Chapter 2
Network-Aware Virtual Machine Placement and Migration in Cloud Data Centers

Md Hasanul Ferdaus
Monash University, Australia

Rodrigo N. Calheiros
The University of Melbourne, Australia

Manzur Murshed
Federation University, Australia

Rajkumar Buyya
The University of Melbourne, Australia

ABSTRACT

With the pragmatic realization of computing as a utility, Cloud Computing has recently emerged as a highly successful alternative IT paradigm. Cloud providers are deploying large-scale data centers across the globe to meet the Cloud customers' compute, storage, and network resource demands. Efficiency and scalability of these data centers, as well as the performance of the hosted applications' highly depend on the allocations of the data center resources. Very recently, network-aware Virtual Machine (VM) placement and migration is developing as a very promising technique for the optimization of compute-network resource utilization, energy consumption, and network traffic minimization. This chapter presents the relevant background information and a detailed taxonomy that characterizes and classifies the various components of VM placement and migration techniques, as well as an elaborate survey and comparative analysis of the state of the art techniques. Besides highlighting the various aspects and insights of the network-aware VM placement and migration strategies and algorithms proposed by the research community, the survey further identifies the benefits and limitations of the existing techniques and discusses on the future research directions.

1. INTRODUCTION

Cloud Computing is a recently emerged computing paradigm that promises virtually unlimited compute, communication, and storage resources where customers are provisioned these resources according to their demands following a pay-per-use business model. In order to meet the increasing consumer demands, Cloud providers are deploying large-scale data centers across the world, consisting of hundreds of thousands of servers. Cloud applications deployed in these data centers such as

DOI: 10.4018/978-1-4666-8213-9.ch002

web applications, parallel processing applications, and scientific workflows are primarily composite applications comprised of multiple compute (e.g., Virtual Machines or VMs) and storage components (e.g., storage blocks) that exhibit strong communication correlations among them. Traditional research works on network communication and bandwidth optimization mainly focused on rich connectivity at the edges of the network and dynamic routing protocols to balance the traffic load. With the increasing trend towards more communication intensive applications in the Cloud data centers, the inter-VM network bandwidth consumption is growing rapidly. This situation is aggravated by the sharp rise in the size of the data that are handled, processed, and transferred by the Cloud applications. Furthermore, the overall application performance highly depends on the underlying network resources and services. As a consequence, the network conditions have direct impact on the Service Level Agreements (SLAs) and revenues earned by the Cloud providers.

Recent advancement in virtualization technologies emerges as a very promising tool to address the above mentioned issues and challenges. Normally, VM management decisions are made by using various capacity planning tools such as VMware Capacity Planner ("VMware Capacity Planner", 2014) and their objectives are set to consolidate VMs for higher utilization of compute resources (e.g., CPU and memory) and minimization of power consumption, while ignoring the network resource consumption and possible prospects of optimization. As a result, this often leads to situations where VM pairs with high mutual traffic loads are placed on physical servers with large network cost between them. Such VM placement decisions not only put stress on the network links, but also have adverse effects on the application performance. Several recent measurement studies in operational data centers reveal the fact that there exists low correlation between the average pairwise traffic rates between the VMs and the end-to-end network costs of the hosting servers (Meng, Pappas, & Zhang, 2010). Also because of the heterogeneity of the deployed workloads, traffic distribution of individual VMs exhibit highly uneven patterns. Moreover, there exists stable per-VM traffic at large timescale: VM pairs with relatively heavier traffic tend to exhibit the higher rates whereas VMs pairs with relatively low traffic tend to exhibit the lower rates. Such observational insights of the traffic conditions in data centers have opened up new research challenges and potentials. One such emerging research area is the network-aware VM placement and migration that covers various online and offline VM placement decisions, scheduling, and migration mechanisms with diverse objectives such as network traffic reduction, bandwidth optimization, data center energy consumption minimization, network-aware VM consolidation, and traffic-aware load balancing.

Optimization of VM placement and migration decisions has been proven to be practical and effective in the arena of physical server resource utilization and energy consumption reduction, and a plethora of research contributions have already been made addressing such problems. Until recently, a handful of research attempts are made to address the VM placement and migration problem focusing on inter-server network distance, run-time inter-VM traffic characteristics, server load and resource constraints, compute and network resource demands of VMs, data storage locations, and so on. These works not only differ in the addressed system assumptions and modeling techniques, but also vary considerably in the proposed solution approaches and the conducted performance evaluation techniques and environments. As a consequence, there is a rapidly growing need for elaborate taxonomy, survey, and comparative analysis of the existing works in this emerging research area. In order to analyze and assess these works in a uniform fashion, this chapter presents an overview of the aspects of Cloud data center management as background information, followed by various state-of-the-art data center network architectures, inter-VM traffic patterns observed in production data centers followed by an elaborate taxonomy and survey of notable research contributions.

