2015 44th International Conference on Parallel Processing

Profiling and Understanding Virtualization Overhead in Cloud

Liuhua Chen, Shilkumar Patel, Haiying Shen and Zhongyi Zhou
Department of Electrical and Computer Engineering
Clemson University, Clemson, South Carolina 29634
Email: {liuhuac, shilkup, shenh, zhongyz}@clemson.edu

***Abstract*—Virtualization is a key technology for cloud datacenters to implement infrastructure as a service (IaaS) and to provide flexible and cost-effective resource sharing. It introduces an additional layer of abstraction that produces resource utilization overhead. Disregarding this overhead may cause serious reduction of the monitoring accuracy of the cloud providers and may cause degradation of the VM performance. However, there is no previous work that comprehensively investigates the virtualization overhead. In this paper, we comprehensively measure and study the relationship between the resource utilizations of virtual machines (VMs) and the resource utilizations of the device driver domain, hypervisor and the physical machine (PM) with diverse workloads and scenarios in the Xen virtualization environment. We examine data from the real-world virtualized deployment to characterize VM workloads and assess their impact on the resource utilizations in the system. We show that the impact of virtualization overhead depends on the workloads, and that virtualization overhead is an important factor to consider in cloud resource provisioning. Based on the measurements, we build a regression model to estimate the resource utilization overhead of the PM resulting from providing virtualized resource to the VMs and from managing multiple VMs. Finally, our trace-driven real-world experimental results show the high accuracy of our model in predicting PM resource consumptions in the cloud datacenter, and the importance of considering the virtualization overhead in cloud resource provisioning.**

I. Introduction

Virtualization is a key technology for cloud datacenters to implement infrastructure as a service (IaaS) and to provide flexible and cost-effective resource sharing [1], [2], [3]. For example, Amazon EC2 cloud service [1] uses Xen virtualization [4] to support multiple virtual machine (VM) instances on a single physical machine (PM). Previous VM placement and migration works [5], [6], [7], [8] assume that the utilization of a particular resource (e.g., CPU, memory) in a PM equals the sum of the utilizations of this resource of its hosted VMs, which is not always true. The virtualization of hardware resources introduces extra resource overhead (called *virtualization overhead*) to the PM because the VM computing and data transfer processes involve other system components (e.g., device driver domain and hypervisor). The behaviors of an application running in a virtual environment and a non-virtualized environment can differ markedly and in surprising ways [9]. As shown in Figure 1, in Xen, the device driver domain (usually Dom0) manages the physical devices (e.g., hard disk drives and network interface cards (NICs)). Rather than directly communicating with the physical devices in the non-virtualized environment, the guest VM communicates with the physical devices via virtual interfaces (VIFs) within Dom0, which generates an additional computational overhead in the Dom0 CPU. Virtualization overhead is also caused by hypervisor, which is responsible for trapping VM activities and CPU scheduling among VMs. It traps every I/O request from the guest VM and schedules multiple VMs co-located in a PM (i.e., determines when to allocate what amount of CPU to which VM, and move the VM out of or into the CPU), which causes extra memory and CPU overhead.

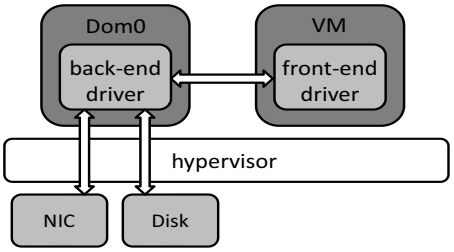


Figure 1. Overview of Xen architecture.

To improve the overall performance of the underlying infrastructure of cloud datacenters, we need an accurate estimation of virtualization overhead. It is also critical to accurately bill cloud customers and for a wide variety of management tasks to guarantee VM performance, such as resource allocation, admission control of new VMs and VM migration.

Knowing the actual resource utilizations helps accurately allocate the amount of resources in a PM to VMs, avoid mistakenly adopting new VMs in the case of insufficient resource, and migrate VMs out of a PM to release load. Some research works study the performance overhead of applications when they are shifted from native systems to virtualized environments [9], [10], [11]. They focused on studying the degradation of overall performance due to virtualization rather than the virtualization overhead. There are several researches [12], [13], [14] that focus on studying Dom0 CPU utilization as a result of either the I/O-intensive or bandwidth-intensive workloads running in the VM. However, these works do not give a comprehensive study on diverse VM workloads (e.g., CPU–intensive,

0190-3918/15 $31.00 © 2015 IEEE
DOI 10.1109/ICPP.2015.12

memory-intensive) or different PM resource utilizations (e.g., bandwidth, I/O). These works also neglect the CPU utilization of the hypervisor, which affects the accuracy of PM CPU utilization estimation. Further, they pay little attention to the influence of co-located VMs in a PM on the virtualization overhead.

In this paper, we aim to comprehensively characterize the virtualization overhead on the PM introduced by VM instances and to understand the impact of virtualization on the PM performance. We conduct measurements on different resource consumptions of the guest VMs and the corresponding virtualization overhead in the Dom0, hypervisor and PM in the Xen virtualized environment. We run CPU-intensive, memory-intensive, I/O-intensive and bandwidth-intensive benchmarks with different resource utilization degrees on VMs in different scenarios with different number of VMs hosting in a PM. We also investigate the influence of co-located VMs in a PM on the virtualization overhead. Through in-depth measurement analysis, we have a better understanding of the set of key factors that lead to the resource utilization overhead of the PM.

To the best of our knowledge, this is the first comprehensive study on resource utilization overhead in Xen virtualized environment.

We summarize our contributions as follows:

- We introduce a measurement method for automatic and synchronized monitoring on different resource consumption in a virtualized system. We create different workloads for studying the impact of different resource consumption of VMs on the virtualization overhead.
- We comprehensively study the virtualization overhead introduced by virtualization and VM co-location. We present the findings that were not previously observed.
- We propose a virtualization overhead estimation model to estimate virtualization overhead in Dom0, the underlying hypervisor and the PM.
- Our trace-driven and real-world experimental results show that the model can accurately estimate PM resource utilizations. We also improve a VM placement algorithm based on the model and perform experiments to show that considering virtualization overhead in resource management can help improve application performance.

The rest of this paper is organized as follows. Section II presents the related work. Section III introduces our measurement methods and the workloads for measurement. Section IV presents the measurement and the analysis on the virtualization overhead in various scenarios. Section V presents the virtualization overhead estimation models. Section VI evaluates the accuracy of the virtualization overhead models in predicting PM resource utilizations when its VMs run an enterprise application, and its effectiveness in guiding VM placement to improve VM performance. Section VII concludes this work with remarks on our future work.

