CONTRIBUTED PAPER

Managing Performance Overhead of Virtual Machines in Cloud Computing: A Survey, State of the Art, and Future Directions

This survey reviews the latest solutions for managing performance overheard in three cloud scenarios, namely, single-server virtualization, single mega datacenter, and multiple geodistributed datacenters.

By Fei Xu, Fangming Liu, *Member IEEE*, Hai Jin, *Senior Member IEEE*, and Athanasios V. Vasilakos, *Senior Member IEEE*

ABSTRACT | Infrastructure-as-a-Service (IaaS) cloud computing offers customers (tenants) a scalable and economical way to provision virtual machines (VMs) on demand while charging them only for the leased computing resources by time. However, due to the VM contention on shared computing resources in datacenters, this new computing paradigm inevitably brings noticeable performance overhead (i.e., unpredictable performance) of VMs to tenants, which has become one of the primary issues of the IaaS cloud. Consequently, increasing efforts have recently been devoted to guaranteeing VM performance for tenants. In this survey, we review the state-of-the-art research on managing the performance overhead of VMs, and summarize them under diverse scenarios of the IaaS cloud, ranging from the single-server virtualization, a single mega datacenter, to multiple geodistributed datacenters. Specifically, we unveil the causes of VM performance overhead by illustrating representative scenarios, discuss the performance modeling methods with a particular focus on their accuracy and cost, and compare the overhead mitigation techniques by identifying their effectiveness and implementation complexity. With the obtained insights into the pros and cons of each existing solution, we further bring forth future research challenges pertinent to the modeling methods and mitigation techniques of VM performance overhead in the IaaS cloud.

KEYWORDS | Cloud computing; predictable performance; virtualization; virtual machine (VM) performance overhead

I. INTRODUCTION

With the ability to scale computing resources on demand and provide a simple pay-as-you-go business model for customers, cloud computing is emerging as an economical computing paradigm, and has gained much popularity in the industry. Currently, a number of big companies such as Netflix and Foursquare [1] have successfully moved their business services from the dedicated computing infrastructure to Amazon Elastic Computing Cloud (EC2) [2], which is a leading public Infrastructure-as-a-Service (IaaS) cloud platform worldwide. Undoubtedly, more individuals (tenants) and enterprises will leverage the cloud to maintain or scale up their business while cutting down the

Manuscript received February 11, 2013; revised October 14, 2013; accepted October 17, 2013. Date of current version December 16, 2013. This work was supported in part by the National Natural Science Foundation of China under Grant 61370232 and by the National Basic Research Program (973 program) under Grant 2014CB347800.

F. Xu, **F. Liu**, and **H. Jin** are with the Services Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: fmliu@hust.edu.cn).
A. V. Vasilakos is with the Department of Computer Science, Kuwait University, Safat 13060, Kuwait.

Digital Object Identifier: 10.1109/JPROC.2013.2287711

Table 1 Measurement Results of Application Performance on Amazon EC2 in Existing Literature

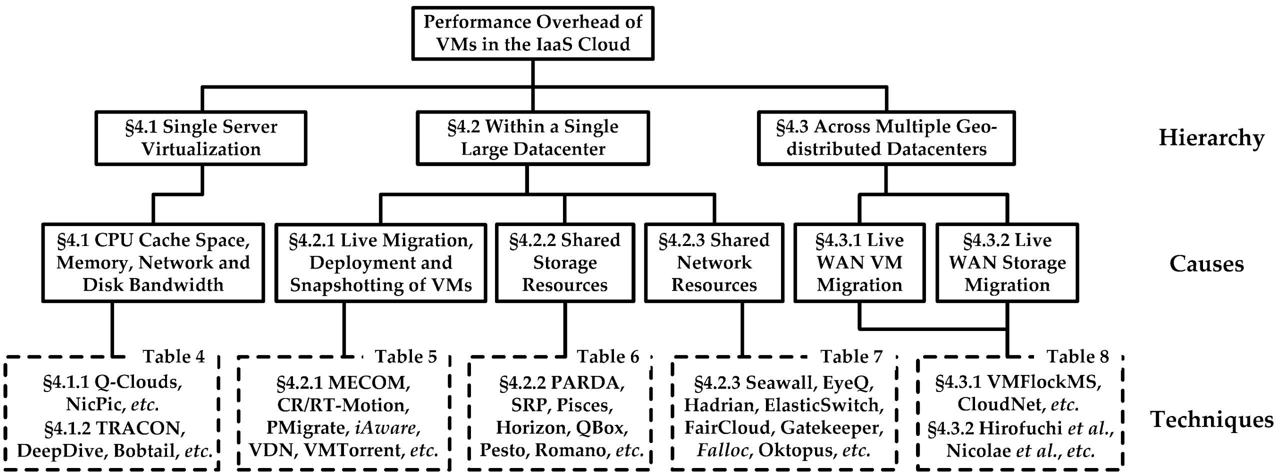
Type of applications	Latency-sensitive [6]		Network-sensitive [7]		Data-intensive [5]
	Doom 3 Game	Apple's Darwin Streaming Server	Network throughput of medium instances (VMs)	Round trip delay (RTT) of small instances	A MapReduce job on a 50-node cluster
Performance degradation and variation	Map load time increase: 25 – 110%	Throughput decrease: 30%	Throughput variation: 66%	RTT variation: ~4.478	Job execution time increase: 100 – 150% variation: 11%

budget, as reported by the International Data Corporation (IDC) that the business revenue brought by cloud computing will reach $1.1 trillion by 2015 [3].

Unfortunately, the running performance of virtual machines (VMs) on the IaaS cloud platform is unpredictable, which significantly impacts the tenant's service-level agreement (SLA) offered by the cloud provider [4]. Specifically, due to the resource contention of VMs, the VM performance obtained in the IaaS cloud is severely degraded and highly variable, compared to the performance of VMs running in an isolated environment (e.g., running alone on a physical server in a local cluster) [5]. In this paper, we formally refer to such VM performance degradation and variation as VM performance overhead. Not surprisingly, several latest measurement studies on Amazon EC2 [2] have recently shown that diverse cloud applications listed in Table 1, ranging from latency sensitive [6], network sensitive [7], to data intensive [5], suffer significant performance overhead, which consequently hinders users from moving their performance-critical business to the cloud. In particular, the performance degradation and variation metrics listed in Table 1 are measured as: 1) the ratio of the degraded performance obtained in the IaaS cloud, in terms of the increased job execution time and the decreased network throughput, to the application performance obtained in isolated VMs; and 2) the variation of application performance obtained in the IaaS cloud over a period of time. Accordingly, these evidences bring forth a series of urgent demands for managing the performance overhead of VMs and guaranteeing VM performance for tenants in the cloud.

From a high level perspective of the IaaS cloud, existing studies control the performance overhead of VMs under three representative scenarios, ranging from the single-server virtualization, a single large datacenter, to multiple geodistributed datacenters, as summarized in Fig. 1. On the one hand, even with built-in resource isolation mechanisms on central processing unit (CPU) cores, memory and disk capacities across VMs [8], the contention on shared cache and input/output (I/O) bandwidth resources among colocated VMs is difficult to be alleviated by existing hypervisors [9] in practice [10]. Recent efforts on single-server virtualization are primarily devoted to alleviating the VM performance overhead caused by the contention on shared server resources (e.g., CPU cache and I/O bandwidth) by resource isolation. Moreover, to gain deep understanding of the relationship between the performance overhead and resource consumption of VMs, there have been a number of recent studies devoted to the performance modeling. With an accurate modeling method, the VM performance overhead can be alleviated in a cost-effective manner, as compared to the resource isolation techniques.

On the other hand, the performance overhead of VMs within a single mega datacenter and across multiple



Performance Overhead of VMs in the IaaS Cloud
§4.1 Single Server Virtualization
§4.2 Within a Single Large Datacenter
§4.3 Across Multiple Geo-distributed Datacenters
Hierarchy
§4.1 CPU Cache Space, Memory, Network and Disk Bandwidth
§4.2.1 Live Migration, Deployment and Snapshotting of VMs
§4.2.2 Shared Storage Resources
§4.2.3 Shared Network Resources
§4.3.1 Live WAN VM Migration
§4.3.2 Live WAN Storage Migration
Causes
Table 4
§4.1.1 Q-Clouds, NicPic, etc.
§4.1.2 TRACON, DeepDive, Bobtail, etc.
Table 5
§4.2.1 MECOM, CR/RT-Motion, PMigrate, iAware, VDN, VMTorrent, etc.
Table 6
§4.2.2 PARDA, SRP, Pisces, Horizon, QBox, Pesto, Romano, etc.
Table 7
§4.2.3 Seawall, EyeQ, Hadrian, ElasticSwitch, FairCloud, Gatekeeper, Falloc, Oktopus, etc.
Table 8
§4.3.1 VMFlockMS, CloudNet, etc.
§4.3.2 Hirofuchi et al., Nicolae et al., etc.
Techniques

Fig. 1. *Classification of causes and mitigation techniques of VM performance overhead from the viewpoint of IaaS cloud hierarchy.*

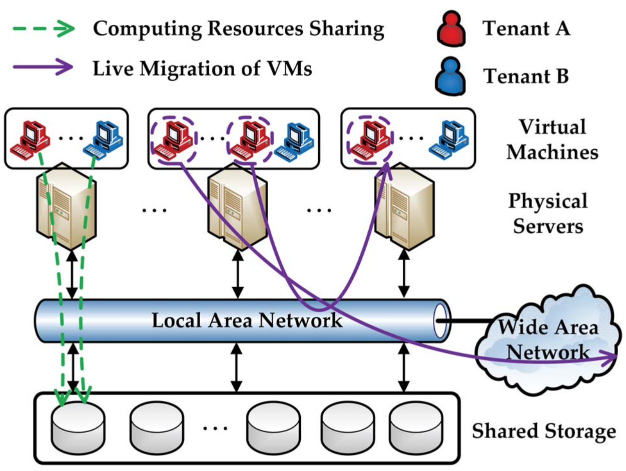


Fig. 2. ***Typical architecture of the IaaS cloud datacenter. The running VMs of tenant A and B contend for the shared computing resources in datacenters, such as network and storage resources. In addition, live migration of tenant A's VMs severely impacts the performance of the applications running on tenant B's VMs.***

geodistributed datacenters, however, can hardly be managed by the traditional resource isolation techniques. Specifically, in a single large datacenter, the distributed network and storage resources are shared among a huge amount of VMs leased by multiple tenants, as illustrated in Fig. 2. Such a cloud scenario undoubtedly makes the VM performance issue more intractable than the single-server virtualization. Furthermore, among multiple geodistributed datacenters, the issue of VM performance overhead becomes even more complicated due to the highly variable and scarce network resource over the wide area network (WAN). While a number of solutions that mitigate such a VM performance issue have emerged recently, the fast innovations of these overhead mitigation techniques mandate comprehensive surveying of the state-of-the-art literature under the three cloud scenarios discussed above, with the aim of highlighting future directions in managing the performance overhead of VMs in IaaS clouds.

Therefore, we compare the latest solutions of managing the performance overhead of VMs under the three cloud scenarios shown in Fig. 1. The focus of this paper is on obtaining the insights into the benefits and costs of these solutions, as well as raising the future research challenges in the performance modeling methods and overhead mitigation techniques.

There also exists several alternative approaches dealing with the VM performance issue. For example, CloudCmp [11] has been developed for guiding tenants in selecting the IaaS cloud platform with the best application performance from several cloud providers, such as Amazon EC2 [2], Microsoft Azure [12], and Rackspace Cloud Servers [13]. Several studies [14], [15] explore the hardware heterogeneity within the same type of Amazon EC2 instances (i.e., VMs). Potential improvements of VM performance can accordingly be obtained by placing workloads onto the instances with good performance in the EC2 datacenters. The strategies above can provide acceptable VM performance and reduce budgets on behalf of tenants, while they might hurt the benefits of cloud providers. Accordingly, we limit the scope of this survey to the solutions that alleviate VM performance overhead on behalf of both tenants and cloud providers.

The rest of this survey is organized as follows. Section II explores the causes and summarizes the metrics of VM performance overhead by illustrating the representative scenarios of the IaaS cloud. The existing measurement and modeling methods of VM performance overhead as well as the future research issues are discussed in Section III. Section IV summarizes and compares the state-of-the-art solutions that manage VM performance overhead under diverse scenarios of the IaaS cloud, along with a fruitful discussion of future research challenges. Finally, we conclude this survey in Section V.

II. CAUSES AND METRICS OF VM PERFORMANCE OVERHEAD IN IaaS CLOUD

There are various kinds of causes that can lead to the performance overhead of VMs in IaaS cloud datacenters, as illustrated in Fig. 1. In this section, we first expose and discuss these causes of VM performance overhead in detail, especially on single-server virtualization, within a single datacenter, and across multiple geodistributed datacenters, respectively. We next summarize the metrics of VM performance overhead used in the literature.

A. Performance Impact of Single-Server Virtualization

The server virtualization technique [9], which serves as the cornerstone of cloud computing, multiplexes the computing resources of a physical server for multiple colocated VMs. Current virtual machine monitors (VMMs) or hypervisors [9] (e.g., Citrix *XenServer*, VMware vSphere, and Microsoft *Hyper-V*) have already provided good performance isolation mechanisms on sharing the resource of CPU cores, memory, and disk capacities among colocated VMs. The isolation mechanisms can be implemented by fully fledged CPU schedulers (e.g., Credit Scheduler in *Xen* [8]) and static partition of memory and disk capacities. However, server resources like CPU cache space, memory bandwidth, network, and disk I/O bandwidth are very hard to be isolated in existing hypervisors [10]. This inevitably leads to the contention on these aforementioned server resources among colocated VMs. As shown in [16], with each VM allocated to a dedicated CPU core, statically partitioned memory capacity and disk space, the execution time of the *lbm* application from the SPECCPU 2006 benchmark [17] deployed on two colocated VMs on an

Intel Core-2-Duo processor can be prolonged by up to 120%, compared with the performance of the *lbm* application VM running alone. The above significant performance overhead of VMs is caused by the contention of cache and memory bandwidth on a physical server.