The rest of this chapter is organized as follows: Section 2 presents the necessary background information relevant to network-aware VM placement and migration in Cloud data centers; Section 3 presents a detailed taxonomy and survey of the VM placement and migration strategies and techniques with elaborate description on the significant aspects considered during the course of the classification; a comprehensive comparative analysis highlighting the significant features, benefits, and limitations of the techniques has been put forward in Section 4; Section 5 focuses on the future research outlooks; and finally, Section 6 summarizes the chapter.

2. BACKGROUND

Cloud Infrastructure Management Systems

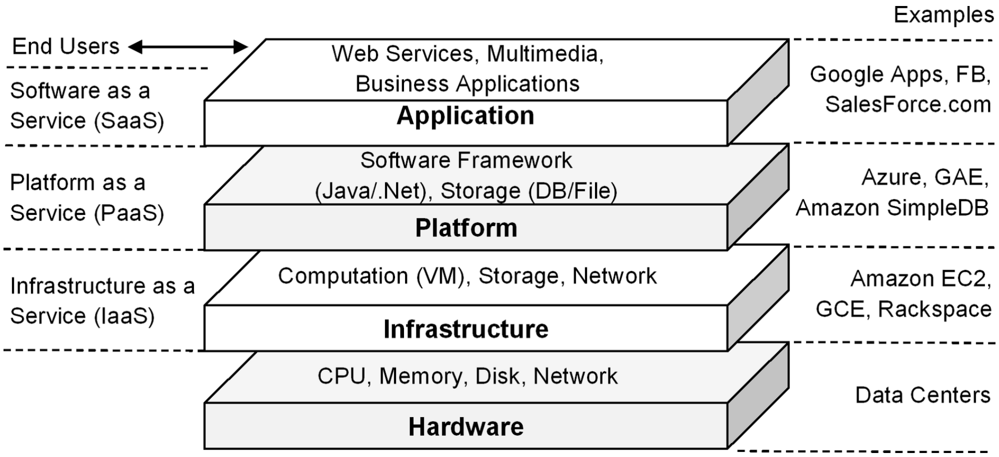
While the number and scale of Cloud Computing services and systems are continuing to grow rapidly, significant amount of research is being conducted both in academia and industry to determine the directions to the goal of making the future Cloud Computing platforms and services successful. Since most of the major Cloud Computing offerings and platforms are proprietary or depend on software that is not accessible or amenable to experimentation or instrumentation, researchers interested in pursuing Cloud Computing infrastructure questions as well as future Cloud service providers have very few tools to work with (Nurmi et al., 2009). Moreover, data security and privacy issues have created concerns for enterprises and individuals to adopt public Cloud services (Armbrust et al., 2010). As a result, several attempts and ventures of building open-source Cloud management systems came out of both academia and industry collaborations including Eucalyptus (Nurmi et al., 2009), OpenStack, OpenNebula (Sotomayor, Montero, Llorente, & Foster, 2009), and Nimbus ("Nimbus is cloud computing for science", 2014). These Cloud solutions provide various aspects of Cloud infrastructure management such as:

1. Management services for VM life cycle, compute resources, networking, and scalability.
2. Distributed and consistent data storage with built-in redundancy, failsafe mechanisms, and scalability.
3. Discovery, registration, and delivery services for virtual disk images with sup-port of different image formats (VDI, VHD, qcow2, VMDK).
4. User authentication and authorization services for all components of Cloud management.
5. Web and console-based user interface for managing instances, images, crypto-graphic keys, volume attachment/detachment to instances, and similar functions.

Figure 1 shows the four essential layers of Cloud Computing environment from the architectural perspective. Each layer is built on top of the lower layers and provides unique services to the upper layers.

1. **Hardware Layer:** This layer is composed of the physical resources of typical data centers, such as physical servers, storage devices, load balancers, routers, switches, communication links, power systems, and cooling systems. This layer is essentially the driving element of Cloud services and as a consequence, operation and management of the physical layer incurs continuous costs for the Cloud providers. Example includes the numerous data centers of Cloud providers such as Amazon, Rackspace, Google, Microsoft, Linode, and GoGrid that spread all over the globe.

Figure 1. The Cloud Computing architecture


Examples
End Users
Software as a Service (SaaS)
Web Services, Multimedia, Business Applications
Application
Google Apps, FB, SalesForce.com
Platform as a Service (PaaS)
Software Framework (Java/.Net), Storage (DB/File)
Platform
Azure, GAE, Amazon SimpleDB
Infrastructure as a Service (IaaS)
Computation (VM), Storage, Network
Infrastructure
Amazon EC2, GCE, Rackspace
CPU, Memory, Disk, Network
Hardware
Data Centers

2. **Infrastructure Layer:** This layer (also known as Virtualization Layer) creates a pool of on-demand computing and storage resources by partitioning the physical resources utilizing virtualization technologies such as Xen (Barham et al., 2003) and VMware. Efficient allocation and utilization of the virtual resources in accordance with the computing demands of Cloud users are important to minimize the SLA violations and maximize revenues.
3. **Platform Layer:** Built on top of the infrastructure layer, this layer consists of customized operating systems and application frameworks that help automate of application development, deployment, and management. In this way, this layer strives to minimize the burden of deploying applications directly on the VM containers.
4. **Application Layer:** This layer consists of the actual Cloud applications which are different from traditional applications and can leverage the on-demand automatic-scaling feature of Cloud Computing to achieve better performance, higher availability and reliability, as well as operating cost minimization.