II. Related Work

Application performance and resource consumption in virtualized environments can be very different from its performance and usage profile on native hardware. Some research works study the performance degradation of applications when they are shifted from native systems to virtualized environments [9], [10], [11]. Menon *et al.* [9] measured Xen's performance degradation on network throughput for network I/O device virtualization. Shea *et al.* [10] performed a measurement and analysis to reveal that the network I/O performance variation of a VM in various environments. Gulati *et al.* [11] measured disk workload characteristics and performance metrics in a consolidated virtualized environment. All these works focus on the application performance impact of both workload virtualization and consolidation, while they pay little attention to the specific resource utilization overheads from different resource workloads.

Some researches [12], [13], [14] focus on studying the virtualization overhead of VM resource utilizations or the PM resource utilizations. Apparao *et al.* [12] measured CPU and network I/O performance in a Xen virtualized environment and compared them to those from the native Linux machine. They provided a detailed architectural characterization of network intensive workload but did not measure other workloads such as CPU- or memory-intensive workloads. Mei *et al.* [13] conducted performance measurement study of network I/O applications in virtualized cloud to understand the CPU resource sharing across VMs running on a single PM. Cherkasova *et al.* [14] measured the CPU overhead in Dom0 caused by I/O processing on behalf of a particular VM. They regarded Dom0 CPU utilization as the CPU overhead of running an application in virtualized environment but neglected the CPU overhead in Xen hypervisor. The above works pay little attention to the resources other than CPU utilization on PM, such as memory, I/O and bandwidth. Moreover, they do not give a comprehensive study on diverse VM workloads except the I/O and bandwidth intensive workloads. Furthermore, they pay little attention to the resource utilization overhead correlated to multiple co-located VMs in the PM.

III. Measurement Method and Workloads

A. Measurement Methods and Tools

There are several measurement tools associated with Xen that can be directly used to measure the resource utilizations of its guest VMs. Table I shows the tools and what they can and cannot measure. However, none of them can concurrently measure different metrics (i.e., CPU, memory, bandwidth and disk I/O utilization) without introducing extra resource consumption (on VMs or Dom0), which however is critical for the accurate virtualization overhead study. Therefore, we developed a script that incorporates different tools for different metrics for automatic and synchronized execution

Table I
FEATURES OF MEASUREMENT TOOLS.

tool	VM				Dom0				PM/hypervisor			
	cpu	mem	i/o	bw	cpu	mem	i/o	bw	cpu	mem	i/o	bw
xentop[15]	Y+	-	Y+	Y+	Y+	-	Y+	Y+	-	-	-	-
top[16]	Y*	Y*+	-	-	Y	Y+	-	-	-	-	-	-
mpstat[17]	Y*	-	-	-	-	-	-	-	Y+	-	-	-
ifconfig[18]	-	-	-	Y*	-	-	-	-	-	-	-	Y+
vmstat[17]	Y*	Y*	Y*	-	-	Y	-	-	Y	-	Y+	-

Y: can, -: cannot, *: need to run inside the VM, +: included in our script

of measurements. The CPU, I/O and network bandwidth utilization information of both Dom0 and the guest VMs are obtained from executing *xentop* in Dom0, while the memory utilization is obtained by executing Linux *top* command in each corresponding VM. The CPU utilization of the Xen hypervisor is obtained by running *mpstat* in Xen. In order to measure host PM resource utilization metrics, we use *vmstat* and *ifconfig* in Dom0 to measure I/O and network bandwidth utilizations, respectively. The memory utilization of the host PM is estimated by the summation of the memory utilizations in Dom0 and the guest VMs. Measurement interval and inspection time can be tuned by the input parameters of this script.

B. Measurement Workloads

To study the resource utilization overhead of applications in virtualized environments, previous works either use testing tools [13], [14], [12] (e.g., *httperf* [19] and *Iperf* [20]) or use self-developed applications [21] (e.g., calculating Fibonacci series) to generate benchmark workload in the VMs. However, these benchmarks cannot provide a workload that has high utilization on a sole resource and low overhead on other resources (e.g., CPU-intensive workload has low overhead on other resources). However, such a workload is important for understanding the impact of different resource utilizations on the virtualization overhead. To handle this problem, as shown in Table II, we used *lookbusy* [22] to create three CPU-, memory-, I/O-intensive workloads and used Linux command *ping* to generate network bandwidth (BW) workload. Each workload has 5 levels of the resource utilizations. We remove "-intensive" in the workload/benchmark names for simplicity.

Table II
OUR GENERATED BENCHMARKS FOR MEASUREMENT STUDY.

Workload	Workload intensity				
CPU-intensive (%)	1	30	60	90	99
MEM-intensive (Mb)	0.03	5	10	20	50
I/O-intensive (blocks/s)	15	19	27	46	72
BW-intensive (Mb/s)	0.001	0.16	0.32	0.64	1.28

C. Measurement Environment and Reported Results

To study the relationship between resource consumption of the VMs and of the underlying PMs in Xen, we deployed the XenServer 6.2 [23] virtualization infrastructure in a local cluster, which consists of 7 PMs. Each PM in the cluster has the following configuration: one 2.66 GHz Quad Core Xeon CPU, 2 GB main memory, 60 GB SATA hard disks and one Gigabit network card. Since we had full control of this cluster, we made sure that there was no additional workload on the cluster during our experiments. We used the default settings from Xen in all our experiments. During the experiment, we used our shell script to concurrently measure the resource (CPU, memory, I/O and bandwidth) utilizations of all the VMs, Dom0 and the hypervisor (or PM) every second for 2 minutes and we finally report the average of these 120 measurements. We carried out the same experiment in different PMs and the results are the same. Then, we report the results from one PM.

Table III
DEFINITION OF UTILIZATION OVERHEAD.

Metrics	Resource util. overhead	Intensity workload			
		CPU	MEM	I/O	BW
CPU	$\|Dom0\|+\|hypervisor\|$	$\surd$			$\surd$
I/O	$\|\sum VM_{io} - PM_{io}\|$			$\surd$	
BW	$\|\sum VM_{bw} - PM_{bw}\|$				$\surd$
MEM	$\|\sum VM_{mem} - PM_{mem}\|$				

We selected the results with obvious resource utilization overhead (as marked in Table III) to report. We sum the CPU utilizations of Dom0, hypervisor and the guest VMs to indirectly calculate the CPU utilization of the PM. The CPU utilizations of Dom0 and VM are in percentage of virtual CPU (VCPU), while the utilization of hypervisor is in percentage of real CPU. For simplicity, we use CPU for both. In all our memory-intensive experiment, the CPU utilizations of Dom0 and hypervisor have constant values of 16.8% and 3.0%, respectively. Dom0's I/O and bandwidth utilizations are always zero, and the PM's I/O and bandwidth utilizations have constant values of 18.8 blocks/s and 254 bytes/s, respectively. Therefore, we do not show the results from the memory benchmark.