Furthermore, with the tide of green computing and the recent prevalence of deploying multicore architecture systems in datacenters, VM consolidation is now widely used to improve the resource utilization of a multicore physical server [18] and reduce the energy consumption of datacenters [19]. As more VMs are colocated on a physical server, VM consolidation, however, increases the severity of performance overhead of these colocated VMs on a physical server. As a result, the contention of shared cache and I/O bandwidth resources among colocated VMs on a physical server becomes one of the causes of VM performance overhead in the IaaS cloud.

B. Cause of VM Performance Overhead Within a Single Datacenter

It is essential for tenants to lease a cluster of VMs from cloud providers when processing large jobs with a large amount of data (e.g., terabyte), such as MapReduce [20] and scientific workloads. In such cases, one natural question is as follows: Are the performance isolation techniques for colocated VMs on a single physical server enough to eliminate the performance overhead among multiple VMs in a single datacenter? Unfortunately, the routine operations of the IaaS cloud datacenter, such as (concurrent) live migration, deployment and snapshotting of multiple VMs, and the contention of multiple VMs on the shared cloud network and storage resources, complicate the issue of VM performance overhead and significantly degrade the performance of VMs. We elaborate them in the following three cases, respectively.

1) *The (concurrent) live migration of VMs* [21] is an indispensable tool that enables the load balancing and power saving of a datacenter. It can nevertheless cause the performance overhead to the migrating VMs and colocated VMs on the migration source and destination servers, which is illustrated in Fig. 2. As shown in [22], live migration of a VM configured with 256-MB memory hosting the MySQL server, over a 100-Mb/s Ethernet link, severely prolongs the end-to-end mean response time of the multitier Web applications (e.g., RUBiS [23]) by 211%, compared with the application performance before the live VM migration. Undoubtedly, migrating multiple VMs simultaneously in a cloud datacenter will overload the datacenter network even worse, which further constrains the performance of CPU and network resources on the migration and destination servers, and incurs a long migration time (i.e., from several minutes to hours) [24].
2) *The concurrent deployment and snapshotting of multiple VMs* are two commonly occurring patterns of the routine operations in an IaaS cloud datacenter [25]. For example, to execute jobs with a large amount of data, it is the basic requirement of tenants to fast deploy (initialize) a large number of VMs simultaneously in a public IaaS cloud platform [26]. As for the private cloud owned by a big IT enterprise, it is also the common demand of the employees for fast provisioning a large number of VMs concurrently in the daily working hours, even when there are a burst of VM deployment requests. Once the VMs are initialized and running, there is another basic requirement of tenants to fast take snapshots of their leased cluster of VMs simultaneously. Fast snapshotting is also desired by Amazon Elastic Block Store (EBS) [27] for the checkpoint of VM states or the VM data sharing among workmates. Currently, both the public and private clouds, however, suffer the VM performance overhead caused by the concurrent deployment and snapshotting of multiple VMs. This is because these two operation patterns can generate a significant amount of network traffic, which inevitably interferes with the other VMs in the same datacenter [25].
3) *The inability to isolate shared cloud network and storage resources among multiple VMs* is another cause of performance overhead of VMs within a datacenter. In general, to serve the disk and network I/O requests for tenant VMs, the IaaS cloud datacenter deploys the shared back–end storage such as storage area network (SAN), and shares the network resources such as network bandwidth on physical servers and switches, as shown in Fig. 2. This accordingly leads to the I/O contention on the shared storage and network resources among multiple VMs of tenants in a cloud datacenter. It is revealed by the latest measurements on Amazon EC2 that the standard medium instances experience up to 66% variation of the network I/O throughput [28], and the write I/O bandwidth of standard small instances can vary by as much as 50% from the mean [6].

C. Cause From Live WAN Migration Across Multiple Geodistributed Datacenters

It is reported by Gartner Inc. that the hybrid use of public and private clouds becomes an imperative trend of cloud computing [29], and more than half of large enterprises around the world will have such hybrid cloud deployments by the end of 2017 [30]. A commercial solution, Amazon Virtual Private Cluster (VPC) [31], now allows the private datacenter of IT companies to connect to a cluster of Amazon EC2 instances. These evidences hence bring new challenges in managing VM performance overhead across multiple geodistributed datacenters, such as the performance overhead of VMs caused by live migration of VMs and storage over the WAN connection.

Specifically, live migration of VMs and storage across geodistributed datacenters over a WAN connection serves as an indispensable technique for managing computing resources of these datacenters. For big IT enterprises with multiple geodistributed datacenters, for example, live VM migration over the WAN enables the load balancing and power saving across datacenters, by migrating several VMs from heavy-loaded datacenters to light-loaded datacenters and consolidating the VMs from the small datacenter into the large datacenter, respectively. For small IT companies deploying their business on the hybrid cloud, live WAN migration of VMs allows them to move or replicate their business services to the public cloud from the private cloud when the loads of service requests are bursting [32].

Different from the live VM migration in a local area network (LAN) discussed in Section II-B, the live WAN migration of VMs includes the data transfer of not only the VM memory state, but also the VM disk image and the ongoing network connections [33]. Accordingly, this would cause a long VM migration time, application downtime, and a large amount of network traffic, which further leads to severe performance overhead to a certain number of VMs in the migration source and destination datacenters over the WAN, as well as the migrating VMs. Such performance overhead cannot be easily managed by VM performance guarantee solutions on a single server or in a single datacenter. For example, over the WAN with a maximum throughput of 85 Mb/s, live migration of a VM configured with 1.7-GB memory and 10-GB disk hosting a Web service application, performed between the datacenters in Illinois and Texas, can undergo up to 40 min of disk transfer and 210 s of memory transfer. The response time of the Web service during the migration process is highly variable, and can be degraded to 52 ms on average, as compared to 10 ms before the migration [32]. Consequently, the migration of VMs over the WAN becomes another cause of performance overhead of VMs in IaaS clouds.

D. Evaluation and Metrics of VM Performance Overhead

After discussing the causes of VM performance overhead, the next natural and fundamental question is as follows: How can we evaluate the performance overhead of VMs, and what are the metrics? To simply *capture* the performance overhead of VMs, the first type of metric is the performance degradation of various applications running in the IaaS cloud, compared with the application performance obtained in the VMs running in an isolated environment. Specifically, the performance degradation of VMs P_d can be measured as

$$P_d = \frac{|x_{\text{iaas}} - x_{\text{isolation}}|}{x_{\text{isolation}}} \tag{1}$$

where x_{iaas} and $x_{\text{isolation}}$ are the performance of VM applications running in the IaaS cloud and in isolation, respectively. The larger magnitude of performance degradation value P_d indicates the more severe performance overhead of VMs.

The second type of metric that *captures* the performance overhead of VMs is the variation of VM performance obtained in the IaaS cloud over a period of time. Specifically, the VM performance variation P_v can be represented by the coefficient of variation [34], which is formulated as

$$P_v = \frac{1}{\bar{x}}\sqrt{\frac{1}{n_x - 1}\sum_{i=1}^{n_x}(x_i - \bar{x})^2} \tag{2}$$

where x_i is the VM performance measured in the IaaS cloud over time. n_x is the number of performance measurements. $\bar{x}$ is the average of VM performance over time. The larger magnitude of performance variation value P_v implies the more severe performance overhead of VMs.

Due to the security concern and noticeable measurement costs, the performance degradation of VM applications can hardly be obtained in several scenarios. In such cases, the third type of metric is required to *reflect* the performance overhead of VMs by examining the performance of routine operations in IaaS cloud datacenters, as introduced in Section II-B and C. Specifically, such a type of metric includes the duration of live VM migration (i.e., migration time), downtime of the migrating VM, energy consumption of live VM migration, and the amount of network traffic caused by the live VM migration, deployment, and snapshotting. The smaller migration time, migration downtime, amount of network traffic, and energy consumption implies the better performance of routine operations and lighter performance overhead of VMs. In summary, existing studies in the literature leverage the above three types of metrics to evaluate the performance overhead of VMs in the IaaS cloud.

III. MODELS AND MEASUREMENTS ON VM PERFORMANCE OVERHEAD

In order to well manage the performance overhead of VMs in datacenters, the primary requirement is to accurately model and measure VM performance overhead. Existing studies have devoted great efforts to modeling the performance overhead of VMs on two aspects: performance interference of colocated VMs on a physical server and VM performance overhead caused by live migration of VMs. We review the representative performance modeling methods on the two aspects, and summarize them in Tables 2 and 3, respectively. In particular, the column "kind/type of model" indicates whether the model is linear

Table 2 Comparison of Modeling Performance Interference of Colocated VMs on a Physical Server

Approaches	Pre-run of workload	Scale of VMs	Kind/Type of model	Resource consumption profile	Prediction error
Zhu *et al.* [35]	√	Two	Linear	CPU, memory, network, disk	Less than 8%
Koh *et al.* [36]	√	Two	Linear	CPU, memory, network, disk, cache, number of VM switches, etc.	~ 5%
TRACON [37]	√	Two	Non-linear	CPU, network	19%
Bu *et al.* [38]	√	Multiple	Non-linear	CPU, network, disk	12%
Cuanta [16]	√	Multiple	N/A	Cache, memory	Less than 4%
pSciMapper [19]	×	Two	Non-linear	CPU, memory, network, disk	N/A
Q-Clouds [39]	×	Multiple	Linear/non-linear	CPU	~ 13%

or nonlinear to its parameters. The column "resource consumption profile" in Table 2 represents the parameters of VM performance interference models. The column "prediction error" shows the accuracy of the predicted VM performance overhead in comparison to the measured overhead data. Specifically, the smaller value of the prediction error implies the more accurate prediction modeling approach.

A. Models of VM Performance Overhead Caused by Single-Server Virtualization

To model the performance interference of colocated VMs on a single server, the representative approach is to develop an accurate function of resource consumption profile of these VMs. For example, Zhu and Tung [35] propose a simple dilation factor to represent the performance interference between two colocated VMs on each of the 4-D resource consumptions of VMs (i.e., CPU, memory, network, and disk I/O). By prerunning each workload which is colocated with a background benchmark workload, this method manually varies the resource consumption of workloads (e.g., varying CPU consumption from 10% to 100% with a step of 10%), and acquires the dilation factors *offline* for the workloads. The larger dilation factor means the more severe performance interference of colocated VMs. Similar with the previous approach, Koh *et al.* [36] predict the performance interference between two VMs by two linear statistical approaches (i.e., weighted mean method and linear regression analysis). In addition, this method enriches the resource consumption profile of VMs with other system-level workload metrics, such as the statistics of cache hits and misses, VM switches. The above linear statistical model of performance interference I is presented as

$$I = \alpha_0 + \sum_{i=1}^{p} \alpha_i X_i \tag{3}$$

where X_i is the ith dimension of VM resource consumption. p is the dimensions of VM resource consumption, and α_0 and α_i are the model coefficients.

With a particular focus on the colocated VMs hosting data-intensive applications, two nonlinear statistical models have been developed in [37] and [38]. Specifically, by extending the linear model shown in (3), TRACON [37] models the performance interference I between two colocated VMs as a *quadratic* function of CPU and I/O bandwidth consumption X_i of VMs, which is given by

$$I = \alpha_0 + \left(\sum_{i=1}^{4} \alpha_i X_i \right)^2. \tag{4}$$

A more accurate model has been presented in [38], where the performance interference of multiple colocated VMs is

Table 3 Comparison of Modeling VM Performance Overhead Caused by Live Migration of VMs

Approaches	Pre-run of workload	Online prediction	Kind/Type of model	Objective of model	Prediction error
Wu *et al.* [40]	√	√	Non-linear	Migration time	Less than 9.2%
CloudScale [41]	√	√	Linear	Migration time	Less than 14%
Akoush *et al.* [42]	√	√	Non-linear	Migration time, downtime	Less than 9.3%
Liu *et al.* [43]	√	√	Linear/non-linear	Migration time, downtime, network traffic, energy	Less than 10%
Jung *et al.* [22]	√	×	N/A	SLO violations of co-located VMs	N/A
Lim *et al.* [44]	√	√	Non-linear	Expanded execution time of co-located VMs	Less than 15.8%
Breitgand *et al.* [45]	√	√	Non-linear	SLO violations of the migrating VM	N/A

modeled as an *exponential* function of VM CPU and I/O resource consumption, which is shown as

$$I = \alpha_0 + \alpha_1 \exp\left(\sum_i \gamma_i X_i^{\text{cpu}}\right) + \alpha_2 \exp\left(\sum_i \omega_i X_i^{\text{io}}\right) \tag{5}$$

where X_i^{cpu} and X_i^{io} are the CPU and I/O resource consumptions of VMs, respectively. γ_i and ω_i are the model coefficients. Nevertheless, these linear and nonlinear statistical models in (3)–(5) require the *offline training* of experimental statistics to obtain the model coefficients, e.g., α_i and γ_i. To particularly predict the performance interference on the cache and memory bandwidth among multiple colocated VMs, Cuanta [16] first leverages a certain synthetic cache benchmark and an interference matrix to create the cache clones with the same cache behaviors of these VMs. This method then looks up a degradation table to model the effect of interference from colocated VMs. In particular, the interference matrix and degradation table are generated *offline* by the preruns of the cache benchmark and VMs.

