers are concerned with overall speedup since total execution time is what matters. Although parallel code alone may attain speedups of as much as 80, none of CUDA benchmarks reaches speedup of 6 for total execution time.

Table III presents a closer view of program structure in terms of serial code fraction for both implementations. CUDA serial codes represent greater fractions than their OpenMP counterparts because CUDA parallel fractions are smaller. Serial code fractions (including IO) for five CUDA benchmarks are at least 90% of the total time. Only four OpenMP benchmarks have serial fractions greater than 70%. Serial code remains a limiting factor for achieving greater speedups for both implementations.

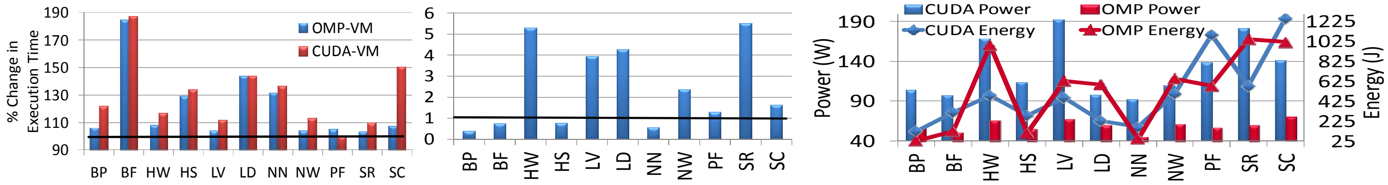
The tradeoff for greater speedups in GPUs is power consumption; Figure 2(c) indicates that CUDA implementations consume between 2× and 4× the power of OpenMP across all benchmarks. In contrast, energy consumption is a mixed lot. Six CUDA benchmarks consume greater energy than their OpenMP versions; for example CUDA *PF* consumes almost 2× that of OpenMP implementation, but CUDA *LD* consumes 62% less energy. This suggests that not all CUDA workloads are more energy efficient than OpenMP implementations.

Figure 3(a) shows that CUDA and OpenMP VM powers are within 5% and 10% of their bare metal powers. CUDA VM powers seem closer to bare metal because GPU power dwarfs any power increase due to virtualization and thus show smaller variation. Figure 3(b) compares CUDA's average server power to peak server power; the maximum power consumed by a workload during execution. The dynamic range of peak power varies widely between 24%-123% of its average power. For example, peak power for *LD* is about 2.23× that of its average power while *LV* indicates only 24% increase. The actual power draw of GPUs rarely reaches its nameplate power rating. Our Tesla M2050 GPU is rated at 225W TDP [18] and Figure 3(b) indicates that total peak *server* power running CUDA workloads never reached 240W. Thus, a data center power infrastructure can be oversubscribed to maximize power efficiency and reduce stranded power [19,20]. Using our detailed power profiling, peak and average powers can be determined as shown in Figure 3(c). The figure shows a power trace for *LV*. GPU starts withdrawing its dynamic power as soon as a kernel is launched as indicated by the 12V lines.

We can conclude that performance of OpenMP and CUDA workloads is affected by virtualization overhead and the degree depends on workload characteristics. We can also infer that serial code is a bottleneck for both especially CUDA workloads, and that architectures which maximize performance of serial code are essential for increasing speedup. In terms of energy consumption and efficiency, we found that only half of CUDA implementations were more efficient than OpenMP implementations. The heterogeneity due to GPU pass-through greatly improves the viability of future HPC computing on the cloud.

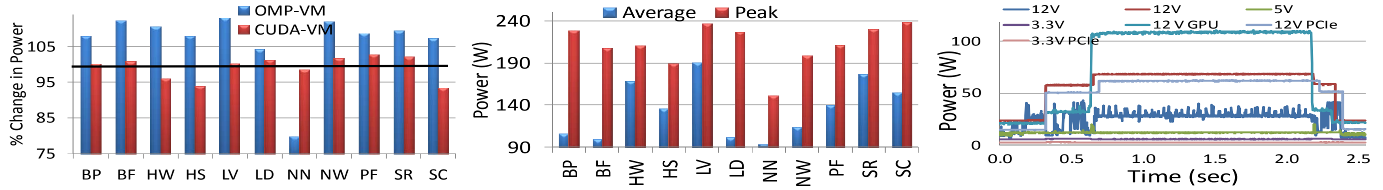
B. Virtualization

Another level of heterogeneity on the cloud is the set of resources allocated to VMs like number of virtual CPUs, memory size, storage size, and network bandwidth. In this section, we characterize benchmarks by varying the number of allocated VCPUs.