In alignment with the architectural layers of Cloud infrastructure resources and services, the following three services models evolved and used extensively by the Cloud community:

- **Infrastructure as a Service (IaaS):** Cloud provides provision computing resources (e.g., processing, network, storage) to Cloud customers in the form of VMs, storage resource in the form of blocks, file systems, databases, etc., as well as communication resources in the form bandwidth. IaaS provides further provide management consoles or dashboards, APIs (Application Programming Interfaces), advanced security features for manual and autonomic control and management of the virtual resources. Typical examples are Amazon EC2, Google Compute Engine, and Rackspace Cloud Servers.
- **Platform as a Service (PaaS):** PaaS providers offer a development platform (programming environment, tools, etc.) that allows Cloud consumers to develop Cloud services and applications, as well as a deployment platform that hosts those services and applications, thus supports full software lifecycle management. Examples include Google App Engine and Windows Azure platform.
- **Software as a Service (SaaS):** Cloud consumers release their applications on a hosting environment fully managed and controlled by SaaS Cloud providers and the applications can be accessed through

Internet from various clients (e.g., web browser and smartphones). Examples are Google Apps and Salesforce.com.

Virtualization Technologies

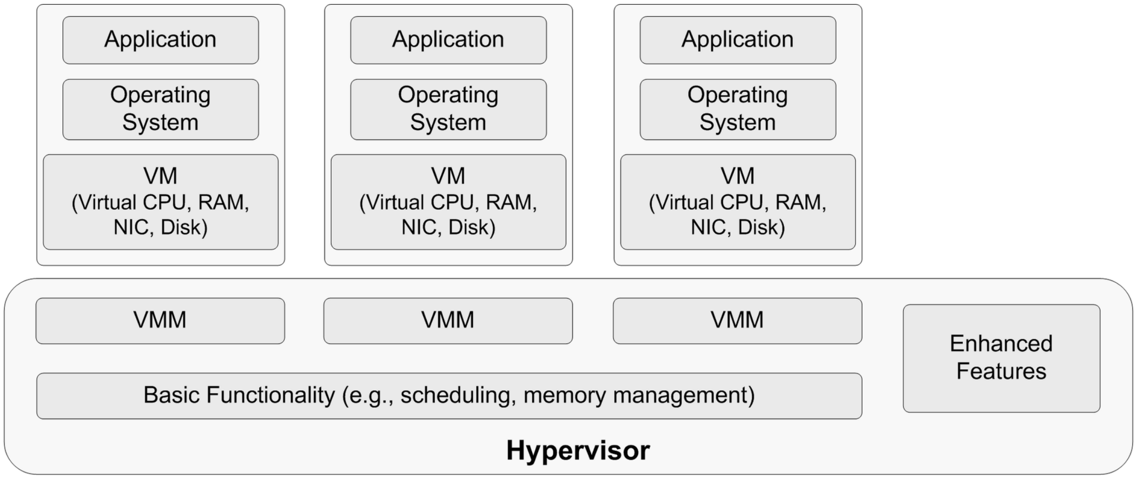
One of the main enabling technologies that paved the way of Cloud Computing towards its extreme success is virtualization. Clouds leverage various virtualization technologies (e.g., machine, network, and storage) to provide users an abstraction layer that provides a uniform and seamless computing platform by hiding the underlying hardware heterogeneity, geographic boundaries, and internal management complexities (Zhang, Cheng, & Boutaba, 2010). It is a promising technique by which resources of physical servers can be abstracted and shared through partial or full machine simulation by time-sharing and hardware and software partitioning into multiple execution environments each of which runs as complete and isolated system. It allows dynamic sharing and reconfiguration of physical resources in Cloud Computing infrastructure that makes it possible to run multiple applications in separate VMs having different performance metrics. It is virtualization that makes it possible for the Cloud providers to improve utilization of physical servers through VM multiplexing (Meng, Isci, Kephart, Zhang, Bouillet, & Pendarakis, 2010) and multi-tenancy (i.e. simultaneous sharing of physical resources of the same server by multiple Cloud customers). It also enables on-demand resource pooling through which computing resources like CPU and memory, and storage resources are provisioned to customers only when needed (Kusic, Kephart, Hanson, Kandasamy, & Jiang, 2009). This feature helps avoid static resource allocation based on peak resource demand characteristics. In short, virtualization enables higher resource utilization, dynamic resource sharing, and better energy management, as well as improves scalability, availability, and reliability of Cloud resources and services (Buyya, Broberg, & Goscinski, 2010).