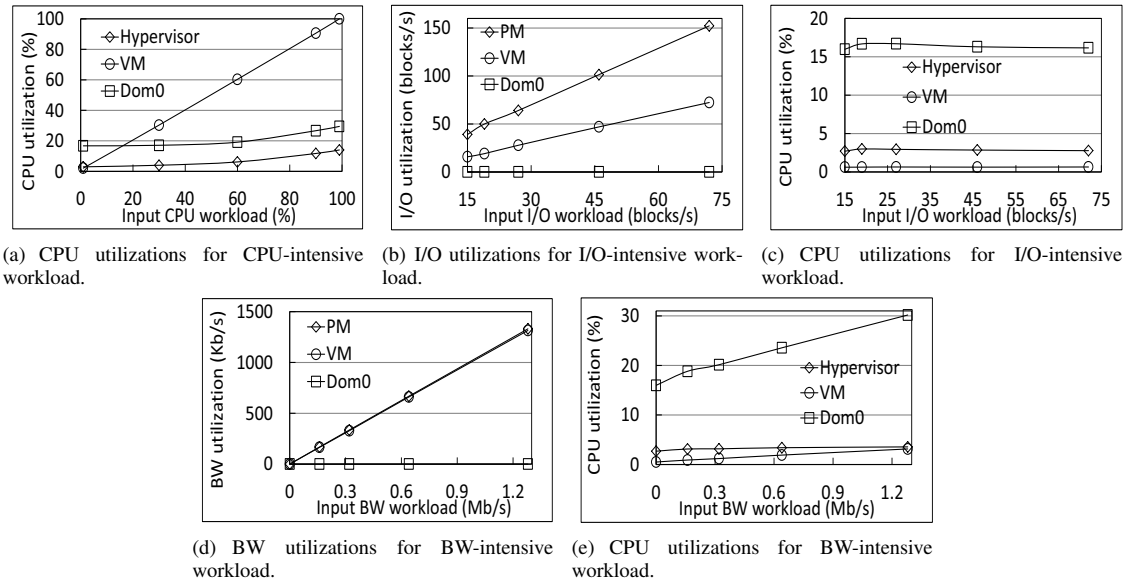
IV. VIRTUALIZATION OVERHEAD MEASUREMENT AND ANALYSIS

We ran the different benchmarks with different intensity degrees in three scenarios: i) a single VM, ii) two VMs, and iii) four VMs hosting on a PM.

A. Virtualization Overhead of One VM

In this section, we conduct measurement to answer the following questions: i) Is there any extra resource utilization in the PM to support the VM running. ii) What is the relationship between VM resource utilizations and the resource utilizations of Dom0, hypervisor and the underlying PM. iii) What is the magnitude of the virtualization overhead.

Figure 2(a) shows the measured CPU utilizations of the VM, Dom0, and the hypervisor versus the CPU workload in the VM. We define *increase rate* in a figure as $\Delta Y/\Delta X$,



(a) CPU utilizations for CPU-intensive workload.

(b) I/O utilizations for I/O-intensive workload.

(c) CPU utilizations for I/O-intensive workload.

(d) BW utilizations for BW-intensive workload.

(e) CPU utilizations for BW-intensive workload.

Figure 2. Resource utilizations for one VM.

which means the increase of Y value for each unit increase of X value. We see that as the workload intensity increases, the CPU utilization in Dom0 increases from a background utilization 16.8% to 29.5% with the *increase rate* growing from 0.01 to 0.31. The CPU utilization of the hypervisor increases from 3% to 14% with the *increase rate* growing from 0.04 to 0.26. A VM with a higher CPU utilization means an increasing need for the hypervisor in scheduling and for the Dom0 to respond to control signals, which increase their CPU utilizations.

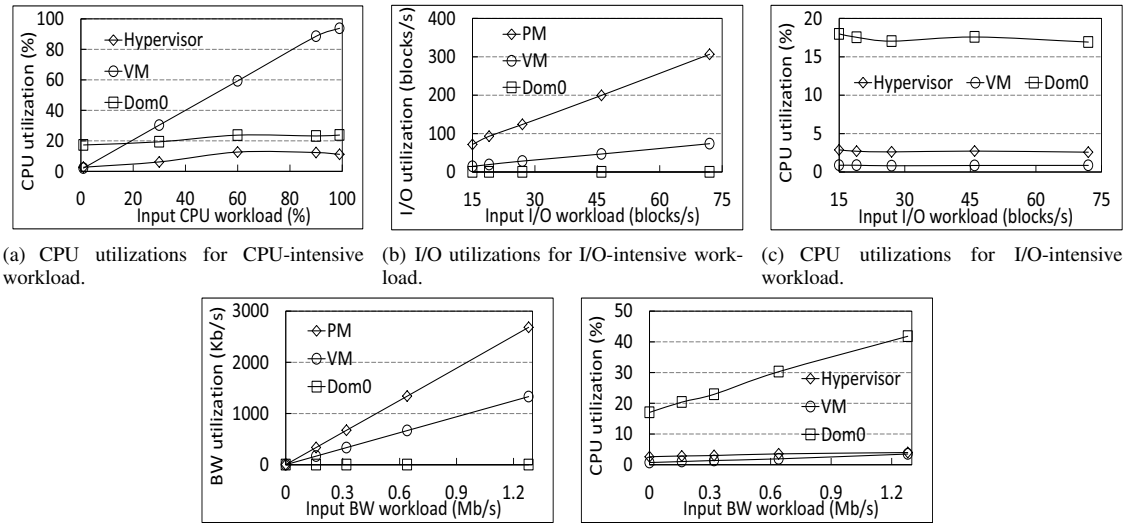
Figure 2(b) shows the I/O utilizations of the VM, Dom0 and the PM versus the I/O workload. The zero I/O utilization in Dom0 indicates that the VM's I/O workload does not impose extra I/O requirement on Dom0 since Dom0 is only responsible for scheduling the I/O request from the VM. The PM's I/O utilization is nearly twice as much as the VM's I/O utilization and they increase at a similar trend as the workload intensity increases. It indicates that the guest VM I/O workload introduces I/O utilization overhead on the host PM, with an amount of approximately the same as the VM's I/O utilization. This is because the virtual disk of the VM is actually striped across many different disks and a single read or write by the guest VM may involve several reads or writes. Figure 2(c) shows the measured CPU utilizations of the VM, Dom0 and the hypervisor as a result of the increased I/O workload in the guest VM. All the three measured CPU utilizations remain stable under varying I/O intensity. Because the default configuration of the VM has a maximum I/O capacity limit of about 90 blocks/s, which is relatively small and hence does not cause obvious CPU utilization changes in Dom0.