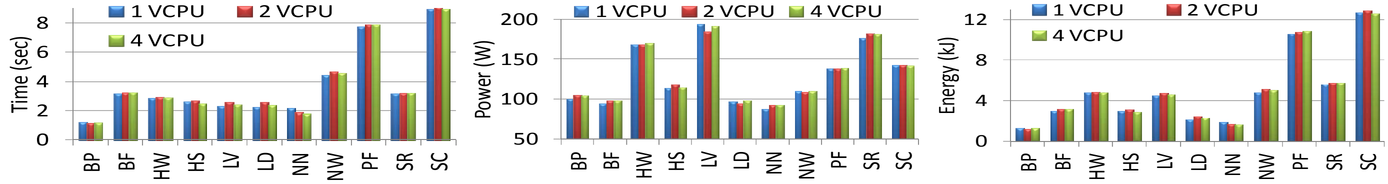
(a) Timing with respect to bare metal (b) Speedup of CUDA relative to OpenMP (c) Power and energy consumptions

Figure 2. Timings and power profiles



(a) Comparison of VM to bare metal powers (b) CUDA average and peak server powers (c) Power trace for CUDA LV

Figure 3. Power taxonomy



(a) Execution time (b) Power (c) Energy

Figure 4. VCPU profiles for CUDA VMs

Since our CUDA benchmarks are single threaded, it is obvious that execution time does not vary with number of allocated VCPUs, as shown in Figure 4(a). Similarly, power and energy consumptions by CUDA benchmarks do not change with number of allocated VCPUs, Figures 4(b) and (c). In contrast, increasing number of VCPUs considerably improves performance of OpenMP benchmarks. For example, *SR* achieves 2.5× speedup after gaining three VCPUs. As expected, OpenMP benchmarks show noticeable change in power and energy consumptions as the number of VCPUs is increased (OpenMP figures not shown). For maximum efficiency in heterogeneous data centers on the cloud, it is best to re-claim excess VCPUs from CUDA workloads and re-allocate them to other collocated VMs; this frees stranded resources and improves utilization.

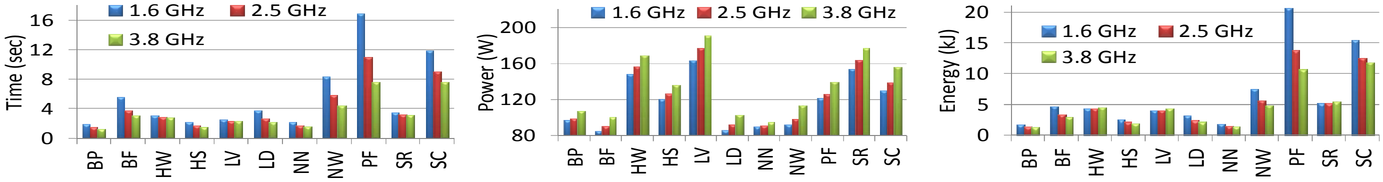
C. *Micro-architectural Variation*

DVFS is a prevalent power management technique in data centers to throttle server power consumption at the expense of performance. The effect of power management on execution time in VMs for CUDA workloads is shown in Figure 5(a). Benchmarks *HW*, *LV*, and *SR* show very little response to increase in frequency while the rest show noticeable change. *NW* for example, shows 1.88× improvement in performance at 3.8 GHz. The reason for the variation in performance is the fraction of serial code within the benchmarks. Power and energy consumptions follow same reasoning, as shown in Figures 5(b) and (c); amount of power or energy variation is directly related to the size of serial code affected by change in frequency. In contrast to CUDA workloads and as expected, OpenMP workloads react more energetically to variation in frequency. For example, *HW* achieves a speedup of 2.2× at the maximum frequency and experiences a 60% increase in power and 27% decrease in energy consumption (OpenMP figures not shown). CUDA workloads that contain a large fraction of parallel code are least affected by frequency scaling. DVFS affects code running on the CPU and has no effect on CUDA code running on the GPU.