From architectural perspective, virtualization approaches are categorized into the following two types:

1. **Hosted Architecture:** The virtualization layer is installed and run as an individual application on top of an operating system and supports the broadest range of underlying hardware configurations. Example of such architecture includes VMware Workstation and Player, and Oracle VM VirtualBox.
2. **Hypervisor-Based Architecture:** The virtualization layer, termed *Hypervisor* is installed and run on bare hardware and retains full control of the underlying physical system. It is a piece of software that hosts and manages the VMs on its *Virtual Machine Monitor* (VMM) components (Figure 2). The VMM implements the VM hardware abstraction, and partitions and shares the CPU, memory, and I/O devices to successfully virtualize the underlying physical system. In this process, the Hypervisor multiplexes the hardware resources among the various running VMs in time and space sharing manner, the way traditional operating system multiplexes hardware resources among the various processes (Smith & Nair, 2005). VMware ESXi and Xen Server (Barham et al., 2003) are examples of this kind of virtualization. Since Hypervisors have direct access to the underlying hardware resources rather than executing instructions via operating systems as it is the case with hosted virtualization, a hypervisor is much more efficient than a hosted virtualization system and provides greater performance, scalability, and robustness.

Among the different processor architectures, the Intel x86 architecture has been established as the most successfully, widely adopted, and highly inspiring. In this architecture, different privilege level instructions are executed and controlled

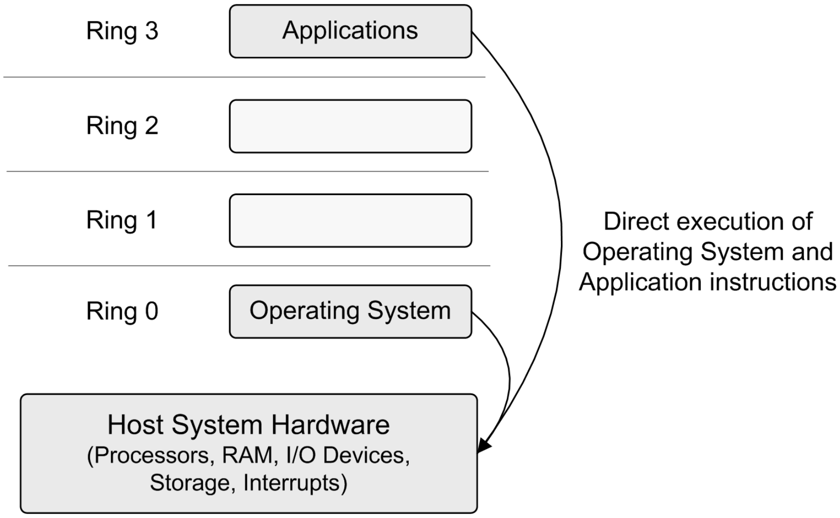
Figure 2. Hypervisor-based virtualization architecture


Application
Operating System
VM (Virtual CPU, RAM, NIC, Disk)
Application
Operating System
VM (Virtual CPU, RAM, NIC, Disk)
Application
Operating System
VM (Virtual CPU, RAM, NIC, Disk)
VMM
VMM
VMM
Enhanced Features
Basic Functionality (e.g., scheduling, memory management)
Hypervisor

through the four privilege rings: Ring 0, 1, 2, and 3, with 0 being the most privileged (Figure 3) in order to manage access to the hardware resources. Regular operating systems targeted to run over bare-metal x86 machines assume full control of the hardware resources and thus are placed in Ring 0 so that they can have direct access to the underlying hardware, while typical user level applications run at ring 0.

Virtualization of the x86 processor required placing the virtualization layer between the operating system and the hardware so that VMs can be created and managed that would share the same physical resources. This means the virtualization layer needs to be placed in Ring 0; however unmodified operating systems assumes to be run in the same Ring. Moreover, there are some sensitive instructions that have different semantics when