Figure 2(d) shows the bandwidth utilizations in Kb/s of the guest VM, Dom0 and the host PM, with an increasing bandwidth workload on the VM. The zero utilization in Dom0 indicates that the bandwidth workload in the guest VM does not impose bandwidth utilization overhead on Dom0. The overhead of bandwidth utilization in the host PM is at the amount of nearly 400 bytes/s, which is negligible especially when the guest VM is undergoing bandwidth-intensive workload with high intensity degree.

Figure 2(e) presents the measured CPU utilizations of the guest VM, Dom0 and the hypervisor as a result of the increasing bandwidth workload in the guest VM. We see that the CPU utilization in the VM slightly increases from 0.5% to 3%. The CPU utilization of Dom0 increases from 16.0% to 30.2% with a constant *increase rate* of 0.01, which indicates that the CPU utilization overhead in Dom0 increases with the bandwidth workload intensity in the guest VM. The CPU utilization overhead in Dom0 is caused by the need of processing network packets as the guest VM talks to the physical network interface via a VIF existing within Dom0. The CPU utilization of the hypervisor increases from 2.5% to 3.5% as a result of the CPU utilization increment in both guest VM and Dom0.

We summarize our observations for the single-VM scenario:

- CPU utilizations in both Dom0 and hypervisor have a background rate due to virtualization and they increase with VM CPU utilization at the *increase rates* of [0.01, 0.31] and [0.04, 0.26], respectively.
- PM I/O utilization is slightly more than twice of the VM I/O utilization.
- CPU utilizations in Dom0 increases with a constant *increase rate* 0.01 while hypervisor's CPU remains constant as the VM bandwidth utilization increases.
- CPU utilizations in both Dom0 and hypervisor keep



(a) CPU utilizations for CPU-intensive workload.

(b) I/O utilizations for I/O-intensive workload.

(c) CPU utilizations for I/O-intensive workload.

(d) BW utilizations for BW-intensive workload.

(e) CPU utilizations for BW-intensive workload.

Figure 3. Resource utilizations for two VMs co-located in a PM.

nearly constant (16±0.3% and 2.8±0.1%) as VM I/O utilization increases (from 15 blocks/s to 72 blocks/s).

- PM bandwidth utilization equals to the VM bandwidth utilization, with near zero overhead.

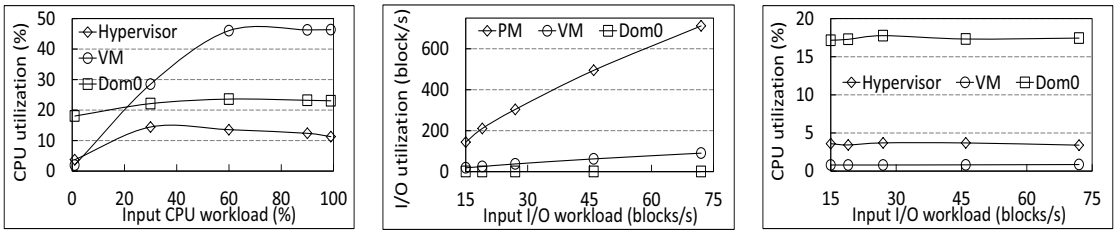
B. Virtualization Overhead of Two and More VMs

Co-located VMs in a PM is one of the attractive features that the virtualization technique brings to the cloud computing. In order to investigate the effect of co-located VMs on the virtualization overhead, we carry out another experiment to study the relationship of resource utilizations of the co-located VMs, Dom0 and the hypervisor (or PM). We launched the four micro benchmarks with increasing workload intensity simultaneously in each VM, and then measured all the metrics. Since the measurements of all VMs are exactly the same, we only show the measurement of one VM. This experiment aims to answer the following questions: i) How are the resource utilizations of Dom0 and the PM (or hypervisor) affected by co-located VMs. ii) How are the resource utilizations of the VMs affected by co-location. iii) What is the magnitude of the virtualization overhead. The following results confirm all our observations from the previous single VM test, so we do not repeat the same observations.

Figure 3 and Figure 4 show the resource utilizations when the PM hosts two VMs and four VMs, respectively. Figures 3(a) and 4(a) show the CPU utilizations with different CPU workloads. We see that the CPU utilization of the VM increases with CPU workload intensity, but the utilization is not exactly equal to the input workload. The guest VM consumes 95% in Figure 3(a) and 47% in Figure 4(a) when the workload is 100%. This result indicates that the CPU utilization in the guest VM decreases due to the co-location of VMs, in which the resources are shared and none of the VMs reaches 100% of CPU utilization. In both figures, the CPU utilizations in Dom0 slightly increase at first with the load in the VMs and keep stable (or slightly decreases) due to the inadequate of available CPU resource in the PM. Similarly the CPU utilizations in the hypervisor increase as the CPU workload increases, but they become stable when the CPU resource consumption is saturated.

Figure 3(b) and Figure 4(b) show the I/O utilizations with different I/O workloads. The I/O utilization of the PM is more than twice of the sum of the utilizations of its guest VMs, which indicates that extra I/O resource in the host PM is required for I/O resource provisioning for the co-located guest VMs. Figure 3(c) and Figure 4(c) show the CPU utilizations with different I/O workloads. All the utilizations remain relatively stable under varying I/O workload; 17.4% and 0.84% for Dom0 and VM in both figures and 2.7% and 3.5% for the hypervisor in two figures, respectively. This result is similar to the result in the previous one VM experiment. We notice that the CPU utilization overhead correlates to VM co-location in Dom0 is very small (about 2% extra utilization compared to Figure 2(c)). Due to the limit of VM I/O capacity, the maximum I/O throughput in our experiment is 360 Kb/s from 4 VMs, which is not high enough to obviously change Dom0 CPU utilization.

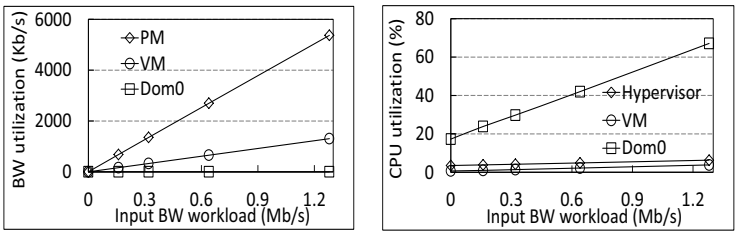
Figure 3(d) and Figure 4(d) show the bandwidth utilizations with different bandwidth workloads. We see that Dom0 does not consume bandwidth resource, while PM has bandwidth utilization approximately equal to the sum of its guest VMs' bandwidth utilizations with a small overhead $\frac{|PM_{bw}-\sum VM_{bw}|}{PM_{bw}}$=3%. Figure 3(e) and Figure 4(e) show the CPU utilizations with different bandwidth workloads. The



(a) CPU utilizations for CPU-intensive workload.