Whereas the previous approaches require the prerun of workloads to acquire necessary parameters of the statistical prediction model, several studies make efforts to break such a limitation. For example, pSciMapper [19] leverages the Pearson Correlation Coefficient [34] of multidimensional VM resource consumption (e.g., CPU, memory, network, and disk I/O) to *estimate* the performance interference I_{XY} between two VMs. Specifically, the interference model is given by

$$I_{XY} = \frac{\|p\sum X_i Y_i - \sum X_i \sum Y_i\|}{\sqrt{p\sum X_i^2 - (\sum X_i)^2}\,\sqrt{p\sum Y_i^2 - (\sum Y_i)^2}} \tag{6}$$

where X_i and Y_i are the resource consumption of the two colocated VMs, and p is the dimensions of VM resource consumption. In particular, the larger correlation result I_{XY} indicates the more severe VM performance interference. Q-Clouds [39] develops a discrete-time (linear or nonlinear) model of history performance feedbacks and current CPU resource caps of VMs u_i to online predict the current performance of colocated VMs P_i. At each time step i, the discrete-time model is given by

$$P_i = \boldsymbol{\Phi}(P_{i-1}, \ldots, P_{i-n}, u_i, \ldots, u_{i-m}) \tag{7}$$

where m and n represent to what extent the history values of performance data and CPU resource caps impact the current performance, respectively. In particular, the model construction of $\boldsymbol{\Phi}(\cdot)$ requires a learning phase using the historical observed performance data P_i and CPU resource caps u_i of multiple VMs.

B. Models of VM Performance Overhead Caused by Live Migration of VMs

As discussed in Section II-D, the migration time is an important metric that reflects VM performance overhead caused by live migration of VMs. A number of methods have recently been proposed to model the migration time. For example, Wu and Zhao [40] particularly study the relationship between the *domain*-0 CPU allocation U_{cpu} and live migration time T_{mig}. Specifically, given a VM to be migrated, T_{mig} can be modeled as a nonlinear function of U_{cpu}, which is given by

$$T_{\text{mig}} = \frac{\beta}{U_{\text{cpu}}^{\kappa}} \tag{8}$$

where β and κ are the model coefficients, and $\beta, \kappa \in (0, \infty)$. In particular, these coefficients require the prerun of workload applications and are trained offline using the experimental results of multiple preruns. Given a VM migration environment, CloudScale [41] estimates the migration time T_{mig} using a linear function of the memory size V_{mem} of the migrating VM. Similar to (8), such a linear function is also derived based on the regression analysis of experimental results of multiple migration preruns. The impact of another two parameters (i.e., network link speed R and memory page dirty rate D) to the VM migration time T_{mig} has been examined by simulation models in [42], where T_{mig} is proved to be nonlinear to R and D.

Based on the four parameters (i.e., U_{cpu}, V_{mem}, R, and D), an integrated model of the VM migration time T_{mig} has been developed in [43], which is given by

$$T_{\text{mig}} = \sum_{i=1}^{n} t_i, \quad t_i = \begin{cases} \dfrac{t_{i-1}D}{r}, & i \geq 2 \\ \dfrac{V_{\text{mem}}}{r}, & i = 1 \end{cases} \tag{9}$$

where n is the total rounds of memory precopy phases.[1] t_i is the phase time for each round of memory precopy, and r is the network transmission rate for the live VM migration. As r highly correlates with U_{cpu} [40] and is restricted by R, it can be further modeled as $r = f(U_{\text{cpu}}, R)$, where $f(\cdot)$ is a nonlinear function. In addition to the performance metric of migration time, Liu *et al.* [43] further develop a series of statistical models to predict several other important VM migration metrics, including the network traffic N_{mig}, energy consumption E_{mig}, and application downtime

[1]We mainly discuss the *precopy migration* [21] in this survey.

T_{down}, respectively. Accordingly, the migration performance P_{mig} of migrating a VM can be comprehensively evaluated as

$$P_{\text{mig}} = aT_{\text{mig}} + bT_{\text{down}} + cN_{\text{mig}} + dE_{\text{mig}} \qquad (10)$$

where a, b, c, and d are model coefficients. In particular, the application downtime T_{down} is the phase time for the last round of memory copy t_r, according to (9). The network traffic N_{mig} are the total transferred memory pages during live VM migration, which are calculated as $V_{\text{mem}} + \sum_{i=1}^{n} t_i \cdot D$. The energy consumption E_{mig} is proportional to the network traffic N_{mig}, which can be modeled as $\rho N_{\text{mig}} + \theta$, where ρ and θ are model coefficients. In particular, these model coefficients are obtained based on the regression analysis of experimental statistics which are measured *offline*.

The former approaches model the performance overhead of VMs caused by live migration from the perspective of migration performance itself. From the perspective of the migrating VM and other colocated VMs on the migration source and destination servers, VM performance overhead can simply be measured *offline* as the *performance degradation* caused by the live migration for each specific application using (1) [22], [41]. In addition, such a VM performance overhead can be modeled mathematically. Specifically, considering live VM migration as a job that consumes the computing resources of physical server, Lim *et al.* [44] predict the performance overhead (i.e., the prolonged execution time T_{ext}) of colocated VMs on the migration source and destination servers, using a two-resource contention model, which is given by

$$T_{\text{ext}} = \tau(q_1 q_2 + (1 - q_1)(1 - q_2)) \qquad (11)$$

where τ is the application execution time running alone on a physical server. q_1 and q_2 are the probabilities of accessing a kind of VM resource by live VM migration and the colocated VM, respectively, and $q_1, q_2 \in [0, 1]$. The parameters in (11) are acquired *offline* using the VM performance statistics. Especially for the migrating VM hosting a Web server, Breitgand *et al.* [45] adopt the queuing theory to model the service-level objective (SLO) violation of such a migrating Web application, which is quantified as a function of the network bandwidth allocated to the live VM migration B_m. Specifically, given the SLO of the Web service t_{SLO} and the maximum network bandwidth of the Web server B, the probability of SLO violations $P(t > t_{\text{SLO}})$ is modeled by

$$P(t > t_{\text{SLO}}) = \exp(t_{\text{SLO}}(\lambda - \mu(B - B_m))) \qquad (12)$$

where $\mu(B - B_m)$ denotes the requests' response rate of the Web server, given the migration bandwidth B_m. λ is the request's arrive rate on the Web server.

C. Summaries, Insights, and Open Research Issues

From the above comparison of performance modeling methods summarized in Tables 2 and 3, we raise the following conclusions, insights, and open research issues.

First, statistical methods, such as regression analysis and correlation analysis [34], have been proved to be effective in the model construction and estimation of VM performance overhead caused by the single-server virtualization and live VM migration. Although the statistical methods are widely used in the existing modeling approaches with a high prediction accuracy, most of them require the prerun of VM workloads to acquire necessary parameters of the prediction models of VM performance overhead. Such a requirement is not always practical in datacenters, as it is under the assumption that few kinds of workloads are running on the homogeneous physical servers. In real-world cloud datacenters, the heterogeneity of hardwares [14] and workloads [46] becomes common. The prerun of workloads or offline measurements on each type of hardware will undoubtedly bring noticeable processing overhead to the model construction. Accordingly, how to develop a lightweight, yet effective, prediction model of VM performance overhead without the prerun of workloads becomes an open research problem.

Second, from the above discussion on the complexity of model construction, we further obtain a key insight into the relationship between the accuracy and cost (e.g., the prerun of VM workloads) of the model of VM performance overhead. For example, although the correlation analysis of VM resource consumption in (6) does not require the prerun of VM workloads, this method only provides an *estimation* of performance interference between two VMs, rather than an *accurate* value of the VM performance interference. As a result, the accuracy of the prediction model should be determined by its practical usage. On the one hand, a lightweight estimation model that catches the qualitative trend of VM performance overhead is sufficient to guide in determining an appropriate assignment of VMs, for mitigating the performance overhead of VMs (e.g., [19]). On the other hand, a sophisticated prediction model that accurately quantifies the performance overhead of VMs is required to reallocate the computing resources to the competing VMs within a single datacenter (e.g., [39]).

Third, although a number of performance models of single-server virtualization have been proposed (refer to Table 2), there is a lack of a holistic prediction model to quantify the performance overhead of multiple (i.e., more than three) colocated VMs on multidimensional shared server resources, i.e., CPU cache, memory bandwidth, network, and disk I/O bandwidth. Hence, how to design such a holistic model of performance overhead among multiple VMs becomes a promising research problem. One potential modeling solution is to extend the discrete-time model of CPU resource in [39], i.e., (7), to support multidimensional server resources. However, this method requires learning the relationship between the current performance data and the historical resource consumption of VMs for the model construction. The overhead (e.g., CPU cycles) caused by such a learning phase cannot be neglected. Accordingly, it would also be interesting to mitigate such an overhead incurred by the learning phase for constructing this model.

Fourth, to quantify the performance overhead of VMs caused by live VM migration, existing modeling methods focus on the performance of either the migrating VM or colocated VMs on migration source and destination servers (refer to Table 3). Accordingly, it would be desirable to design a prediction model that jointly quantifies the performance overhead of the migrating VM and its colocated VMs in order to comprehensively evaluate the performance overhead caused by the live VM migration process [47]. Furthermore, as the multitier application that requires multiple interdependent VMs are widely deployed in datacenters [22], the migration of a VM (e.g., the SQL server) can adversely impact the performance of other associated VMs (e.g., the Web server). Hence, the evaluation of performance overhead of live VM migration cannot be restricted to the single migrating VM (i.e., [45]). The cascading performance effects to other associated VMs should be incorporated into the model of VM performance overhead, particularly for evaluating multitier VM applications.

Last but not least, although the performance of live VM migration has been comprehensively studied and modeled (e.g., [43]) in the LAN environment, the performance model of live VM migration over the WAN has not been investigated yet. Such a performance model would be beneficial for the tenant to utilize the computing resources in the hybrid cloud (i.e., public and private clouds across multiple geodistributed datacenters). For example, the migration performance model in the WAN is effective in improving the performance of the migrating VM and the migration itself, by selecting an appropriate migrating VM and a migration destination datacenter. However, the performance model of live VM migration in the LAN cannot be directly applied in the WAN environment due to at least two reasons: 1) the network link bandwidth can vary significantly over time in the WAN; and 2) the VM disk image also requires to be transferred over the WAN. Consequently, how to develop a performance model of live WAN migration of VMs becomes another future research problem.

IV. MANAGING PERFORMANCE OVERHEAD OF VMs IN IaaS CLOUD

A large number of approaches have recently been proposed to manage VM performance overhead and guarantee the performance of VMs in the IaaS cloud. We broadly classify

Table 4 Comparison of Approaches to Alleviate VM Performance Overhead Caused by Single-Server Virtualization

Approaches	Resources	Optimization	Complexity	Techniques	Effectiveness
Q-Clouds [39]	CPU cache	Performance	Medium	CPU core compensation	CPU utilization improvement: 35%
Zhang *et al.* [50]		Performance	Medium	Cache partitioning	Performance improvement: 15%
Boutcher *et al.* [51]	Disk I/O	Performance, fairness	Low	VMM- and VM-level I/O schedulers	I/O throughput improvement: 72%
Blagojevic *et al.* [52]		Performance	Low	Priority-based I/O scheduler	Response time improvement: 200%
Hardware [53]	Network I/O	Performance	High	Multiple or multi-queue network adapters	Guarantee I/O throughput
Xen manual [8] Barker *et al.* [6]			Low	Bandwidth capping	Mitigate I/O interference
NicPic [54]		Performance, scalability	Low	Network packets scheduling	Mitigate I/O interference, reduce CPU overhead
Zhu *et al.* [35] TRACON [37]	CPU, memory, disk, network	Performance	Medium, $\mathcal{O}(m{\cdot}n)$	Interference calculation, VM assignment	Application completion time reduction: 50%
Cuanta [16]	CPU cache	Performance, energy	Medium, $\mathcal{O}(n{\cdot}m\log\frac{n}{m})$	Cache pressure clone, VM assignment	Balance tradeoff between performance and energy
pSciMapper [19]	CPU, memory, disk, network	Performance, energy	Medium, $\mathcal{O}(n^2)$	Metrics correlation, VM assignment	Power reduction: 56%
Ahn *et al.* [55]	CPU cache	Performance	Medium, $\mathcal{O}(m{\cdot}n)$	VM assignment	Performance improvement: 17%
DeepDive [10]	CPU, memory, disk, network	Performance	Low, $\mathcal{O}(n)$	Metrics clustering, VM assignment	Identify and mitigate VM interference with low overhead
Bobtail [56]	Network I/O	Performance	Low, $\mathcal{O}(m{\cdot}n)$	VM examining and assignment	Network I/O latency reduction: 50 – 65%

Table 5 Comparison of Approaches to Manage VM Performance Overhead Caused by Live Migration, Deployment, and Snapshotting of Multiple VMs Within a Single Datacenter