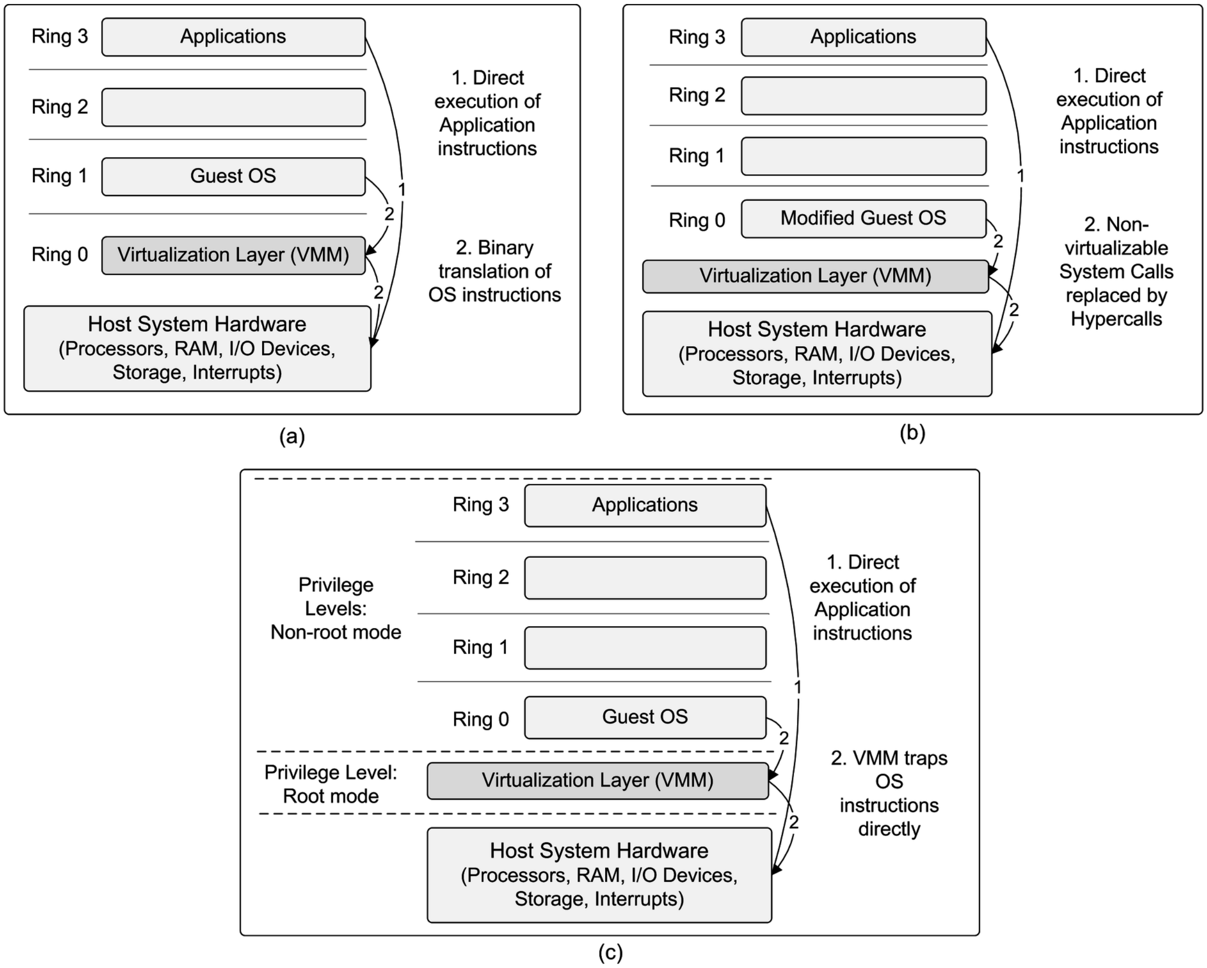
Figure 3. The x86 processor privilege rings without virtualization


Ring 3
Applications
Ring 2
Ring 1
Ring 0
Operating System
Direct execution of Operating System and Application instructions
Host System Hardware
(Processors, RAM, I/O Devices, Storage, Interrupts)

they are not executed in Ring 0 and thus cannot be effectively virtualized. As a consequence, the industry and research community have come up with the following three types of alternative virtualization techniques:

1. **Full Virtualization:** This type of virtualization technique provides full abstraction of the underlying hardware and facilitates the creation of complete VMs in which guest operating systems can execute. Full virtualization is achieved through a combination of binary translation and direct execution techniques that allow the VMM to run in Ring 0. The binary translation technique translates the OS kernel level code with alternative series of instructions in order to substitute the non-virtualizable instructions so that it has the intended effect on the virtual hardware (Figure 4(a)). As for the user level codes, they are executed directly on the processor to achieve high performance. In this way, the VMM provides the VM with all the services of the physical machine like virtual processor, memory, I/O devices, BIOS, etc. This approach have the advantage of providing total virtualization of the physical machine as the guest operating system is fully abstracted and decoupled from the underlying hardware separated by the virtualization layer. This enables unmodified operating systems and applications to run on VMs, being completely unaware of the virtualization. It also facilitates efficient and simplified migration of applications and workloads from one physical machine to another. Moreover, full virtualization provides complete isolation of VMs that ensures high level of security. VMware ESX Server and Microsoft Virtual Server are examples of full virtualization.
2. **Paravirtualization***:* Different from the binary translation technique of full virtualization, Paravirtualization (also called OS Assisted Virtualization) works through the modification of the OS kernel code by replacement of the non-virtualizable instructions with hypercalls that communicate directly with the hypervisor virtualization layer (Figure 4(b)). The hypervisor further provides hypercall interfaces for special kernel operations such as interrupt handling, memory management, timer management, etc. Thus, in paravirtualization each VM is presented with an abstraction of the hardware that is similar but not identical to the underlying physical machine. Since paravirtualization requires modification of guest OSs, they are not fully un-aware of the presence of the virtualization layer. The primary advantage of paravirtualization technique is lower virtualization overhead over full virtualization where binary translations affect instruction executing performance. However, this performance advantage is dependent on the types of workload running on the VMs. Paravirtualization suffers from poor compatibility and portability issues since every guest OS running on it top of paravirtualized machines needs to be modified accordingly. For the same reason, it causes significant maintenance and support issues in production environments. Example of paravirtualization is the open source Xen project (Crosby & Brown, 2006) that virtualizes the processor and memory using a modified Linux kernel and virtualizes the I/O subsystem using customized guest OS device drivers.
3. **Hardware Assisted Virtualization:** In response to the success and wide adaptation of virtualization, hardware vendors have come up with new hardware features to help and simplify virtualization techniques. Intel Virtualization Technology (VT-x) and AMD-V are first generation virtualization supports allow the VMM to run in a new root mode below Ring 0 by

Figure 4. Alternative virtualization techniques: (a) full virtualization through binary translation, (b) paravirtualization, and (c) hardware assisted virtualization


Ring 3
Applications
Ring 2
Ring 1
Guest OS
Ring 0
Virtualization Layer (VMM)
Host System Hardware (Processors, RAM, I/O Devices, Storage, Interrupts)
1
2
2
1. Direct execution of Application instructions
2. Binary translation of OS instructions
(a)
Ring 3
Applications
Ring 2
Ring 1
Ring 0
Modified Guest OS
Virtualization Layer (VMM)
Host System Hardware (Processors, RAM, I/O Devices, Storage, Interrupts)
1
2
2
1. Direct execution of Application instructions
2. Non-virtualizable System Calls replaced by Hypercalls
(b)
Privilege Levels: Non-root mode
Ring 3
Applications
Ring 2
Ring 1
Ring 0
Guest OS
Privilege Level: Root mode
Virtualization Layer (VMM)
Host System Hardware (Processors, RAM, I/O Devices, Storage, Interrupts)
1
2
2
1. Direct execution of Application instructions
2. VMM traps OS instructions directly
(c)

the introduction of a new CPU execution mode. With this new hardware assisted feature, privileged and critical system calls are automatically trapped by the hypervisor and the guest OS state is saved in Virtual Machine Control Structures (VT-x) or Virtual Machine Control Blocks (AMD-V), removing the need for either binary translation (full virtualization) or paravirtualization (Figure 4 (c)). The hardware assisted virtualization has the benefit that unmodified guest OSs can run directly and access to virtualized resources without any need for modification or emulation. With the help of the new privilege level and new instructions, the VMM can run at Ring -1 (between Ring 0 and hardware layer) allowing guest OS to run at Ring 0. This reduces the VMM's burden of translating every privileged instruction, and thus helps achieve better performance compared to full virtualization. The hardware assisted virtualization requires explicit virtualization support from the physical host processor, which is available only to modern processors.

Among the various virtualization systems, VMware, Xen (Barham et al., 2003), and KVM (Kernel-based Virtual Machine) (Kivity, Kamay, Laor, Lublin, & Liguori, 2007) have proved to be the most successful by combing features that make them uniquely well suited for many important applications:

- VMware Inc. is the first company to offer commercial virtualization technology. It offers VMware vSphere (formerly VMware Infrastructure 4) for computer hardware virtualization that includes VMware ESX and ESXi hypervisors that virtualize the underlying hardware resources. VMware vSphere also includes vCenter Server that provides a centralized point for management and configuration of IT resources, VMotion for live migrating VMs, and VMFS that provides a high performance cluster file system. VMware products support both full virtualization and paravirtualization.
- Xen Server is one of a few Linux hypervisors that support both full virtualization and paravirtualization. Each guest OS (termed Domain in Xen terminology) uses a pre-configured share of the physical server. A privileged Domain called Domain0 is a bare-bone OS that actually controls physical hardware and is responsible for the creation, management, migration, and termination other VMs.
- KVM also provides full virtualization with the help of hardware virtualization support. It is a modification to the Linux kernel that actually makes Linux into a hypervisor on inserting a KVM kernel module. One of the most interesting KVM features is that each guest OS running on it is actually executed in user space of the host system. This approach makes each guest OS look like a normal process to the underlying host kernel.