(b) I/O utilizations for I/O-intensive workload.

(c) CPU utilizations for I/O-intensive workload.



(d) BW utilizations for BW-intensive workload.

(e) CPU utilizations for BW-intensive workload.

Figure 4. Resource utilizations for four VMs co-located in a PM.

CPU utilization in Dom0 increases from 17.1% to 41.8% in Figure 3(e) and from 17.3% to 67.1% in Figure 4(e) as the bandwidth intensity increases to the maximum value. Both figures have *increase rates* of 0.01, which is consistent with Figure 2(e). This result confirms the conclusion that the bandwidth workload in guest VMs imposes extra CPU utilization in Dom0. Note that the slope of Dom0 in Figure 4(e) is twice as much as that in Figure 3(e), because it has twice input bandwidth intensity (four VMs compared to two VMs). The CPU utilization of the hypervisor increases from 2.6% to 4.0% in Figure 3(e), and from 3.5% to 6.3% in Figure 4(e) as bandwidth workload increases. Also, both figures exhibit *increase rates* of 0.0005. Since the hypervisor is responsible for the management of multiple guest VMs, its CPU utilization increases with the bandwidth workload intensity and the number of VMs.

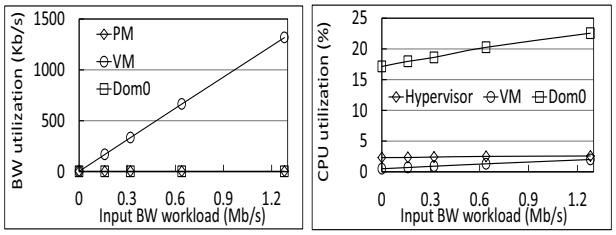
In the above experiments for bandwidth workloads, VMs communicate with VMs in other PMs. It is possible that one VM has frequent network communication with another VM residing in the same PM, then the bandwidth utilizations are within the PM. In order to study how the resource utilizations correlate to network transmission between co-located VMs, we carry out another experiment. We use *ping* in one VM (VM_1) to ping 64Kb size packet to the other VM (VM_2) in the same PM. Figure 5 shows the measured resource utilizations of VM_1, Dom0 and the host PM with different intra-PM bandwidth workloads. Figure 5(a) shows that the bandwidth utilizations of Dom0 and the PM are zero. This result indicates that the bandwidth workload between the guest co-located VMs does not actually consume physical bandwidth resource, because the packets sent from VM_1 are redirected to VM_2 inside the PM, and do not need to occupy the network interface hardware of the host PM. Figure 5(b) shows that the CPU utilization of Dom0 increases with guest VM bandwidth workload at an *increase rate* of 0.002. It is 5X less compared to the *increase rate* (0.01) in Figures 2(e), 3(e) and 4(e). The Dom0 CPU increment is caused by the requirement for processing network packets from the guest VMs. Although the network communication between the two co-located VMs does not introduce bandwidth utilization to their host PM, it still imposes CPU overhead on Dom0 with an *increase rate* 5X less than the inter-PM communications between VMs.

We summarize our observations for the multi-VM scenario:

- CPU utilizations in Dom0 and hypervisor increase with VM CPU utilization and stay at constants (23.4% and 12.0%, respectively), due to the insufficient CPU resource in the PM to support multiple VMs.
- PM bandwidth utilization has 3% overhead ($\frac{|PM_{bw}-\sum VM_{bw}|}{PM_{bw}}$) compared to the sum of bandwidth utilizations of its guest VMs in inter-PM communication.
- The bandwidth utilizations of co-located VMs due to their network I/O communication do not generate bandwidth utilization of their hosting PM.
- The network I/O communication between two co-located VMs leads to the increase of CPU utilization in Dom0 at an *increase rate* of 0.002, which is 5X less than the inter-PM communications between VMs.

V. Modeling Virtualization Overhead

In this section, we describe our models to characterize the relationship between VM resource utilizations and virtualization overhead (on different resources) of a PM when it holds a single VM and multiple VMs.



(a) Bandwidth utilizations. (b) CPU utilizations.

Figure 5. Resource utilizations for intra-PM bandwidth-intensive workload.

A. Virtualization Overhead from a Single VM

To find the relationship between the virtualization overhead and resource usage of a single VM, we use our previously collected data for the resource utilizations on PM and VMs from the Xen server when the VMs run the micro benchmarks. Using the gathered data from repeated experiments, we derive a set of equations to calculate the virtualization overhead as a linear combination of different metrics as below.

$$\hat{M}_c = a_o + a_c M_c + a_m M_m + a_i M_i + a_n M_n \quad (1)$$

where M_c, M_m, M_i and M_n are utilization values of the VM gathered for CPU, memory, I/O and network bandwidth, respectively. $\hat{M}_c$ is a measured CPU utilization of the PM. a_c, a_m, a_i and a_n are corresponding coefficients for different metrics, and a_o is a constant denoting the resource utilization of the guest VM without running any benchmarks, that is, the resource consumption of the guest operating system (OS).

We use $\mathbf{a}_c = [a'_o, a'_c, a'_m, a'_i, a'_n]$ to denote the set of coefficients that describes the relationship. By applying a regression method [24] to the gathered utilization data, we derive $\mathbf{a}_c$ that minimize error $e = \sqrt{\sum_j (\hat{M}'^j_c - \hat{M}^j_c)^2}$, where the superscript j is the sequence number of different sets of measurements. Accordingly, the approximated solution for Equ (1) is $\hat{M}'^j_c = a'_o + a'_c M^j_c + a'_m M^j_m + a'_i M^j_i + a'_n M^j_n$.