For power capping at the cost of small performance penalty and minimum change in power and energy consumptions, DVFS can be invoked on CUDA workloads containing a large parallel fraction. The large available parallelism in GPUs minimizes the effects on performance degradation in serial code. However, for CUDA workloads with small parallel fraction, serial code is a bottleneck and frequency scaling has significant effect on performance, power, and energy consumptions. As expected, frequency scaling for OpenMP workloads can be applied to optimize for power or energy. VCPUs are decoupled from physical CPUs and Xen's scheduling policy can be enabled to map CUDA workloads with large fraction of parallel code to CPUs with low frequency. CUDA workloads with small fractions of parallel code and OpenMP workloads can be mapped to CPUs with high frequency. This optimizes performance and energy consumption.

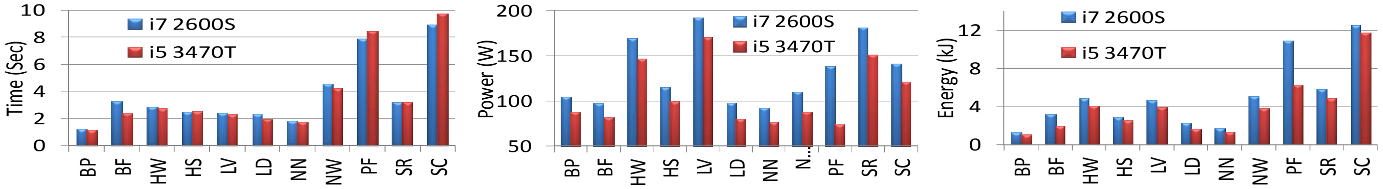
D. *Platform Heterogeneity*

New generations of low power processors serve a class of applications that does not require the rich set of configuration



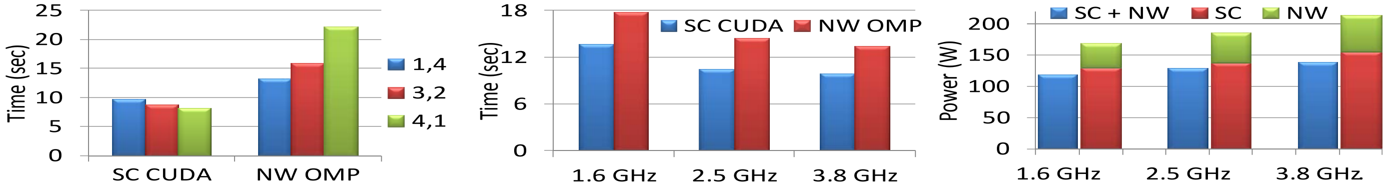
(a) Execution time (b) Power (c) Energy

Figure 5. DVFS profiles for CUDA VMs



(a) Execution time (b) Power (c) Energy

Figure 6. Platform heterogeneity for CUDA VMs



(a) VCPU redistribution (b) Execution time due to DVFS (c) Power consumption with DVFS

Figure 7. Consolidation of VMs

options that come with high performance systems [15]. In this sub-section, we study the effects of low power architectures on HPC workloads using a platform with a Generation 3 low power Intel i5 processor (TDP 35W) and comparing it to a platform with a main stream i7 processor (65W).

Figure 6(a) shows that CUDA performance on the low power platform is equal to or slightly better than the main stream platform except for two benchmarks *PF* and *SC*. This can mainly be attributed to high performance of I/O and serial code execution on the newer processor. In terms of power consumption, Intel i5 platform achieves 13% to 87% savings compared to the main stream platform, Figure 6(b). Similar energy savings that range from 12% to 62% are indicated in Figure 6(c).

For CUDA workloads, the performance of platform architecture depends mainly on I/O system and serial code performance. In contrast, seven OpenMP workloads show better performance on the Intel i7 platform (figure not shown). This can be attributed to the intense competition among the parallel execution threads for a smaller set of available resources on the i5 platform. The results indicate that CUDA workloads running on low power platform architectures can deliver similar performance as main stream platforms because the bulk of the computation is offloaded to the GPU and the CPU's task reduces to I/O and serial code execution.