Virtual Machine Migration Techniques

One of the most prominent features of the virtualization system is the *VM Live Migration* (Clark et al., 2005) which allows for the transfer of a running VM from one physical machine to another, with little downtime of the services hosted by the VM. It transfers the current working state and memory of a VM across the network while it is still running. Live migration has the advantage of transferring a VM across machines without disconnecting the clients from the services. Another approach for VM migration is the *Cold* or *Static VM Migration* (Takemura & Crawford, 2009) in which the VM to be migrated is first shut down and a configuration file is sent from the source machine to the destination machine. The same VM can be started on the target machine by using the configuration file. This is a much faster and easier way to migrate a VM with negligible increase in the network traffic; however static VM migration incurs much higher downtime compared to live migration. Because of the obvious benefit of uninterrupted service and much less VM download time, live migration has been used as the most common VM migration technique in the production data centers.

The process of live-migrating a VM is much more complicated than just transferring the memory pages of the VM from the source machine to the destination machine. Since a running VM can execute write instructions to memory pages in the source machine during the memory copying process, the new dirty pages must also be copied to the destination. Thus, in order to ensure a consistent state of the migrating VM, copying process for all the dirty pages must be carried out until the migration process is completed. Furthermore, each active VM has its own share and access to the physical resources such as storage, network, and I/O devices. As a result, the VM live migration process needs to ensure that the corresponding physical resources in the destination machine must be attached to the migrated VM.

Transferring VM memory from one machine to another can be carried out in many different ways. However, live migration techniques utilize one or more of the following memory copying phases (Clark et al., 2005):

- **Push Phase:** The source host VMM pushes (i.e. copies) certain memory pages across the network to the destination host while the VM is running. Consistency of VM's execution state is ensured by resending any modified (i.e. dirty) pages during this process.
- **Stop-and-Copy Phase:** The source host VMM stops the running VM on certain stop condition, copies all the memory pages to the destination host, and a new VM is started.
- **Pull Phase:** The new VM runs in the destination host and, if a page is accessed that has not yet been copied, a page fault occurs and this page is copied across the network from the source host.

Performance of any VM live migration technique depends on the balance of the following two temporal parameters:

1. **Total Migration Time:** The duration between the time when the migration is initiated and when the original VM may be discarded after the new VM is started in the destination host. In short, the total time required to move the VM between the physical hosts.
2. **VM Downtime:** The portion of the total migration time when the VM is not running in any of the hosts. During this time, the hosted service would be unavailable and the clients will experience service interruption.

Incorporating the above three phases of memory copying, several VM live migration techniques are presented by the research communities with tradeoffs between the total migration time and VM downtime:

- **Pure Stop-and-Copy:** The VM is shut down at the source host, all the memory pages are copied to the destination host, and a new VM is started. This technique is simple and, the total migration time is relatively small compared to other techniques and directly proportional to the size of the active memory of the migrating VM. However, the VM can experience high VM downtime, subject to the memory size, and as a result, this approach can be impractical for live services (Sapuntzakis, Chandra, Pfaff, Chow, Lam, & Rosenblum, 2002).
- **Pure Demand-Migration:** The VM at the source host is shut down and essential kernel data structures (CPU state, registers, etc.) are transferred to the destination host using a short stop-and-copy phase. The VM is then started in the destination host. The remaining pages are transferred across the network when they are first referenced by the VM at the destination. This approach has the advantage of much shorter VM downtime; however the total migration time is generally much longer since the memory pages are transferred on-demand upon page fault. Furthermore, post-migration VM performance is likely to be hampered substantially due to large number of page faults and page transfers across the network (Zayas, 1987).
- **Post-Copy Migration:** Similar to the pure demand-migration approach, the VM is suspended at the source host, a minimal VM kernel data structure (e.g., CPU execution state, registers values, and non-pageable memory) is transferred to the destination host, and the VM is booted up. Unlike of pure demand-migration, the source VMM actively sends the remaining memory pages to the destination host, an activity termed *pre-paging*. When the running VM at the destination attempts to access a page that is not copied yet, a page

fault occurs (known as *network faults*) and the faulted page is transferred from the source host to the destination host over the communication network. As in the case of pure demand-migration, post-copy migration suffers from VM performance degradation due to on-demand page transfer upon page fault. However, pre-paging technique can help reduce the performance degradation by adapting the page transmission order dynamically in response to the network faults by pushing the pages near the last page fault (Hines, Deshpande, & Gopalan, 2009).

- **Pre-Copy Migration:** Unlike the above approaches, the VM continues running in the source host while the VMM iteratively transfers memory pages to the destination host. Only after a substantial amount of memory pages are copied, or a predefined number of iterations are completed, or any other terminating condition is met, the VM is stopped at the source, the remaining pages are transferred to the destination, and the VM is restarted. Pre-copy migration has the obvious benefit of short stop-and-copy phase since most of the memory page would be copied to the destination by this time. So, the VM downtime is comparatively much shorter than other live migration techniques, making this approach suitable for live services. Furthermore, pre-copy migration offers higher reliability since it retains an up-to-date state of the VM in the source machine during the migration process, an added advantage absent in other migration approaches. However, pre-copy migration can suffers from longer total migration time since the same memory pages can be transmitted multiple time in several rounds depending on page dirty rate. For the same reason, it can generate much higher network traffic compared to other techniques (Clark et al., 2005).