Similarly, we derive the relationship between VM different resource utilizations and PM memory, I/O and network bandwidth utilizations (denoted by $\hat{M}_m$, $\hat{M}_i$ and $\hat{M}_n$), respectively. We use $\mathbf{a}_m$, $\mathbf{a}_i$ and $\mathbf{a}_n$ to denote each set of coefficients that describes each of the relationships. By applying the regression method on the gathered VM utilization metrics, we derive $\mathbf{a}_m$, $\mathbf{a}_i$ and $\mathbf{a}_n$. Therefore, given a set of VM resource utilization measurements $\mathbf{M} = [M_c, M_m, M_i, M_n]^T$, we estimate the approximated PM resource utilization by

$$\hat{\mathbf{M}} = \mathbf{a}\mathbf{M} \quad (2)$$

where $\hat{\mathbf{M}} = [\hat{M}_c, \hat{M}_m, \hat{M}_i, \hat{M}_n]^T$ is the estimated PM resource utilization. $\mathbf{a} = [\mathbf{a}_c^T, \mathbf{a}_m^T, \mathbf{a}_i^T, \mathbf{a}_n^T]^T$ is the set of coefficients that describes the characteristics of virtualization overhead of the system.

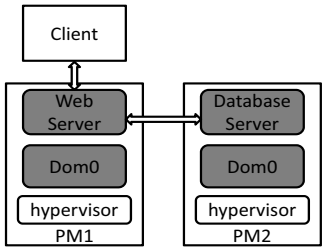


Figure 6. Experiment setup.

B. Virtualization Overhead from Multiple VMs

When multiple VMs are co-located in one PM, the resulting PM resource utilizations are not always equal to the sum of the VM utilizations. Extra physical resource is required. For example, Section IV shows that extra computing resource is required for managing multiple VMs. The PM resource utilizations could be less than the sum of the VM utilizations due to intra-PM communication between VMs in the same PM. Below, we present the model of resource utilization overhead of co-located VMs in this section.

Suppose there are N VMs (VM_1, VM_2, ...) co-located in one PM. Each VM has resource utilization profile $\mathbf{M}_k$=$[M_{ck}, M_{mk}, M_{ik}, M_{nk}]^T$, where $1 \le k \le N$ is the VM identification. PM has resource utilization $\hat{\mathbf{M}}$=$[\hat{M}_c, \hat{M}_m, \hat{M}_i, \hat{M}_n]^T$. Since the resource utilizations of Dom0 and the hypervisor are caused by the workload resource usages of all the VMs. We model the resource consumption of the PM as a combination of the VMs and an overhead to represent the synthesized effect of co-located VMs as below.

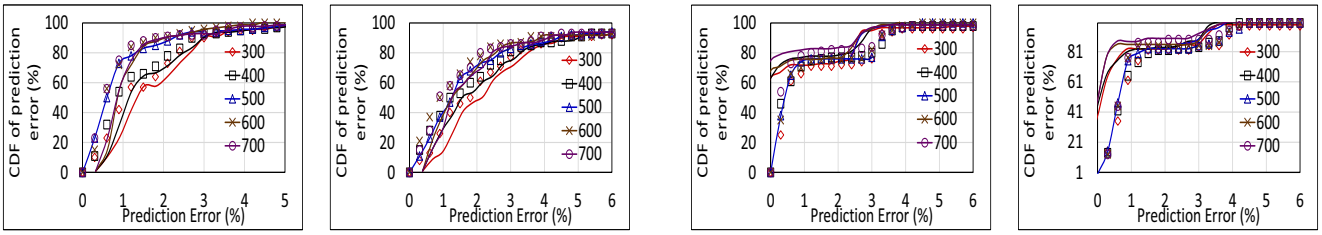
$$\hat{\mathbf{M}} = \mathbf{a}(\sum_{n=1}^{N} \mathbf{M}_n) + \alpha(N) \cdot \mathbf{o}(\sum_{n=1}^{N} \mathbf{M}_n) \quad (3)$$

Similar to $\mathbf{a}$, $\mathbf{o} = [\mathbf{o}_c^T, \mathbf{o}_m^T, \mathbf{o}_i^T, \mathbf{o}_n^T]^T$ is a set of the coefficients that describes the relationship between resource utilization overhead from co-located VMs and the resource utilizations of the VMs. $\alpha(N)$ is a coefficient determined by N, which can be simply derived from measurement experiments. When N=1, there is no overhead for co-located VMs and $\alpha(N)$=0; when N=2, there are two co-located VMs, $\alpha(N)$=1, and $\hat{\mathbf{M}} = \mathbf{a}(\mathbf{M}_1 + \mathbf{M}_2) + \mathbf{o}(\mathbf{M}_1 + \mathbf{M}_2)$. The combination effect of placing multiple VMs on a single PM can be quite complex due to the diverse resource consumption features of multiple types of VMs. As shown in Section IV-B, the resource consumption overhead caused by multiple VMs exhibits a near linear trend. Therefore, we assume that the coefficient $\alpha(N)$ is a linear function of N to simplify the analysis. Similarly, using the regression method, we derive $\mathbf{a}$ and $\mathbf{o}$, which can be used to estimate $\hat{\mathbf{M}}$ based on Equ. (3).

VI. Performance Evaluation

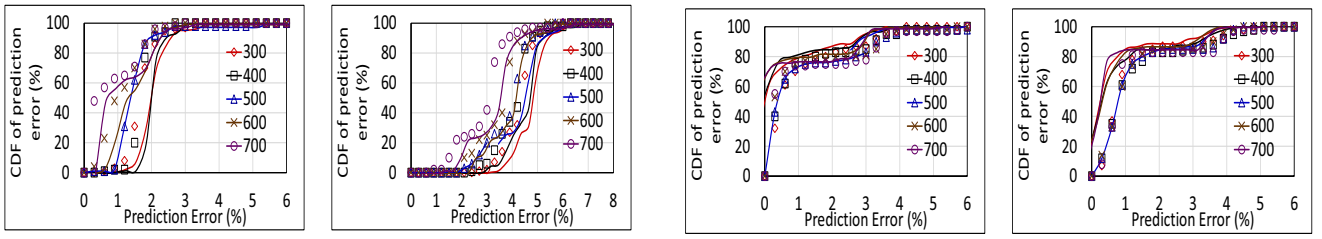
A. Overhead Prediction Accuracy

To test the accuracy of our virtualization overhead prediction model, we first derived this model from the trace



(a) PM_1 CPU prediction. (b) PM_2 CPU prediction. (c) PM_1 bandwidth prediction. (d) PM_2 bandwidth prediction.

Figure 7. Resource utilization prediction for a PM hosting one VM.



(a) PM_1 CPU prediction. (b) PM_2 CPU prediction. (c) PM_1 bandwidth prediction. (d) PM_2 bandwidth prediction.