In terms of power and energy consumptions, low power platforms can potentially save 25% and 30% in power and energy compared to high-end platforms. GPU's power is same on both platforms but the low power platform contributes less idle and dynamic powers to the total power than the main stream platform.

E. Consolidation of Virtual Machines

While performance and resource contention have been extensively studied in collocated environments, power efficiency of collocated VMs did not receive equal attention. In previous sections, we analyzed single VMs running on a physical platform. In this section, we analyze performance and power consumption of consolidated VMs under various resource and power management configurations. We select two representative benchmarks for consolidation. Figure 7(a) shows two collocated VMs, CUDA VM running *SC* and an OpenMP VM running *NW*. Five VCPUs are divided among both VMs; for example, the legend (1,4) means one VCPU for CUDA and four VCPUs for OpenMP. As shown in Figure 4, CUDA VMs do not benefit from extra VCPUs; however, OpenMP *NW* does benefit from additional VCPUs and execution time improves by 40% when four VCPUs are allocated. In Figure 7(b), DVFS is applied to both VMs and both show improvement in performance at higher frequencies; CUDA *SC* contains 67% serial code that is affected by the variation in frequency and improves by 27%. Due to resource competition, OpenMP *NW* takes three seconds longer to execute when collocated compared to running singly and improves by 25%. Figure 7(c) compares platform power consumption of consolidated VMs to those of individual VMs each running on its own platform. The figure shows that average platform power consumption of consolidated VMs (*SC*+*NW*) is almost the same as that of CUDA *SC* running alone. The consolidated power is 35% less than the sum of individual power consumptions for CUDA *SC* and OpenMP *NW* when each is running on its own platform (stacked bar). It is evident that a CUDA VM dominates total platform power and that power consumption due to a collocated OpenMP VM is negligible. The results show that while there is slight performance degradation, it is much more power efficient to collocate OpenMP VMs with CUDA VMs than to collocate VMs of the same type or to run them individually on different platforms. The advantage is due to the disparity between GPU and CPU power consumptions, less competition for resources among different type VMs, and the fact that CPU power is only a small fraction of total platform power. This allows OpenMP VMs to piggyback on CUDA VMs for almost free power at the cost of about 25% degradation in performance in OpenMP *NW*'s case. Similar or greater degradation occurs if OpenMP VMs are collocated with other OpenMP VMs but at the cost of greater power consumption.

V. Related Work

The closest work to ours is by Che et al. [10]. Our work extends but differs starkly from [10] in that our evaluation a) adopts the data center point of view not the GPU, b) is performed in a virtualized environment with GPU passed-through thus incurring overheads, c) emphasizes total program execution time which is more relevant from the data center point of view, d) employs realistic heterogeneous test platforms (with/without GPUs). Mars et al. [1] exploit heterogeneity by mapping workloads to hardware using an opportunity factor that quantifies an application's sensitivity to available heterogeneity. Nathuji et al. [21] leverage heterogeneity by mapping workloads to the best fitting platform using an analytical layer to predict workload power/performance. Delimitrou et al. [3] proposed an online scheduler that is heterogeneity and interference aware. Work on improving performance in virtual machines addressed improving I/O and pass-through capabilities. Huang et al. [8] proposed VMM-bypass I/O and scalable VM image management to improve performance. Gupta et al. [22] reported that HPC performance on the cloud is viable especially for non-communication intensive applications. Reuther et al. [9] also advocate the use of virtual machines in HPC scenarios where productivity outweighs performance degradation. Our work builds on previous ideas but distinctly addresses exposing the available opportunities in heterogeneous HPC data centers with emphasis on GPU workloads.