Approaches	Objective	Optimization	Complexity	Techniques	Effectiveness
MECOM [57]	Live migration of a VM	Migration time, application downtime, network traffic	Medium	Compression	Migration time reduction: 32%
Koto *et al.* [58] Chiang *et al.* [59]			Low	Filter un-useful memory pages	Migration time reduction: 34 – 68.3%
Svard *et al.* [60]			Medium	Compression, page delta transfer	Migration time reduction: 40%
CR/RT -Motion [61]			Medium	Transfer and replay of execution logs	Migration time reduction: 31.5%
Jo *et al.* [62]			Low	Transfer storage locations of memory	Migration time reduction: over 30%
PMigrate [63]		Migration time, application downtime	Low	Parallelizing migration	Migration speedup factor: 2.49 – 9.88
Liu *et al.* [43]		Migration performance, energy	Medium, $\mathcal{O}(n)$	Migrate the VM with minimum cost	Migration cost reduction: 72.9%
Breitgand *et al.* [45]		Performance of migration and the migrating VM	Low	Migration bandwidth allocation	minimize SLO violations
CloudScale [41]	Resource conflict handling	Performance of co-located VMs	Low, $\mathcal{O}(m{\cdot}n)$	Migrate the VM with minimum cost on VM performance	SLO violation reduction: 83%
iAware [47]	Load balancing, power saving	Performance of VMs on source and destination	Low, $\mathcal{O}(m{\cdot}n)$		Performance improved: 16 – 65%
Deshpande *et al.* [24]	Concurrent migration of multiple VMs	Concurrent migration time, network traffic	Medium	De-duplication	Network traffic reduction: 75%
VDN [64]	Concurrent deployment and snapshotting of multiple VMs	Startup delay of multiple VMs	High	De-duplication, chunking	Speedup factor: 30 – 80
Razavi *et al.* [65]		Scalability, VM booting delay	Low	VM image caching	Speedup factor: 8
Nicolae *et al.* [25]		Performance of VM snapshotting and deployment	Medium	Lazy VM deployment, store incremental update	Speedup factor: 2 – 25, bandwidth reduction: 90%
VMTorrent [26]		Scalability, VM booting delay	Medium	VM image profile, block pre-fetching	Speedup factor: 30

and summarize these solutions according to the high-level hierarchy of IaaS cloud datacenters shown in Fig. 1. We compare these solutions on single-server virtualization, within a single mega datacenter, and across multiple geo-distributed datacenters, in Section IV-A–C, respectively. These overhead mitigation techniques are summarized in Tables 4–8. In particular, the column "complexity" represents the implementation complexity of techniques introduced to the original systems (e.g., hypervisors, shared storage, and networks). It also shows the time complexity of VM assignment algorithms in a number of overhead mitigation techniques (e.g., pSciMapper [19], *iAware* [47], Pesto [48], and Oktopus [49]). The time complexity notations m and n denote the number of physical servers and virtual machines in a datacenter, respectively.

A. Alleviating VM Performance Overhead Caused by Single-Server Virtualization

Currently, hypervisors [9], such as Citrix *XenServer* and Microsoft *Hyper-V*, provide a baseline of performance isolation on CPU, memory, and disk resources by simply allocating each VM (including *domain*-0) with dedicated CPU core(s) and an amount of nonoverlapped memory and disk capacities. However, the cache and I/O bandwidth resources shared among multiple colocated VMs shown in Fig. 3 cannot be easily isolated in existing hypervisors, as we discussed in Section II-A. To deal with such a performance issue, a number of approaches have been proposed to mitigate the performance overhead of VMs caused by the single-server virtualization. We summarize these solutions in Table 4.

1) Resource Isolation Among Colocated VMs: To alleviate performance overhead of VMs on a single physical server, one conventional approach is to isolate the computing resources among colocated VMs. To mitigate the performance interference on the CPU cache and memory bandwidth, Q-Clouds [39] dynamically provisions the underutilized or idle CPU resource to the impacted (victim) VMs using closed-loop resource management. Accordingly, the workload SLAs can be guaranteed and the datacenter utilization is improved. Page coloring mechanism[2] assigns a

[2]Page coloring mechanism is the traditional cache and memory management technique in the operating system.

Table 6 Comparison of Approaches to Mitigate Contention of VMs on Shared Storage Resource of a Single Datacenter

Approaches	Objective	Optimization	Complexity	Techniques	Effectiveness
PARDA [66]	Storage I/O isolation and fairness	Weighted sharing	Medium	I/O queue control, fair I/O scheduler	Guarantee I/O performance, fairness
SRP [67]		Minimum guarantee, limits, weighted sharing	Medium	I/O queue control, demand prediction	
Pisces [68]		Weighted sharing, high utilization	Medium	I/O queue control, data placement, replica selection	max-min ratio: 0.99, I/O throughput improvement: 20%
Tarasov *et al.* [69]		Storage cache allocation	Medium	I/O classification, data pre-fetching	I/O throughput improvement: 11 – 95%
Soundararajan *et al.* [70]	SLO guarantee of storage I/O service	I/O bandwidth allocation	Low	I/O bandwidth partitioning	Minimize SLO violation of I/O service
Horizon [71], QBox [72]		Storage I/O requests scheduling	Low	I/O requests re-ordering	92 – 99% deadlines of I/O requests are met
Pesto [48]	Storage I/O hotspot mitigation	I/O Load balancing, migration cost, benefit	Medium, $\mathcal{O}(m{\cdot}n)$	VM assignment, I/O performance modeling	Peak I/O latency reduction: 19%
Romano [73]		Migration benefit, storage I/O interference	High, $\mathcal{O}(n^2)$		Average I/O latency reduction: 58%

different color to each virtual memory page and physical cache page, where the virtual memory page can only be allocated to the physical cache page with the same color. Such a technique can accordingly be used to partition the cache space and control the contention on the cache resource among corunning applications [50]. However, whether page coloring can alleviate the performance interference on the cache and memory bandwidth resource among colocated VMs requires further investigation.

To control the disk I/O contention among colocated VMs on a physical server, one possible solution is to leverage the disk I/O scheduling technique on the VM and VMM levels. For example, Boutcher and Chandra [51] empirically show that the aggregate amount of I/O throughput of colocated VMs can significantly be improved, by choosing the appropriate disk I/O schedulers of VM and VMM from the existing Linux I/O schedulers. In particular, the disk I/O fairness among colocated VMs can only be achieved at the cost of I/O throughput and latency. To improve the response time for I/O requests with a high priority, a priority-based scheduler has been proposed in [52]. This scheduler intercepts I/O requests at the VM level, and reorders these requests in the disk I/O queue of a physical server according to their priority information at the VMM level.

Table 7 Comparison of Approaches to Mitigate Contention of VMs on Shared Network Resource of a Single Datacenter

Approaches	Hierarchy	Optimization	Complexity	Techniques	Overhead	Implementation
Seawall [28]	VM/process	Weighted sharing	Low	Rate limiting at sender	3.6% CPU cycles	Hypervisor
EyeQ [76]	VM	Minimum guarantee, weighted sharing	Low	Rate limiting at sender, receiver	3 – 20% CPU cycles	Hypervisor
Hadrian [77]	Tenant	Minimum guarantee, weighted sharing	Medium	Rate limiting at sender, receiver, switch	4% link traffic	Switch, Hypervisor
ElasticSwitch [78]	VM	Minimum guarantee, weighted sharing, scalability	Low	Guarantee partition, rate allocation	10% CPU cycles and a few Kbps	Hypervisor
FairCloud [79]	VM/tenant	Minimum guarantee, weighted sharing, high utilization	High	Weighted fair packet queuing at switch	N/A	Switch, Hypervisor
Guo *et al.* [80], [81]	VM	Minimum guarantee, weighted sharing, fairness	Medium	Nash bargaining game analysis	N/A	Switch
Gatekeeper [82]	Tenant	Minimum guarantee, limits, weighted sharing	Medium	Rate limiting at sender, receiver	10% CPU cycles	Hypervisor
hClock [83]	VM/tenant	Minimum guarantee, limits, weighted sharing	Low	I/O traffic queue scheduling and billing	Within 5% CPU cycles	Hypervisor
Oktopus [49]	VM	Homogeneous and static bandwidth	Low, $\mathcal{O}(m{\cdot}n)$	Static allocation, VM assignment	CPU, network: low	Hypervisor
TIVC [84]	VM	Time-varying bandwidth	High, $\mathcal{O}(m{\cdot}n)$	Profiling demand, VM assignment	CPU, network: moderate	Switch, router
Zhu *et al.* [85]	VM	Heterogeneous bandwidth	$\mathcal{O}(n\cdot \log n)$	VM assignment	N/A	N/A

Table 8 Comparison of Approaches to Manage VM Performance Overhead Caused by Live Migration over the WAN

Approaches	Objective	Optimization	Complexity	Techniques	Effectiveness
Bradford *et al.* [33]	Live WAN migration of a VM	QoS of the migrating VM	Low	Write throttling, IP tunneling	Migrate a running Web server in the WAN
Akiyama *et al.* [88]	Live WAN migration of a data-intensive VM	Migration time	Low	Page cache restoring	Migration time reduction: 50%
VMFlockMS [89]	Live WAN migration of multiple VMs	Migration time, downtime	Medium	De-duplication, data priority transfer	Migration speedup: 3.5
CloudNet [32]		Migration time, downtime, traffic	High	De-duplication, page delta transfer	Migration time reduction: 65%
Mashtizadeh *et al.* [90]	Storage migration in a LAN or WAN	Migration time, downtime	Medium	I/O mirroring	Application downtime: nearly zero
Hirofuchi *et al.* [91]	Storage migration over the WAN	Migration time	Medium	On-demand fetching, background copying	Migration time reduction: 40%
Nicolae *et al.* [92]		Migration time, network traffic	Medium	Modified data transfer, data pre-fetching	Migration speedup: 10
Zheng *et al.* [93]		Network traffic, I/O performance	High	data blocks scheduling on locality, popularity	Network traffic reduction: 51 – 86%

The performance interference on network bandwidth among colocated VMs is managed through two kinds of solutions in existing studies. On the one hand, it is reported that the network bandwidth of VMs can be isolated from the other colocated VMs by hardware solutions, such as installing multiple network adapters or multiqueue network adapters for the VMs on a physical server [53]. On the other hand, the network bandwidth of VMs can be guaranteed by software solutions (i.e., bandwidth capping techniques) via *statically* setting a bandwidth cap in the VM configuration file before booting the VM [8], or limiting the network bandwidth using the "*tc*" *utility* in the Linux kernel [6]. To allow scalable and dynamic programmable rate limiting with low CPU overhead, NicPic [54] classifies and stores the network packets in transmission queues in memory, and schedules the packets from memory to network interface card (NIC) using direct memory access according to the rate limits of VMs.

2) Optimization of VM Assignment: In addition to the resource isolation techniques, it is also effective in alleviating VM performance overhead by finding an optimal mapping of VMs to physical servers. Specifically, by greedily minimizing the VM performance overhead calculated by the model introduced in (3), (4), or (5), such an optimized VM assignment[3] can accordingly minimize the performance interference among colocated VMs [35], [37]. As more physical servers are provisioned, the performance interference of VMs will be reduced while leading to higher energy consumption. Accordingly, to jointly balance the tradeoff between energy consumption and VM performance, Cuanta [16] and pSciMapper [19] determine the VM assignment and the number of physical servers, by greedily maximizing the ratio of VM performance to VM power. A similar performance-aware VM assignment mechanism has been proposed in [55], where the cloud scheduler greedily alleviates the VM contention on last-level cache (LLC) in the entire cloud system. Specifically, for the physical server with maximum LLC misses and the physical server with minimum LLC misses, this approach iteratively swaps the VM with maximum LLC misses on the former server and the VM with minimum LLC misses on the latter server, via live migration of VMs.

To efficiently identify and manage the performance interference of VMs, DeepDive [10] first qualitatively pinpoints culprit resource (i.e., the resource which is the source of interference) using a clustering technique of low-level VM metrics. Then, this approach migrates the VM consuming the culprit resource most aggressively to other servers, so that the VM interference is sufficiently mitigated or totally eliminated. With a particular focus on the network I/O resource, Xu *et al.* [56] observe that the

[3]VM assignment decides which VM is assigned to which physical server. It requires live migration of VMs.

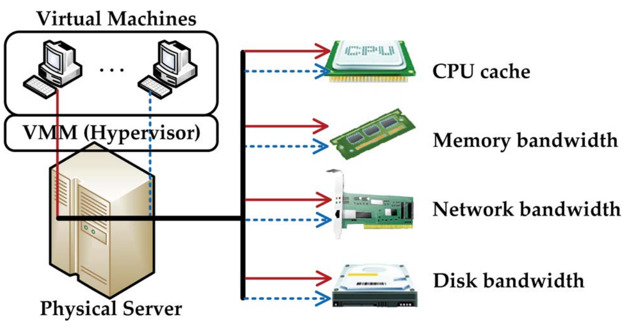


Fig. 3. *Shared computing resources among colocated VMs on a physical server. The CPU cache and I/O bandwidth resources can bring performance interference to colocated VMs in existing hypervisors.*

coscheduling of CPU-bound and latency-sensitive applications on a physical server can cause long I/O latency to VMs. Based on such an observation, they further design a simple VM examining method, which assigns latency-sensitive applications to well-performing VMs, avoiding the colocation with bad neighboring VMs.

B. Mitigating VM Performance Overhead Within a Single Datacenter

As analyzed in Section II-B, the routine operations of an IaaS cloud datacenter including (concurrent) live migration, deployment, and snapshotting of multiple VMs, and the contention of multiple VMs on the shared datacenter network and storage resources, can severely degrade the performance of VMs. We review and summarize the existing approaches that mitigate the performance overhead of VMs within a single datacenter in Tables 5–7, according to the aforementioned three aspects, respectively.

1) Live Migration, Deployment, and Snapshotting of Multiple VMs

a) Live migration of VMs: The migration process causes different aspects of performance overhead to VMs in terms of migration time, application downtime, network traffic, and energy consumption [43]. To alleviate such performance overhead, one intuitive method is to reduce the amount of transferred data during live migration of VMs. For example, by taking advantage of plenty multicore CPU resource, MECOM [57] utilizes the traditional data compression technique to significantly reduce the amount of transferred memory pages on the migration source server. Yet, this method requires decompressing those transferred memory pages to restore VM memory state on the migration destination server.