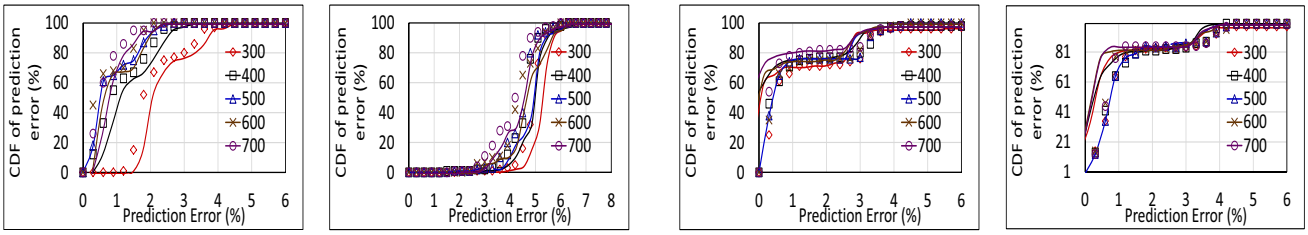
Figure 8. Resource utilization prediction for a PM hosting two VMs.

of resource utilizations in our micro benchmark study using the method introduced in Section V. We used the model to predict the resource utilizations of Xen servers, which hosted VMs running the RUBiS web applications [25]. The actual PM CPU utilization was calculated as the sum of the measured CPU utilizations from all the domains (e.g., Dom0 and VMs) and the hypervisor, while the actual bandwidth utilizations of the PMs were directly measured in Xen. We predicted the PM CPU utilization based on the predicted Dom0 and hypervisor utilizations. Since RUBiS is a network bandwidth-intensive application, we also predicted the bandwidth utilizations of the PMs. We then compared the predicted resource utilizations and the measured resource utilizations to evaluate the prediction accuracy. Due to the limited space, we do not show the prediction of other resources, which has similar accuracy result as CPU and bandwidth.

We carried out the first experiment to evaluate the model of predicting resource utilizations from a single VM. The RUBiS web application was configured to run in two VMs. As shown in Figure 6, we configured the RUBiS with a web server front-end running in VM_1, which was located in PM_1, and a database server running in VM_2, which was located in PM_2. We used a third machine to simulate the benchmark client and generated the requests for the front-end web server. We created a variable rate workload for RUBiS by increasing the number of clients over a ten minute period. The system was loaded between 300 and 700 simultaneous clients. This workload was repeated three times. Since the results from the three times experiments are similar, we show the results from one of them. We recorded VM resource utilizations every second and made predictions for every measurement for a 10 minute interval.

We evaluated the accuracy of the prediction by examining its prediction error, which is calculated by $\frac{|p-m|}{m}$, where p is the predicted amount while m is the measured resource utilization. Figure 7(a) and Figure 7(b) show the cumulative distributed function (CDF) of the prediction errors for the PM CPU utilization prediction. The different curves in the figures represent different number of clients (from 300 to 700) for RUBiS. We see that 90% of the predictions for PM CPU utilizations have prediction errors smaller than 3% in PM_1, and 4% in PM_2. The reason for the difference of the prediction errors in two PMs is that their hosted VMs are playing different roles in the application and hence have different resource utilizations. The prediction errors decrease as the number of simulated clients increases because more clients place heavier load on the RUBiS web server (VM_1) and lead to a larger denominator in $\frac{|p-m|}{m}$. The prediction errors in PM_2 (hosting database server) are higher than PM_1 (hosting web server) because the database server has a lower bandwidth utilization than the web server, which means that the database server imposed less CPU utilization overhead on the Dom0 and the hypervisor and results in relatively lower PM CPU utilizations. Some errors may be caused by irregularities in the data used as input to the model. Figure 7(c) and Figure 7(d) show the CDF of PM bandwidth prediction errors in the two PMs. We see that 90% of the predictions, for both PM_1 and PM_2 bandwidth utilizations, have prediction errors smaller than 4%, and about 80% of the predictions have prediction errors smaller than 1%.

In order to validate the virtualization overhead model for two co-located VMs, we created two sets of independent RUBiS applications by placing two RUBiS web servers in PM_1 and two RUBiS database servers in PM_2. We varied the workload by adjusting the number of emulated clients



(a) PM_1 CPU prediction. (b) PM_2 CPU prediction. (c) PM_1 bandwidth prediction. (d) PM_2 bandwidth prediction.

Figure 9. Resource utilization prediction for a PM hosting multiple VMs.

from 300 to 700, and then measured the resource utilizations in the two PMs. Figure 8(a) and Figure 8(b) show the prediction errors of predicting PM CPU in the two PMs. 90% of the predictions for PM CPU utilizations have prediction errors under 2% in PM_1, and 5% in PM2. The prediction errors in PM_2 are higher than PM_1 due to the same reasons mentioned before. Figure 8(c) and Figure 8(d) present the CDF of the PM bandwidth prediction errors. We see that 90% of the predictions for PM bandwidth have within 3.5% prediction errors for both PM_1 and PM_2. Compared to PM CPU utilization prediction, PM bandwidth prediction has a higher accuracy, because co-located two VMs do not impose much overhead on PM bandwidth utilization, as indicated in the previous micro benchmark studies.

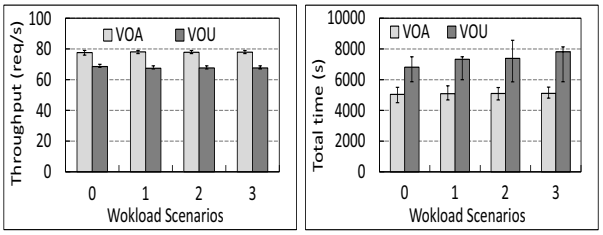
We further applied our model to predict the resource utilizations when the PMs are hosting more than two VMs. The system was configured to have two PMs with six VMs running three sets of RUBiS applications. Specifically, three RUBiS web servers ran in one PM and three RUBiS database servers ran in the other PM. Figure 9(a) and Figure 9(b) show the CDF of the PM CPU prediction errors. In Figure 9(a), 90% of the predictions have prediction errors smaller than 2%. From Figure 9(b), we see that most of the predictions for PM_2 have prediction errors around 4.5%. The predictions for PM_2 CPU utilizations have relatively higher prediction errors than the predictions for PM_1. This is because the workload in PM_2 is relatively lower than in PM_1 and leads to a smaller denominator in $\frac{|p-m|}{m}$. Figure 9(c) and Figure 9(d) show the prediction errors for PM bandwidth utilizations. 80% of the predictions have prediction errors within 1% for both PM_1 and PM_2. The prediction accuracy for the bandwidth utilization on the two PMs is similar since both PMs have high workloads.