VI. Conclusion

In this paper, we expose power performance tradeoffs due to available heterogeneity across real hardware and software layers in HPC data centers. We expose hidden opportunities for optimizing overall efficiency of data centers running VMs configured with GPUs. Our results show that the fraction of serial code is a bottleneck and especially limiting for speedups of CUDA workloads and that some CUDA benchmarks are less energy efficient than their corresponding OpenMP implementations. It is best to reclaim VCPUs from CUDA VMs and reallocate them to collocated VMs that do need them. Our findings also show that power management techniques can be applied to CUDA workloads with large parallel code running inside VMs without noticeable degradation. An interesting finding is that 25% and 30% power and energy savings are possible with no degradation in performance when CUDA workloads are executed on low power platforms as opposed to running on high end platforms. For greatest power efficiency, it is best to collocate OpenMP VMs with CUDA VMs than to collocate VMs of the same type.

Acknowledgment

This work is supported in part by NSF grants 1320100, 1117261, 0845721 (CAREER), and by Microsoft Research Safe and Scalable Multi-core Computing Awards.

References

[1] J. Mars, L. Tang, R. Hundt, "Heterogeneity in homogeneous warehouse-scale computers: A performance opportunity," IEEE Computer Architecture Letters (CAL), Vol. 10, 2011.

[2] V. Mauch, M. Kunze, M. Hillenbrand, "High performance cloud computing," Future Generation Computer Systems, Volume 29, 2013.

[3] C. Delimitrou, C. Kozyrakis, "Paragon: QoS-aware scheduling for heterogeneous datacenters," International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2013.

[4] H. Nagasaka, N. Maruyama, A. Nukada, T. Endo, S. Matsuoka, "Statistical power modeling of GPU kernels using performance counters," International Green Computing Conference, 2010.

[5] Growth in data center electricity use 2005 to 2010. http://www.analyticspress.com/datacenters.html

[6] U.S. EPA. Report to congress on server and data center energy efficiency. EPA, Tech. Rep., 2007.

[7] E. Lee, H. Viswanathan, D. Pompili, "VMAP: Proactive thermal-aware virtual machine allocation in hpc cloud datacenters," High Performance Computing (HiPC), 2012.

[8] W. Huang, J. Liu, B. Abali, D. Panda, "A case for high performance computing with virtual machines," International Conference on Supercomputing, ICS 2006.

[9] A. Reuther, P. Michaleas, A. Prout, J. Kepner, "HPC-VMs: Virtual machines in high performance computing systems," IEEE Conference on High Performance Extreme Computing (HPEC), 2012.

[10] S. Che, M. Boyer, J. Meng, D. Tarjan, J. Sheaffer, S. Lee, K. Skadron, "Rodinia: A benchmark suite for heterogeneous computing," IEEE International Symposium on Workload Characterization (IISWC) 2009.

[11] S. Hong, H. Kim, "An integrated GPU power and performance model," International Symposium on Computer Architecture (ISCA), 2010.

[12] R. Creasy, "The origin of the VM/370 time-sharing system," IBM Journal of Research and Development, 1981.

[13] http://ark.intel.com/Products/VirtualizationTechnology

[14] http://wiki.xen.org/wiki/Xen_VGA_Passthrough

[15] "Flexible, low power microservers for lightweight scale-out workloads," Intel white paper, 2013.

[16] R. Bianchini, R. Rajamony, "Power and energy management for server systems," Computer, volume: 37, issue: 11, 2004.

[17] L. Barroso, U. Hölzle, "The datacenter as a computer: An introduction to the design of warehouse-scale machines," Synthesis Lectures on Computer Architecture # 6. 2009.

[18] http://www.nvidia.co.uk/object/product_tesla_M2050_M2070_uk.html.

[19] D. Meisner, T. Wenisch, "Peak Power Modeling for Data Center Servers with Switched-Mode Power Supplies," International Symposium on Low-Power Electronics and Design (ISLPED), 2010.

[20] X. Fan, W. Weber, L. Barroso, "Power Provisioning for a Warehouse-sized Computer," International symposium on Computer architecture (ISCA), 2007.

[21] R. Nathuji, C. Isci, E. Gorbatov, "Exploiting platform heterogeneity for power efficient data centers," International Conference on Autonomic Computing (ICAC), 2007.

[22] A. Gupta, D. Milojicic, "Evaluation of HPC applications on cloud," Technical Reports, HP Laboratories, HPL-2011-132.