To further reduce the network traffic during live migration of VMs, several practical approaches have been proposed in existing studies (e.g., [58], [60], and [61]). Specifically, instead of transferring the entire dirty memory pages in each round of the precopy phase [21]: 1) the delta compression technique [60] compresses and transfers the data changes between the current and previous versions of memory pages, particularly for the applications with a fast dirty rate of memory pages over a slow network; 2) the memory "pruning" approach [58], [59] identifies and transfers the necessary VM memory pages which are mandatory for the VM to run correctly on the destination server after the VM migration, such as the application data and the OS kernel data; 3) CR/RT-Motion [61] first transfers a small amount of logs for the VM execution events traced on the migration source server, and then the VM memory state can be restored by replying the execution trace logs on the migration destination server; and 4) Jo *et al.* [62] only transfer the memory pages that are not available on the shared storage, along with a list of storage blocks with the memory page locations in order to restore the VM memory state on the destination server.

While the approaches above focus on reducing the amount of transferred data to improve the performance of live VM migration itself, PMigrate [63] aims to parallelize the data and processes of the migration operation using abundant CPU and network resources in order to accelerate live migration of VMs. Different from these approaches above, Liu *et al.* [43] alleviate the performance overhead of the migrating VMs by selecting the VM with the smallest migration cost. In particular, the migration cost is quantified by the migration performance model described in (10).

Furthermore, the migration process also deteriorates the performance of VMs (including the migrating VM) on the migration source and destination servers [47]. As shown in Fig. 4, the migration traffic interferes with the application traffic among the colocated VMs on both migration source and destination servers. In addition, the migration process consumes a certain amount of CPU cycles in *domain*-0 [44]: 1) to mitigate the performance overhead on the migrating VM, CloudScale [41] chooses to migrate the VM with the smallest SLO penalty which is measured offline for each VM application; and 2) to further alleviate the performance overhead of VMs on migration source and destination servers, *iAware* [47] chooses to migrate the VM with the least performance interference among the colocated and migrating VMs. In particular, the interference is predicted online by a multiresource demand/supply model. With a particular focus on the performance of the migrating VM and the migration itself, Breitgand *et al.* [45] develop a mathematical model to learn the tradeoff between the migration time and VM performance, as described in (12). By jointly minimizing the migration time and improving the performance of the migrating VM, this approach allocates an appropriate network bandwidth to the live VM migration.

Given the same VM candidates for live migration, it has been shown that the performance of concurrent live migration of these VMs is much worse than that of sequential live migration of these VMs [44]. To mitigate the migration performance overhead (i.e., reducing the network traffic and migration time), the approach proposed in [24]

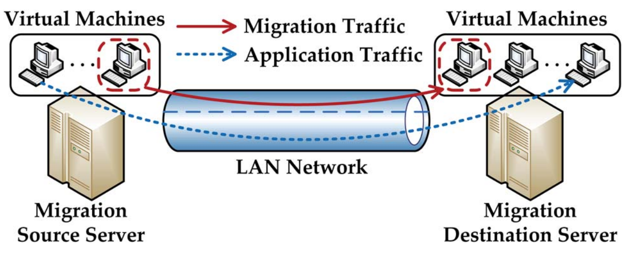


Fig. 4. *Typical live migration process of a single VM. The migration traffic occupies a certain amount of network bandwidth, which inevitably constrains the performance of colocated VMs on migration source and destination servers during live migration of VMs.*

still focuses on reducing the amount of transferred data. Specifically, this method de-duplicates the memory pages among colocated VMs and transfers the identical memory pages only once, during the concurrent live migration of the VMs hosted on a physical server.

b) Deployment and snapshotting of VMs: To reduce the deployment time of multiple VMs, traditional approaches adopt the peer-to-peer (P2P) technique, which allows the VMs booting from the same image to share the content of VM image among themselves [74]. However, it cannot mitigate the potential VM performance overhead caused by a significant amount of network traffic due to the concurrent VM deployment. To mitigate such a traffic overhead, VDN [64] proposes a VM image distribution network based on the cross-image duplication characteristic, i.e., the similarity among real-world VM disk images (e.g., VMware images and Amazon machine images) can be as high as 60% [75]. Specifically, this method divides VM images into disjoint chunks, and allows the VMs to share these chunks in a topology-aware network during provisioning these VMs.

As observed that only a small part of the VM image requires to be loaded for the VM startup [65], the booting part of VM images can be cached in compute nodes in order to reduce the VM booting time and the traffic load to datacenter networks. In addition, a lazy VM deployment mechanism has been proposed in [25], where the VM image blocks are fetched as required by the execution of applications, so as to avoid transferring and loading the unnecessary image data to VMs. VMTorrent [26] further refines this lazy deployment method by prefetching the image data blocks, according to the offline profile information of block access patterns for each pair of a workload and a VM image. To reduce the network traffic caused by the concurrent VM snapshotting, Nicolae *et al.* [25] propose to save the image difference between the current image snapshot and the former-stored VM image to the persistent storage, rather than the snapshot of a whole image file.

2) Contention of Multiple VMs on Shared Storage Resources: Different from the disk I/O sharing among colocated VMs on a single physical server discussed in Section IV-A, shared cloud storage service deploys an entire storage system, such as network attach storage (NAS) and storage area network (SAN), to support I/O requests from multiple VMs hosted on multiple physical servers, as illustrated in Fig. 5. To isolate the storage I/O resource among multiple VMs, the straightforward approach is to well manage and schedule the requests in I/O queues on each physical server and shared storage system. For example, PARDA [66] first controls the length of I/O queue on each physical server based on the aggregated weights of the hosted VMs. Then, this method uses a local fair I/O scheduler to achieve proportional-share fairness of storage I/O resource among colocated VMs within each physical server.

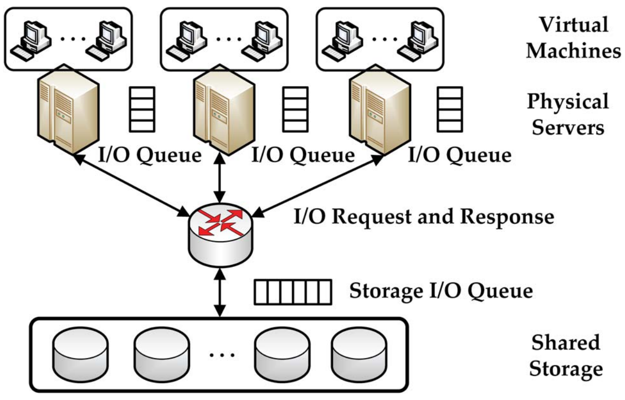


Fig. 5. ***Typical shared storage architecture in the datacenter.***

In addition to the proportional share of storage I/O resource, SRP [67] further provides several other storage resource controls, such as the minimum guarantee and limits of storage I/O bandwidth for an individual VM or a group of VMs. In particular, the I/O reservations, limits, and proportional shares of I/O queue capacity for each VM and physical server can be determined by periodically predicting the workload demands. With a particular focus on the replicated key-value shared storage, Pisces [68] deploys a series of novel mechanisms to achieve the performance isolation and max–min fairness among tenants (rather than VMs). Specifically, these mechanisms include placing and replicating the data blocks across multiple storage devices based on the weights of tenants, as well as scheduling VM I/O requests to storage devices using the deficit (weighted) round robin algorithm. To alleviate the interference on the shared cache of storage arrays among VMs, Tarasov *et al.* [69] fairly allocate the storage cache space according to the classification of VM I/O operations and data. The data prefetching in the storage cache is further applied to improve the I/O throughput of shared storage.

While the former approaches provide the isolation on sharing storage I/O resource among multiple VMs or tenants, they cannot guarantee the SLO of VM applications, such as the latency and throughput of storage I/O service, due to the storage I/O contention. To achieve this goal, Soundararajan and Amza [70] first translate the SLO into the storage I/O allocation by a reinforcement learning algorithm. Then, this method dynamically partitions the I/O bandwidth based on the translated storage I/O allocation using I/O schedulers on physical servers and shared storage. Two SLO-oriented storage I/O scheduling approaches have been proposed in [71] and [72], where the high-level SLO of VM application is translated into a series of deadlines for application I/O requests based on the historical I/O statistics. These I/O requests can be completed before their specific translated deadlines, by reordering them in the storage I/O queue using the early deadline first (EDF) scheduling algorithm.

To mitigate the disk I/O contention in the cloud storage, another practical approach is to leverage live migration of virtual disks with heavy I/O requests. For example, Pesto [48] first constructs an empirical linear model of latency and throughput of I/O requests in the shared storage. Then, such a model is leveraged to perform I/O load balancing with a cost-benefit analysis for all migration choices of virtual disks, so as to alleviate the I/O performance variation, i.e., (2). As compared with the former method, a more accurate I/O performance model has been developed in [73], which uses statistical modeling approaches (i.e., the analysis of variance and linear regression) and takes the interference of colocated virtual disks into account. Based on such a model, Romano [73] designs an algorithm to find the global pseudo-optimal mapping of VMs to physical servers using the simulated annealing approach, to reduce the variation of storage I/O latency in datacenters.

3) Contention of Multiple VMs on Shared Network Resources: Currently, several commercial cloud providers (e.g., Amazon EC2 [2]) cannot provide the performance guarantee of network resource for tenants (or VMs). As discussed in Section II-B, the network bandwidth and latency between two VMs can vary significantly over time. Fortunately, the issue of network performance isolation among tenant VMs in an IaaS cloud datacenter has been extensively studied by recent research, which can be broadly classified into two categories: weighted competition sharing and VM-assignment-based sharing.

a) Weighted-competition-sharing approach: The weighted competition approach is to share the network resource among competing entities (i.e., processes, VMs, or tenants) according to their weights, as illustrated in Fig. 6. In particular, the weights of entities can be assigned by cloud providers [81] or according to minimum bandwidth guarantees [76] and tenant payments [77]. A typical example of this approach has been proposed in Seawall [28], where the network bandwidth on the contention links is fairly allocated among multiple data senders using a VMM-based rate limiter (e.g., NicPic [54]) according to their assigned weights. However, this approach fails to offer minimum bandwidth guarantees to VMs or tenants.

To support minimum guarantees of network bandwidth and share the network resource among VMs in a fine-grained timescale (i.e., milliseconds), EyeQ [76] proposes an end-to-end rate control mechanism along the data transmission path, by implementing rate limiters on the data sender and congestion detectors on the data receiver, respectively. This approach also achieves high utilization of the cloud network resource. To particularly provide bandwidth guarantees for intertenant communication, Hadrian [77] allocates the network bandwidth to inter-tenant flows using the rate limiters on data senders, receivers, and weighted fair queueing in switches. As this approach requires traffic shaping (e.g., vShape [86]) in network switches, the implementation complexity is increased accordingly. ElasticSwitch [78] allocates the pairwise VM-to-VM bandwidth based on the network bandwidth guarantees of VMs, and distributes the unused bandwidth to VMs according to their weights. To reduce the implementation overhead and improve the scalability, this approach is fully implemented in hypervisors and without centralized coordination.

By proposing and comparing three bandwidth allocation policies [proportional sharing at the link level (PS-L), at the network level (PS-N), and on the proximate link (PS-P)], FairCloud [79] experimentally explores the tradeoff space among three requirements for weighted bandwidth sharing of datacenter networks, including minimum guarantee, network proportionality, and high utilization. Specifically, the minimum guarantee and network proportionality cannot be achieved simultaneously (i.e., hard tradeoff), and there exists a tradeoff between the network proportionality and high utilization. To obtain a theoretical insight into the tradeoff between minimum guarantee and proportional bandwidth share, Guo *et al.* [80], [81] model the bandwidth allocation as a Nash bargaining game [87], and design a distributed algorithm named *Falloc* to fairly allocate bandwidth to VMs in a cooperative manner.

In addition to achieving the weighted sharing and minimum guarantees of VM network bandwidth, two approaches [82], [83] further provide another quality-of-service (QoS) control for VM bandwidth allocation, i.e., limits of VM bandwidth. Specifically, Gatekeeper [82] guarantees the network bandwidth of VMs by implementing a weighted packet scheduler on data senders, and determining the bandwidth allocation according to the minimum and maximum rates on data receivers. In particular, this method focuses on guaranteeing the aggregate end-to-end network bandwidth of VMs for tenants in datacenters. hClock [83] allocates VM bandwidth in a hierarchical manner through maintaining and scheduling I/O requests in the leaf queue of VMs. In more detail, the requests are scheduled by periodically checking the minimum guarantees, limits, and weights of VMs, and

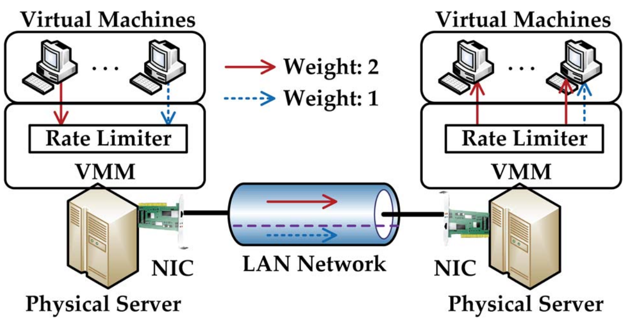


Fig. 6. ***Network bandwidth sharing among VMs according to their respective weights.***

the scheduling of I/O requests is started at the root node of the leaf queue in the hierarchy.

b) VM-assignment-based sharing approach: In response to the VM leasing request from tenants, the VM-assignment-based approach allocates the network resource to tenant VMs and meets their bandwidth requirements by assigning these VMs to an appropriate network location (physical server). A representative example of this approach has been presented in Oktopus [49], where the VM leasing request is simply abstracted to a tuple $\langle N, B\rangle$ (i.e., the number of VMs N with a homogenous network bandwidth B). According to the network abstractions of VM leasing requests, the network resource is statically allocated to tenant VMs, by greedily assigning these VMs to the smallest subtree (i.e., a server, a rack, and a pod) that can host them.