B. Virtualization Overhead Aware Resource Provisioning

In this section, we deployed an experiment to show that concerning virtualization overhead in cloud resource provisioning is important. Properly placing VMs in PMs by considering their virtualization overhead can help better guarantee Service-level agreement (SLA) and improve performance of the VMs. In this experiment, we setup a scenario, in which a cloud provider deployed 5 identical VMs that have the same capacity configuration (1 VCPU, 256 MB memory, Debian Squeeze 6.0 OS) in the cloud. Two VMs corporately ran RUBiS, with the web front-end installed in one VM (VM_1) and the back-end database in the other (VM_2). The RUBiS system was loaded by serving 500 simultaneous clients. The other three VMs (VM_3-VM_5) did nothing and had nearly zero consumptions on CPU, I/O and bandwidth. Based on this scenario (denoted by 0), we test more scenarios by running *lookbusy* with 50% CPU utilization in one, two and all of the three VMs (VM_3-VM_5), denoted by 1, 2 and 3, respectively.

We implemented CloudScale [8], a system that employs online resource demand prediction to achieve robust resource provisioning inside the cloud. First, we deployed the 5 VMs to PMs in a random order, and then we used CloudScale to predict their resource utilizations one by one in order to find suitable PMs for hosting them. CloudScale predicted the resource utilizations of the 5 VMs in a random order and deployed them one by one based on the predictions. We compared the performance of RUBiS in CloudScale with and without the consideration of virtualization overhead in VM allocation, denoted by VOA and VOU, respectively. We repeated this VM placement process for 10 times.

Figure 10(a) compares the average throughput of RUBiS in VOA and VOU measured by the number of requests handled per second. The error bars indicate the 90th and 10th percentile among the 10 test results. We see that VOA achieves a stable throughput under every workload scenario, which is greater than that of VOU. This is because VOU does not consider the virtualization overhead when making VM placement decisions and the PM may become overloaded. VOU placed the first four VMs in a PM, then it predicted that the PM would not have sufficient memory resource for hosting the fifth VM, and placed it in another PM. When VOU was allocating the fourth VM, it did not realize that the remaining CPU resource in the PM was inadequate for the fourth VM since VOU ignores the extra CPU consumptions in Dom0 and the PM. The throughput was reduced as the RUBiS VMs were placed in a PM with exhausted CPU resource. The throughput for VOU further decreases as the workload in the VMs increases. Figure 10(b) shows the total time of RUBiS for processing the requests. VOU has a higher total time compared to VOA, because the RUBiS VMs cannot get sufficient CPU resource for processing incoming requests. The lack of CPU



(a) Average throughput. (b) Total time for simulation.

Figure 10. Performance of virtualization overhead aware VM placement.

resource for the RUBiS VMs is caused by inappropriate VM placement due to the neglect of virtualization overhead by the placement algorithm.

VII. Conclusions

We presented a comprehensive and in-depth study on the resource utilization overhead caused by virtualization on Xen, which includes the overhead of the server for providing virtualized resource for hosting VMs and for managing multiple VMs. We also proposed a virtualization overhead estimation model to study the relationship between VM resource utilization and the utilization overhead in Dom0, the hypervisor and the PM. Our trace-driven real-world experiments show that the proposed model can effectively characterize the different virtualization overhead. We also showed that using the proposed estimation model can help improve the performance of the VMs in the virtualized environment. In the future, we are interested in improving the model for estimating the resource utilization overhead for different types of VMs with diverse configurations, when they are co-located in a PM.

Acknowledgements

This research was supported in part by U.S. NSF grants NSF-1404981, IIS-1354123, CNS-1254006, CNS-1249603, and Microsoft Research Faculty Fellowship 8300751

References

[1] "Amazon Elastic Compute Cloud," http://aws.amazon.com/ec2/, [Accessed in Nov. 2014].

[2] "Microsoft Azure," http://www.windowsazure.com, [Accessed in Nov. 2014].

[3] "BEA System Inc." http://www.bea.com, [Accessed in Nov. 2014].

[4] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. L. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization." in *Proc. of SOSP*, 2003.

[5] T. Wood, P. J. Shenoy, A. Venkataramani, and M. S. Yousif, "Sandpiper: Black-box and gray-box resource management for virtual machines." *Computer Networks*, 2009.

[6] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers." in *Proc. of WWW*, 2007.

[7] L. Chen, H. Shen, and K. Sapra, "Distributed autonomous virtual resource management in datacenters using finite-markov decision process," in *Proc. of SOCC*, 2014, pp. 1–13.

[8] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, "Cloudscale: Elastic resource scaling for multi-tenant cloud systems." in *Proc. of SOCC*, 2011.

[9] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel, "Diagnosing performance overheads in the xen virtual machine environment," in *Proc. of VEE*, 2005.

[10] R. Shea, F. Wang, H. Wang, and J. Liu, "A deep investigation into network performance in virtual machine based cloud environments," in *Proc. of INFOCOM*, 2014.

[11] A. Gulati, C. Kumar, and I. Ahmad, "Storage workload characterization and consolidation in virtualized environments," in *Proc. of VPACT*, 2009.

[12] P. Apparao, S. Makineni, and D. Newell, "Characterization of network processing overheads in xen," in *Proc. of VTDC*, 2006.

[13] Y. Mei, L. Liu, X. Pu, and S. Sivathanu, "Performance measurements and analysis of network I/O applications in virtualized cloud," in *Proc. of CLOUD*, 2010.

[14] L. Cherkasova and R. Gardner, "Measuring CPU overhead for I/O processing in the xen virtual machine monitor," in *Proc. of ATC*, 2005.

[15] J. Fischbach, D. Hendricks, and J. Triplett, "Xentop," *Xen builtin Utility*, 2005.

[16] "Linux/Unix Command: top," http://linux.about.com/od/commands/l/blc mdl1_top.htm, [Accessed in Nov. 2014].

[17] "Sysstat," http://sebastien.godard.pagesperso-orange.fr/, [Accessed in Nov. 2014].

[18] "Linux/Unix Command: ifconfig," http://linux.about.com/od/commands/ l/blcmdl8_ifconfi.htm, [Accessed in Nov. 2014].

[19] "httperf," http://www.hpl.hp.com/research/linux/httperf/, [Accessed in Nov. 2014].

[20] "Iperf," https://iperf.fr/, [Accessed in Nov. 2014].

[21] T. Wood, L. Cherkasova, K. Ozonat, and P. Shenoy, "Profiling and modeling resource usage of virtualized applications," in *Proc. of Middleware*, 2008.

[22] "lookbusy," http://devin.com/lookbusy/, [Accessed in Nov. 2014].

[23] "Xenserver," http://www.citrix.com/products/ xenserver/overview.html, [Accessed in Nov. 2014].

[24] P. J. Rousseeuw, "Least median of squares regression," *JASA*, vol. 79, no. 388, pp. 871–880, 1984.

[25] "Rubis: Rice university bidding system," http://rubis.ow2.org/, [Accessed in Nov. 2014].