To break the assumption of static and homogeneous network bandwidth B of the VM leasing request, two approaches [84], [85] extend the basic network abstraction model $\langle N, B\rangle$ to more realistic models, where tenants are allowed to specify time-varying (i.e., temporal) and heterogeneous (i.e., spatial) allocations of network bandwidth for their leased VMs, respectively. The objective of VM allocations in these two approaches is to improve the utilization of network resource, so as to support more tenant VMs in a datacenter than Oktopus [49]. Specifically, by profiling the network bandwidth demands offline for each VM application, TIVC [84] fits the leased VMs into four types of network abstractions with the time-varying bandwidth. According to the fitted abstraction models of these VMs, this method then assigns them to the datacenter by a first-fit VM assignment algorithm. Zhu *et al.* [85] propose an abstraction model of heterogeneous VM network bandwidth allocations, and mathematically prove the problem of VM allocations with heterogeneous bandwidth demands to be NP-complete. In addition, an online heuristic VM assignment algorithm is developed to place the tenant VMs with heterogeneous bandwidth demands into an appropriate subtree of the datacenter network.

C. Controlling VM Performance Overhead Across Multiple Geodistributed Datacenters

Live migration of VMs or storage over the WAN is a common tool to manage computing resources of multiple geodistributed datacenters (e.g., load balancing among datacenters and disaster recovery of IT systems). However, it brings noticeable performance overhead of VMs in IaaS clouds, as we have discussed in Section II-C. In this section, we review and compare the solutions that alleviate the performance overhead of VMs caused by the live WAN migration in Table 8.

1) Live WAN Migration of VMs: As illustrated in Fig. 7, live WAN migration of VMs requires transferring the VM disk image and memory state as well as the ongoing network connections via the WAN, which is more complicated than live VM migration in a LAN. Bradford *et al.* [33] make the first attempt to migrate a single VM hosting a running Web server over the WAN. Specifically, the disk image and VM memory state are iteratively copied to the destination to complete the whole live VM migration. To particularly deal with the migrating VM hosting write-intensive workloads, a write throttling mechanism is proposed. During transferring the disk or memory state, write throttling reduces the number of dirty pages by restricting the number of write operations below a predefined threshold. Furthermore, the existing network connections of the migrating VM can be transparently redirected to the VM migration destination server by the IP tunneling technique, which dynamically routes the network packets to the new IP address of the migrating VM. To accelerate the migration of data-intensive applications over the WAN, Akiyama *et al.* [88] propose page cache teleportation which reduces the amount of transferred memory for the VM page cache. In particular, the page cache of the migrating VM can be restored on the destination server based on its memory location which is identified before the migration.

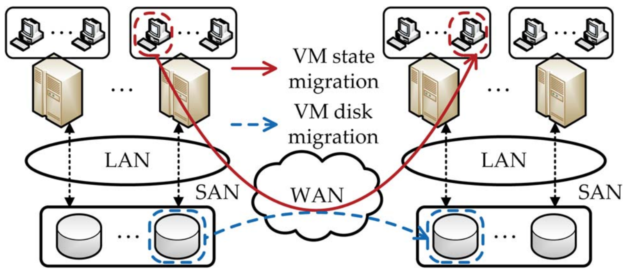


Fig. 7. ***Live WAN migration of a single VM across multiple geodistributed datacenters.***

To reduce the migration time and network traffic caused by the live WAN migration of multiple VMs, one straightforward, yet effective, approach is de-duplication. Specifically, this approach explores the duplicate contents among the disk images and VM memory states of the migrating VMs, and transfer only one copy of them [32], [89]. Particularly, to reduce the application downtime of the migrated VMs, VMFlockMS [89] adopts the data prioritization transfer mechanism by transferring the data that is responsible for booting the VM and running application services with the highest priority. With the same objective, CloudNet [32] proposes a page delta approach which only sends the difference between the current memory page and the page previously transferred, thereby reducing the rounds of memory precopy and the number of dirty pages. Generally, the migration cost of transferring the VM disk image is much higher than that of transferring the VM memory state. To avoid such a high migration cost of disk image transfer, one speculative approach [32] is to replicate the VM images in the background before live migration of VMs, and synchronize the disk state during live migration.

2) Live WAN Migration of VM Storage: Live migration of VM storage (i.e., one or more VM disk images) is a critical part of live migration of VMs over the WAN, which has recently attracted much interest from researchers. In order to guarantee a minimal application downtime, Mashtizadeh *et al.* [90] develop a technique named I/O mirroring in VMWare ESX. During the transfer of the VM disk image, I/O mirroring blocks the write operations on the migration source server until these operations are finished on the destination server. Hirofuchi *et al.* [91] propose to fetch the required data blocks on demand from the migration source server, and transfer the remaining disk data blocks to the destination server using the available bandwidth of the WAN. However, these two approaches above can cause high disk I/O latency and low I/O throughput during the migration of I/O-intensive workloads. Particularly, to deal with the I/O-intensive workloads, Nicolae and Cappello [92] further optimize the migration strategy by two mechanisms: 1) transferring only the modified content of VM disk image except the VM operating system to the migration destination server; and 2) a hybrid data fetching technique, in which the migration source server pushes (copies) the cold data chunks when transfer control is on the source server, and the migration destination server pulls the remaining (dirty) data chunks from the source server when transfer control is on the destination server.

A workload-aware scheduling approach for storage migration has been proposed in [93], where the disk data blocks are scheduled in an optimized order for migration, so that the migration network traffic is reduced and the storage I/O performance is improved during the live WAN migration of storage. In particular, the scheduled order of data blocks is determined by tracking the temporal and spatial locality characteristics of the read and write operations, as well as the popularity characteristic of data blocks of the migrating workload.

D. Summaries, Insights, and Open Research Issues

From the above detailed review and comparison of the existing techniques that mitigate the performance overhead of VMs in Section IV-A–C, we bring forth the following summaries, insights, and open research issues.

First, we come to the conclusion that live migration of VMs serves as the cornerstone of managing the computing resources and alleviating the performance overhead of VMs in IaaS clouds. Specifically, in addition to enabling the routine operations, including load balancing and power saving [47], live VM migration has also been proved to be effective in controlling the VM performance overhead in diverse scenarios, ranging from the single-server virtualization (e.g., [10]) to a single datacenter (e.g., shared network resource [49] and shared storage resource [73]). In particular, a number of overhead mitigation techniques (e.g., [10] and [48]) require adjusting the mapping of VMs to physical servers. Such a VM adjusting process is enabled by live VM migration during the execution of workloads. As a result, managing the performance overhead of VMs caused by live VM migration (Section IV-B1) plays a critical role in guaranteeing the performance of VMs in the IaaS cloud.

Second, the conclusion above gives an impetus to mitigate the performance overhead of VMs caused by live VM migration in a cost-effective manner. To improve the performance of live VM migration itself in a LAN or a WAN, existing techniques mainly rely on two aspects (refer to Tables 5 and 8): reducing the amount of transferred data during live VM migration and lowering the memory or disk dirty rate, specifically: 1) a number of techniques have been proposed to reduce the amount of transferred data for migration, for example, memory page compression (e.g., [57]), logging and replying VM execution events [61], memory pages pruning [58], spatial de-duplication of memory content and disk images of multiple VMs (e.g., [24] and [89]), and temporal de-duplication of the current and former memory pages of the migrating VM [32], [60]; and 2) to control the dirty rate of memory or disks, existing techniques focus on throttling or blocking I/O requests to the migrating VM during the migration (e.g., [33] and [90]). Although these techniques can significantly reduce the migration time, they either bring noticeable CPU and memory performance overhead to colocated VMs, or significantly impact the QoS of applications running on the migrating VM. Therefore, how to design a less "noisy" live migration approach while improving the performance of live VM migration itself will continue to be a challenge.

Third, although a number of migration policies have been extensively studied in the literature (e.g., [41] and [43]), they focus on performance overhead of either migrating VM or colocated VMs on migration source and destination servers during live migration of VMs (refer to Table 5). Few migration policies have devoted adequate attention to the VM performance overhead caused by VM colocation on a single server after live VM migration [47]. Specifically, as the migrating VM is moved onto a new physical server after the migration, it will be likely to contend with the VMs colocated on the destination server for scarce physical resources. Accordingly, the VM performance overhead caused by VM colocation on the migration destination server cannot be overlooked (Section II-A), when evaluating a specific migration choice (i.e., a tuple of the migrating VM and migration destination server). Consequently, it is essential for the VM migration policy to jointly consider the VM performance overhead caused by both live VM migration and VM colocation (i.e., single-server virtualization) during and after migration. It would be a promising research problem to design such a performance-centric VM migration policy.

Fourth, it would be an interesting research topic to explore the tradeoff between alleviating VM performance overhead and saving the monetary cost of hardware provisioning (i.e., physical servers, storage, network switches and links) in IaaS cloud datacenters. Specifically, with hosting the same number of tenant VMs, the more physical hardware the datacenter provisions, the lower performance overhead these VMs will experience. To the extreme, it is possible, but not practical, to provision each VM with a physical server, each network connection between VMs with a physical network link, and each VM virtual disk with a physical storage device [a disk or redundant array of independent disks (RAID)]. In such an environment, the performance overhead of VMs can be dramatically decreased or nearly eliminated, whereas the resource utilization can be extremely low, which apparently hurts the benefits of cloud providers. Hence, it would be beneficial for cloud providers to explore such a tradeoff in order to obtain insights into the relationship between guaranteeing the VM performance for tenants and maximizing the profits for cloud providers.

Fifth, most storage I/O-sharing solutions in the literature focus on achieving one or two of the three sharing requirements for VMs or tenants: the minimum guarantee, the proportional share, or high utilization of the storage resource (refer to Table 6). Although they have isolated well the storage resource and achieved good storage I/O performance among multiple VMs or tenants in the datacenter, whether one of the above three I/O-sharing requirements contradicts with another remains unknown. It would be interesting to explore the tradeoff space among the three requirements for sharing the datacenter storage resource. Such a tradeoff analysis can be further used for exploring the design space of storage I/O-sharing solutions. In addition, the solid state driver (SSD) is already displacing the traditional hard disk driver (HDD) in existing datacenters (e.g., Facebook, Dropbox) [94], and Amazon EC2 has recently released a new type of high I/O VM instance which is equipped with the SSD-based local storage [2]. This evidence above mandates a novel VM I/O sharing mechanism particularly for the SSD storage system, which has been seldom studied in the literature. It would be a future research problem to investigate VM I/O-sharing solutions on the SSD-based storage or the hybrid storage system of HDD and SSD in datacenters.

Sixth, a number of existing network-sharing solutions (e.g., [76], [78], [79], [82], and [84]; refer to Table 7) are based on the hose model, where all VMs are connected to a nonblocking logical switch through dedicated network connections. Although the hose model can simplify VM networking abstractions, the network capacity of central switches is unlikely to be infinite in practice. It has also been shown that, even in a datacenter network with very high bisection bandwidth, the network congestion can still happen anywhere, i.e., network links or switches [95]. As a result, the hose model does not quite conform to the networks in real-world datacenters. How to develop a realistic networking model, and then design a network-sharing solution based on such a model, becomes one potential research challenge. Furthermore, although existing network-sharing solutions have studied the VM allocations with heterogeneous bandwidth demands (e.g., [84] and [85]), few of them have focused on sharing the network resource among VMs in a heterogeneous hardware environment, which apparently complicates the bandwidth-sharing problem. Accordingly, how to design an effective VM network-sharing solution under the environment of heterogeneous physical servers, network switches and links become a promising research topic.

Seventh, as revealed by an analysis of workload traces in Google's production datacenter, the computing resource demands of cloud workloads are highly varying over short time intervals. In addition, various kinds of workloads are concurrently running in the datacenter. This evidence undoubtedly makes the prediction of workload resource demands much difficult [46]. Unfortunately, a number of VM performance management solutions (e.g., [67] and [84]) rely on an accurate prediction of workload resource demands. In such an environment with heterogeneous and dynamic workloads, how to design an effective resource prediction technique for these workloads with a reasonable prediction error becomes a compelling research problem. In addition, the issue of VM performance overhead in a heterogeneous hardware environment becomes more severe and more complicated than that in a homogeneous environment. For example, as for selecting the migration destination server for a live VM migration or evaluating the assignment of VMs to physical servers, the heterogeneity of physical servers should be taken into consideration. As the hardware heterogeneity is likely to be encountered in IaaS cloud datacenters, even within the same type of Amazon EC2 instances [14], how to incorporate the hardware heterogeneity into existing solutions of managing VM performance overhead would be an interesting problem.

Last but not least, the hybrid use of public and private clouds will become a popular computing paradigm for tenants in the near future [96]. We believe that the research on managing VM performance overhead across geodistributed datacenters will be enriched by the emerging applications or usages in the hybrid cloud, not limited to the live WAN migration of VMs (Section IV-C). For example, in the scenario of "cloud bursting" for an enterprise [32], the synchronization of VMs between the application service running in the local datacenter and the replica service running in the public cloud is mandatory. Such VM synchronization can generate a large amount of data, including VM states, snapshots, or even virtual disk images, thereby causing a significant amount of network traffic to the other VMs running in clouds. Therefore, how to alleviate such a heavy and intractable network load to tenant VMs? Can existing solutions on live WAN migration still handle such a network traffic problem? These questions are practical future research problems on guaranteeing the performance of VMs across multiple geodistributed datacenters.

V. CONCLUSION

As the trend of shifting the business service from the private datacenter to the public IaaS cloud prevails, one of the most important concerns of tenants lies in the performance overhead (i.e., unpredictable performance) of their leased VMs in the IaaS cloud. In this survey, we first elaborate on the causes of VM performance overhead, from the single-server virtualization, a single mega datacenter, to multiple geodistributed datacenters. With a broad review and a detailed comparison of the state-of-the-art approaches that manage the VM performance overhead in the aforementioned three aspects, we further extract fruitful insights into the benefits and costs of existing solutions. Although the latest techniques are able to control well the performance overhead of VMs in IaaS clouds, a series of future research issues on the modeling methods (Section III-C) and mitigation techniques (Section IV-D) of VM performance overhead remain to be solved in order to provide predictable performance of VMs and guarantee the performance SLA of applications for tenants. ■

REFERENCES

[1] Amazon.com, *Customer success. Powered by the AWS Cloud*. [Online]. Available: http://aws.amazon.com/solutions/case-studies/

[2] Amazon.com, *Amazon elastic compute cloud (Amazon EC2)*. [Online]. Available: http://aws.amazon.com/ec2/

[3] J. F. Gantz, S. Minton, and A. Toncheva, *Cloud computing's role in job creation*, Mar. 2012. [Online]. Available: http://www.microsoft.com/en-us/news/features/2012/mar12/03-05cloudcomputingjobs.aspx

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A Berkeley view of cloud computing," Univ. California at Berkeley, Berkeley, CA, USA, Tech. Rep., Feb. 2009.

[5] J. Schad, J. Dittrich, and J.-A. Q.-Ruiz, "Runtime measurements in the cloud: Observing, analyzing, and reducing variance," *Proc. VLDB Endowment*, vol. 3, no. 1-2, pp. 460–471, Sep. 2010.

[6] S. K. Barker and P. Shenoy, "Empirical evaluation of latency-sensitive application performance in the cloud," in *Proc. 1st Annu. ACM SIGMM Conf. Multimedia Syst.*, Feb. 2010, pp. 35–46.

[7] G. Wang and T. S. E. Ng, "The impact of virtualization on network performance of Amazon EC2 data center," in *Proc. 29th Conf. Inf. Commun.*, Mar. 2010, pp. 1163–1171.

[8] *Xen Users' Manual, Citrix Systems, Inc., University of Cambridge, U.K., XenSource Inc., IBM Corp., Hewlett-Packard Co., Intel Corp., AMD Inc.*, 2008. [Online]. Available: http://bits.xensource.com/Xen/docs/user.pdf

[9] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *Proc. 19th ACM Symp. Oper. Syst. Principles*, Oct. 2003, pp. 164–177.

[10] D. Novakovic, N. Vasic, S. Novakovic, D. Kostic, and R. Bianchini, "DeepDive: Transparently identifying and managing performance interference in virtualized environments," in *Proc. Annu. Tech. Conf.*, Jun. 2013, pp. 219–230.

[11] A. Li, X. Yang, S. Kandula, and M. Zhang, "CloudCmp: Comparing public cloud providers," in *Proc. 10th ACM SIGCOMM Conf. Internet Meas.*, Nov. 2010, DOI: 10.1145/1879141.1879143.

[12] Windows Azure. [Online]. Available: http://www.windowsazure.com/

[13] The Rackspace Cloud. [Online]. Available: http://www.rackspace.com/

[14] Z. Ou, H. Zhuang, J. K. Nurminen, A. Y.-Jaaski, and P. Hui, "Exploiting hardware heterogeneity within the same instance type of Amazon EC2," in *Proc. 4th USENIX Conf. Hot Topics Cloud Comput.*, Jun. 2012.

[15] B. Farley, V. Varadarajan, K. D. Bowers, A. Juels, T. Ristenpart, and M. M. Swift, "More for your money: Exploiting performance heterogeneity in public clouds," in *Proc. 3rd ACM Symp. Cloud Comput.*, Oct. 2012, DOI: 10.1145/2391229.2391249.

[16] S. Govindan, J. Liu, A. Kansal, and A. Sivasubramaniam, "Cuanta: Quantifying effects of shared on-chip resource interference for consolidated virtual machine," in *Proc. 2nd ACM Symp. Cloud Comput.*, Oct. 2011, DOI: 10.1145/2038916.2038938.

[17] SPEC CPU2006. [Online]. Available: http://www.spec.org/cpu2006/

[18] H. Lv, Y. Dong, J. Duan, and K. Tian, "Virtualization challenges: A view from server consolidation perspective," in *Proc. 8th ACM SIGPLAN/SIGOPS Conf. Virtual Execution Environ.*, Mar. 2012, pp. 15–26.

[19] Q. Zhu, J. Zhu, and G. Agrawal, "Power-aware consolidation of scientific workflows in virtualized environments," in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, Nov. 2010, DOI: 10.1109/SC.2010.43.

[20] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[21] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proc. 2nd Conf. Symp. Netw. Syst. Design Implement.*, May 2005, vol. 2, pp. 273–286.

[22] G. Jung, K. R. Joshi, M. A. Hiltunen, R. D. Schlichting, and C. Pu, "A cost-sensitive adaptation engine for server consolidation of multitier applications," *Middleware 2009*, Lecture Notes in Computer Science. Berlin, Germany: Springer-Verlag, Dec. 2009, vol. 5896, pp. 163–183.

[23] E. Cecchet, A. Chanda, S. Elnikety, J. Marguerite, and W. Zwaenepoel, "Performance comparison of middleware architectures for generating dynamic web content," in *Proc. ACM/IFIP/USENIX Int. Conf. Middleware*, Jun. 2003, pp. 242–261.

[24] U. Deshpande, X. Wang, and K. Gopalan, "Live gang migration of virtual machines," in *Proc. 20th Int. Symp. High Perform. Distrib. Comput.*, Jun. 2011, pp. 135–146.

[25] B. Nicolae, J. Bresnahan, K. Keahey, and G. Antoniu, "Going back and forth: Efficient multideployment and multisnapshotting on clouds," in *Proc. 20th Int. Symp. High Perform. Distrib. Comput.*, Jun. 2011, pp. 147–158.

[26] J. Reich, O. Laadan, E. Brosh, A. Sherman, V. Misra, J. Nieh, and D. Rubenstein, "VMTorrent: Scalable P2P virtual machine streaming," in *Proc. 8th Int. Conf. Emerging Netw. Exp. Technol.*, Dec. 2012, pp. 286–300.

[27] Amazon.com, *Amazon elastic block store (EBS)*. [Online]. Available: http://aws.amazon.com/ebs/

[28] A. Shieh, S. Kandula, A. Greenberg, C. Kim, and B. Saha, "Sharing the data center network," in *Proc. 8th USENIX Conf. Netw. Syst. Design Implement.*, Mar. 2011, p. 23.

[29] Gartner, Inc., *Gartner outlines five cloud computing trends that will affect cloud strategy through 2015*, Apr. 2012. [Online]. Available: http://www.gartner.com/newsroom/id/1971515

[30] Gartner, Inc., *Gartner says nearly half of large enterprises will have hybrid cloud deployments by the end of 2017*, Oct. 2013. [Online]. Available: http://www.gartner.com/newsroom/id/2599315

[31] Amazon.com, *Amazon virtual private cloud (Amazon VPC)*. [Online]. Available: http://aws.amazon.com/vpc/

[32] T. Wood, P. Shenoy, K. K. Ramakrishnan, and J. V. der Merwe, "CloudNet: Dynamic pooling of cloud resources by live WAN migration of virtual machines," in *Proc. 7th ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ.*, Jun. 2011, pp. 121–132.

[33] R. Bradford, E. Kotsovinos, A. Feldmann, and H. Schioberg, "Live wide-area migration of virtual machines including local persistent state," in *Proc. 3rd Int. Conf. Virtual Execution Environ.*, Jun. 2007, pp. 169–179.

[34] M. G. Kendall and A. Stuart, Eds., The Advanced Theory of Statistics. London, U.K.: Charles Griffin, 1973.

[35] Q. Zhu and T. Tung, "A performance interference model for managing consolidated workloads in QoS-aware clouds," in *Proc. 5th Int. Conf. Cloud Comput.*, Mar. 2012, pp. 170–179.

[36] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu, "An analysis of performance interference effects in virtual environments," in *Proc. Int. Symp. Perform. Anal. Syst. Softw.*, Apr. 2007, pp. 200–209.

[37] R. C. Chiang and H. H. Huang, "TRACON: Interference-aware scheduling for data-intensive applications in virtualized environments," in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, Nov. 2011, DOI: 10.1145/2063384.2063447.

[38] X. Bu, J. Rao, and C.-Z. Xu, "Interference and locality-aware task scheduling for MapReduce applications in virtual clusters," in *Proc. 22nd Int. Symp. High Perform. Distrib. Comput.*, Jun. 2013, pp. 227–238.

[39] R. Nathuji, A. Kansal, and A. Ghaffarkhah, "Q-Clouds: Managing performance interference effects for QoS-aware clouds," in

Proc. 5th Eur. Conf. Comput. Syst., Apr. 2010, pp. 237–250.

[40] Y. Wu and M. Zhao, "Performance modeling of virtual machine live migration," in *Proc. IEEE 4th Int. Conf. Cloud Comput.*, Jul. 2011, pp. 492–499.

[41] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, "CloudScale: Elastic resource scaling for multi-tenant cloud systems," in *Proc. 2nd ACM Symp. Cloud Comput.*, Oct. 2011, DOI: 10.1145/2038916.2038921.

[42] S. Akoush, R. Sohan, A. Rice, A. W. Moore, and A. Hopper, "Predicting the performance of virtual machine migration," in *Proc. IEEE Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst.*, Aug. 2010, pp. 37–46.

[43] H. Liu, C.-Z. Xu, H. Jin, J. Gong, and X. Liao, "Performance and energy modeling for live migration of virtual machines," in *Proc. 20th Int. Symp. High Perform. Distrib. Comput.*, Jun. 2011, pp. 171–182.

[44] S.-H. Lim, J.-S. Huh, Y. Kim, and C. R. Das, "Migration, assignment, and scheduling of jobs in virtualized environment," in *Proc. 3rd USENIX Conf. Hot Topics Cloud Comput.*, Jun. 2011.

[45] D. Breitgand, G. Kutiel, and D. Raz, "Cost-aware live migration of services in the cloud," in *Proc. 11th USENIX Conf. Hot Topics Manage. Internet Cloud Enterprise Netw. Services*, Mar. 2011.

[46] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proc. 3rd ACM Symp. Cloud Comput.*, Oct. 2012, DOI: 10.1145/2391229.2391236.

[47] F. Xu, F. Liu, L. Liu, H. Jin, B. Li, and B. Li, "iAware: Making live migration of virtual machines interference-aware in the cloud," *IEEE Trans. Comput.*, 2013, DOI: 10.1109/TC.2013.185.

[48] A. Gulati, G. Shanmuganathan, I. Ahmad, C. Waldspurger, and M. Uysal, "Pesto: Online storage performance management in virtualized datacenters," in *Proc. 2nd ACM Symp. Cloud Comput.*, Oct. 2011, DOI: 10.1145/2038916.2038935.

[49] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," in *Proc. ACM SIGCOMM Conf.*, Aug. 2011, pp. 242–253.

[50] X. Zhang, S. Dwarkadas, and K. Shen, "Towards practical page coloring-based multicore cache management," in *Proc. 4th ACM Eur. Conf. Comput. Syst.*, Mar. 2009, pp. 89–102.

[51] D. Boutcher and A. Chandra, "Does virtualization make disk scheduling pass?" in *Proc. HotStorage*, Oct. 2009, pp. 20–24.

[52] F. Blagojevic, C. Guyot, Q. Wang, T. Tsai, R. Mateescu, and Z. Bandic, "Priority IO scheduling in the cloud," in *Proc. USENIX Conf. Hot Topics Cloud Comput.*, Jun. 2013, pp. 1–6.

[53] Wikipedia, *Temporal isolation among virtual machines*. [Online]. Available: http://en.wikipedia.org/wiki/Temporal_isolation_among_virtual_machines

[54] S. Radhakrishnan, V. Jeyakumar, A. Kabbani, G. Porter, and A. Vahdat, "NicPic: Scalable and accurate end-host rate limiting," in *Proc. USENIX Conf. Hot Topics Cloud Comput.*, Jun. 2013, pp. 7–12.

[55] J. Ahn, C. Kim, J. Han, Y.-R. Choi, and J. Huh, "Dynamic virtual machine scheduling in clouds for architectural shared resources," in *Proc. 4th USENIX Conf. Hot Topics Cloud Comput.*, Jun. 2012, p. 19.

[56] Y. Xu, Z. Musgrave, B. Noble, and M. Bailey, "Bobtail: Avoiding long tails in the cloud," in *Proc. 10th USENIX Conf. Netw. Syst. Design Implement.*, Apr. 2013, pp. 329–342.

[57] H. Jin, L. Deng, S. Wu, X. Shi, and X. Pan, "Live virtual machine migration with adaptive memory compression," in *Proc. Int. Conf. Cluster Comput. Workshops*, Sep. 2009, DOI: 10.1109/CLUSTR.2009.5289170.

[58] A. Koto, H. Yamada, K. Ohmura, and K. Kono, "Towards unobtrusive VM live migration for cloud computing platforms," in *Proc. Asia-Pacific Workshop Syst.*, Jul. 2012, DOI: 10.1145/2349896.2349903.

[59] J.-H. Chiang, H.-L. Li, and T. C. Chiueh, "Introspection-based memory de-duplication and migration," in *Proc. 9th ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ.*, Mar. 2013, pp. 51–62.

[60] P. Svard, B. Hudzia, J. Tordsson, and E. Elmroth, "Evaluation of Delta compression techniques for efficient live migration of large virtual machines," in *Proc. 7th ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ.*, Mar. 2011, pp. 111–120.

[61] H. Liu, H. Jin, X. Liao, L. Hu, and C. Yu, "Live migration of virtual machine based on full system trace and replay," in *Proc. 18th ACM Int. Symp. High Perform. Distrib. Comput.*, Jun. 2009, pp. 101–110.

[62] C. Jo, E. Gustafsson, J. Son, and B. Egger, "Efficient live migration of virtual machines using shared storage," in *Proc. 9th ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ.*, Mar. 2013, pp. 41–50.

[63] X. Song, J. Shi, R. Liu, J. Yang, and H. Chen, "Parallelizing live migration of virtual machines," in *Proc. ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ.*, Mar. 2013, pp. 85–96.

[64] C. Peng, M. Kim, Z. Zhang, and H. Lei, "VDN: Virtual machine image distribution network for cloud data centers," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 181–189.

[65] K. Razavi and T. Kielmann, "Scalable virtual machine deployment using VM image caches," in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, Nov. 2013, p. 65.

[66] A. Gulati, I. Ahmad, and C. A. Waldspurger, "PARDA: Proportional allocation of resources for distributed storage access," in *Proc. 7th Conf. File Storage Technol.*, Feb. 2009, pp. 85–98.

[67] A. Gulati, G. Shanmuganathan, X. Zhang, and P. Varman, "Demand based hierarchical QoS using storage resource pools," in *Proc. Annu. USENIX Tech. Conf.*, Jun. 2012, pp. 1–13.

[68] D. Shue, M. J. Freedman, and A. Shaikh, "Performance isolation and fairness for multi-tenant cloud storage," in *Proc. 10th USENIX Conf. Operat. Syst. Design Implement.*, Oct. 2012, pp. 349–362.

[69] V. Tarasov, D. Jain, D. Hildebrand, R. Tewari, G. Kuenning, and E. Zadok, "Improving I/O performance using virtual disk introspection," in *Proc. 5th USENIX Conf. Hot Topics Storage File Syst.*, Jun. 2013, p. 11.

[70] G. Soundararajan and C. Amza, "Towards end-to-end quality of service: Controlling I/O interference in shared storage servers," in *Proc. 9th ACM/IFIP/USENIX Int. Conf. Middleware*, Dec. 2008, pp. 287–305.

[71] A. Povzner, D. Sawyer, and S. Brandt, "Horizon: Efficient deadline-driven disk I/O management for distributed storage systems," in *Proc. 19th ACM Int. Symp. High Perform. Distrib. Comput.*, Jun. 2010, DOI: 10.1145/1851476.1851478.

[72] D. Skourtis, S. Kato, and S. Brandt, "QBox: Guaranteeing I/O performance on black box storage systems," in *Proc. 21st Int. Symp. High Perform. Distrib. Comput.*, Jun. 2012, pp. 73–84.

[73] N. Park, I. Ahmad, and D. J. Lilja, "Romano: Autonomous storage management using performance prediction in multi-tenant datacenters," in *Proc. 3rd ACM Symp. Cloud Comput.*, Oct. 2012, DOI: 10.1145/2391229.2391250.

[74] Z. Chen, Y. Zhao, X. Miao, Y. Chen, and Q. Wang, "Rapid provisioning of cloud infrastructure leveraging peer-to-peer networks," in *Proc. 29th IEEE Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2009, pp. 324–329.

[75] S. Bazarbayev, M. Hiltunen, K. Joshi, W. H. Sanders, and R. Schlichting, "Content-based scheduling of virtual machines (VMs) in the cloud," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst.*, Nov. 2013, pp. 1–10.

[76] V. Jeyakumar, M. Alizadeh, D. Mazieres, B. Prabhakar, C. Kim, and A. Greenberg, "EyeQ: Practical network performance isolation at the edge," in *Proc. 10th USENIX Conf. Netw. Syst. Design Implement.*, Apr. 2013, pp. 297–312.

[77] H. Ballani, K. Jang, T. Karagiannis, C. Kim, D. Gunawardena, and G. O'Shea, "Chatty tenants and the cloud network sharing problem," in *Proc. 10th USENIX Conf. Netw. Syst. Design Implement.*, Apr. 2013, pp. 171–184.

[78] L. Popa, P. Yalagandula, S. Banerjee, J. C. Mogul, Y. Turner, and J. R. Santos, "ElasticSwitch: Practical work-conserving bandwidth guarantees for cloud computing," in *Proc. ACM SIGCOMM Conf.*, Aug. 2013, pp. 351–362.

[79] L. Popa, G. Kumar, M. Chowdhury, A. Krishnamurthy, S. Ratnasamy, and I. Stoica, "FairCloud: Sharing the network in cloud computing," in *Proc. SIGCOMM*, Aug. 2012, pp. 187–198.

[80] J. Guo, F. Liu, H. Tang, Y. Lian, H. Jin, and J. C. Lui, "Falloc: Fair network bandwidth allocation in IaaS datacenters via a bargaining game approach," in *Proc. IEEE Int. Conf. Netw. Protocols*, Oct. 2013, pp. 1–9.

[81] J. Guo, F. Liu, D. Zeng, J. C. Lui, and H. Jin, "A cooperative game based allocation for sharing data center networks," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 2139–2147.

[82] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. Guedes, "Gatekeeper: Supporting bandwidth guarantees for multi-tenant datacenter networks," in *Proc. 3rd Conf. I/O Virtualiz.*, Jun. 2011.

[83] J.-P. Billaud and A. Gulati, "hClock: Hierarchical QoS for packet scheduling in a hypervisor," in *Proc. 8th ACM Eur. Conf. Comput. Syst.*, Apr. 2013, pp. 309–322.

[84] D. Xie, N. Ding, Y. C. Hu, and R. Kompella, "The only constant is change: Incorporating time-varying network reservations in data centers," in *Proc. ACM SIGCOMM Conf. Appl. Technol. Architect. Protocols Comput. Commun.*, Aug. 2012, pp. 199–210.

[85] J. Zhu, D. Li, J. Wu, H. Liu, Y. Zhang, and J. Zhang, "Towards bandwidth guarantee in multi-tenancy cloud computing networks," in *Proc. 20th IEEE Int. Conf. Netw. Protocols*, Nov. 2012, DOI: 10.1109/ICNP.2012.6459986.

[86] G. Kumar, S. Kandula, P. Bodik, and I. Menache, "Virtualizing traffic shapers for practical resource allocation," in *Proc.*

USENIX Conf. Hot Topics Cloud Comput., Jun. 2013, pp. 13–18.

[87] J. F. Nash, "The bargaining problem," *Econometrica*, vol. 18, no. 2, pp. 155–162, 1950.

[88] S. Akiyama, T. Hirofuchi, R. Takano, and S. Honiden, "Fast wide area live migration with a low overhead through page cache teleportation," in *Proc. 13th IEEE/ACM Int. Symp. Cluster Cloud Grid Comput.*, May 2013, pp. 78–82.

[89] S. A.-Kiswany, D. Subhraveti, P. Sarkar, and M. Ripeanu, "VMFlock: Virtual machine co-migration for the cloud," in *Proc. 20th Int. Symp. High Perform. Distrib. Comput.*, Jun. 2011, pp. 159–170.

[90] A. Mashtizadeh, E. Celebi, T. Garfinkel, and M. Cai, "The design and evolution of live storage migration in VMware ESX," in *Proc. ATC*, Jun. 2011, p. 14.

[91] T. Hirofuchi, H. Ogawa, H. Nakada, S. Itoh, and S. Sekiguchi, "A live storage migration mechanism over WAN for relocatable virtual machine services on clouds," in *Proc. 9th IEEE/ACM Int. Symp. Cluster Comput. Grid*, May 2009, pp. 460–465.

[92] B. Nicolae and F. Cappello, "A hybrid local storage transfer scheme for live migration of I/O intensive workloads," in *Proc. 21st Int. Symp. High Perform. Distrib. Comput.*, Jun. 2012, pp. 85–96.

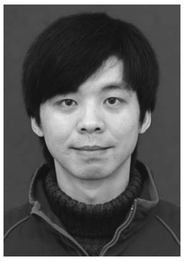
[93] J. Zheng, T. S. E. Ng, and K. Sripanidkulchai, "Workload-aware live storage migration for clouds," in *Proc. 7th ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ.*, Jun. 2011, pp. 133–144.

[94] C. Metz, "Flash drives replace disks at Amazon, Facebook, Dropbox," *Wired*, Jun. 13, 2012. [Online]. Available: http://www.wired.com/wiredenterprise/2012/06/flash-data-centers/

[95] V. Jeyakumar, M. Alizadeh, D. Mazieres, B. Prabhakar, and C. Kim, "EyeQ: Practical network performance isolation for the multi-tenant cloud," in *Proc. 4th USENIX Conf. Hot Topics Cloud Comput.*, Jun. 2012.

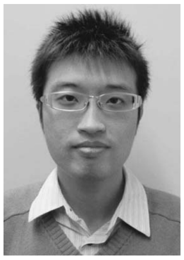
[96] L. Schlesinger, "Cloud strategists weigh in: Top trends in cloud computing," Right Scale Cloud Computing, Santa Barbara, CA, USA, Jul. 2013. [Online]. Available: http://www.rightscale.com/blog/cloud-industry-insights/cloud-strategists-weigh-top-trends-cloud-computing

ABOUT THE AUTHORS



Fei Xu is currently working toward the Ph.D. degree in computer science and technology at the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China.

His current research interests focus on cloud computing and virtualization technology.



Fangming Liu (Member, IEEE) received the B.Eng. degree in computer science and technology from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2005 and the Ph.D. degree in computer science and engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2011.

He is an Associate Professor in the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China. Since 2012, he has been a StarTrack Visiting Young Faculty member in Microsoft Research Asia (MSRA), Beijing, China. From 2009 to 2010, he was a Visiting Scholar at the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada. His research interests include cloud computing and datacenter networking, mobile cloud, green computing and communications, software-defined networking and virtualization technology, large-scale Internet content distribution, and video streaming systems.

Prof. Liu was named the CHUTIAN Scholar of Hubei Province, China. He is the Youth Scientist of the National 973 Basic Research Program Project on Software-defined Networking (SDN)-based Cloud Datacenter Networks, which is one of the largest SDN projects in China. He was the recipient of two Best Paper Awards from the 2011 IEEE Global Communications Conference, Exhibition, and Industry Forum (GLOBECOM) and the 2012 IEEE International Conference on Cloud Computing Technology and Science (CloudCom). He is a member of the Association for Computing Machinery (ACM) and the China Computer Federation (CCF) Internet Technical Committee. He was a Guest Editor for the IEEE NETWORK MAGAZINE and an Associate Editor for *Frontiers of Computer Science*, and served as the Technical Program Committee (TPC) member for the 2013–2014 IEEE International Conference on Computer Communications (INFOCOM) and the 2012–2013 GLOBECOM Conference.

Hai Jin (Senior Member, IEEE) received the Ph.D. degree in computer engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1994.

He is a Cheung Kung Scholars Chair Professor of Computer Science and Engineering at HUST. He is now the Dean of the School of Computer Science and Technology at HUST. He worked at the University of Hong Kong, Hong Kong, between 1998 and 2000, and as a Visiting Scholar at the University of Southern California, Los Angeles, CA, USA, between 1999 and 2000. His research interests include computer architecture, virtualization technology, cluster computing and grid computing, peer-to-peer computing, network storage, and network security.

Prof. Jin was awarded a German Academic Exchange Service fellowship to visit the Technical University of Chemnitz, Chemnitz, Germany, in 1996. He was awarded the Excellent Youth Award from the National Science Foundation of China in 2001. He is the Chief Scientist of ChinaGrid, the largest grid computing project in China, and the Chief Scientist of the National 973 Basic Research Program Project of Virtualization Technology of Computing Systems. He is a member of the Association for Computing Machinery (ACM).

Athanasios V. Vasilakos (Senior Member, IEEE) received the B.S. degree in electrical and computer engineering from the University of Thrace, Xanthi, Greece, in 1983, the M.S. degree in computer engineering from the University of Massachusetts, Amherst, MA, USA, in 1986, and the Ph.D. degree in computer engineering from the University of Patras, Patras, Greece, in 1988.

He is currently a Professor in the Department of Computer Science and Engineering, Kuwait University, Safat, Kuwait. He has authored or coauthored over 250 technical papers in major international journals and conferences. He is the author/coauthor of five books and 20 book chapters in the areas of communications.

Prof. Vasilakos has served as the General Chair and the Technical Program Committee Chair for many international conferences. He has served or is serving as an Editor or/and Guest Editor for many technical journals, such as the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, the IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE, the *ACM Transactions on Autonomous and Adaptive Systems*, and the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS special issues in May 2009, January 2011, and March 2011. He is the founding Editor-in-Chief of the *International Journal of Adaptive and Autonomous Communications Systems* and the *International Journal of Arts and Technology*. He is the Chairman of the Council of Computing of the European Alliances for Innovation.