Almost all the modern virtualization environments offers VM live migration feature, including Xen Server, VMware ESX Server (through VMotion (Nelson, Lim, & Hutchins, 2005)), KVM, Microsoft Hyper-V, Oracle VM VirtualBox, and OpenVZ. A high level flow chart of the logical steps followed during the pre-copy migration technique implemented in Xen Server is depicted in Figure 5 (Clark et al., 2005). Focusing primarily on high reliability against system failure, the Xen pre-copy migration takes a transactional approach between the source and target hosts:

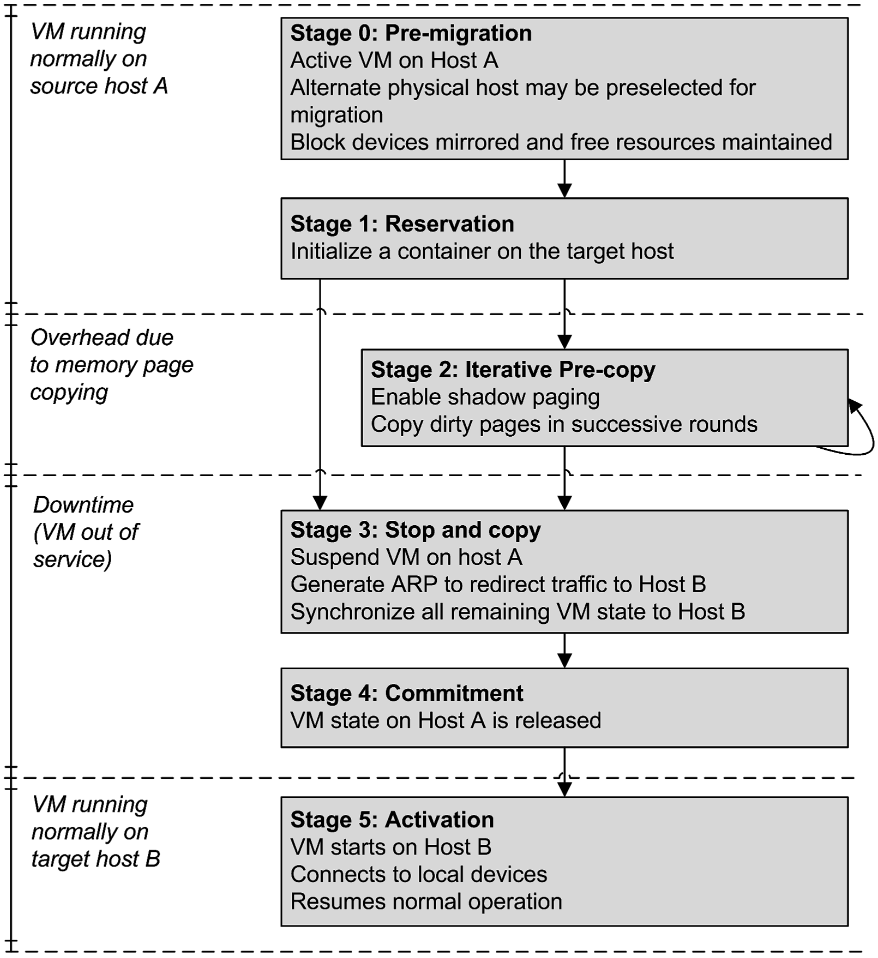
Stage 0 (Pre-Migration): Source host A has an active VM to be migrated. The target host B can be pre-selected in advance in order to speed up future migrations through guaranteed resources required for the migration process.

Stage 1 (Reservation): The request to migrate the VM from source host A to target host B is issued. Host B confirms that it has the required resources and reserves a VM container of that size. If host B fails to secure enough resources, the migration request is discarded and the VM runs on host A without any changes.

Stage 2 (Iterative Pre-Copy): In the first iteration, all the memory pages are transmitted (i.e. copied) from host A to host B. In the remaining iterations, only the pages that have been modified during the previous iteration are transmitted.

Stage 3 (Stop-and-Copy): The VM is shut down in host A and all the network traffic is redirected to host B. Then, the critical kernel data structures (e.g., CPU states and registers) and the remaining dirty pages are transmitted. At the end of this stage, the two copies of the VM at both host A and B are consistent; however, the copy at A is still considered primary and is resumed in the incident of failure.

Figure 5. Stages of the pre-copy VM live migration technique
(Clark et al., 2005).


VM running normally on source host A
Stage 0: Pre-migration
Active VM on Host A
Alternate physical host may be preselected for migration
Block devices mirrored and free resources maintained
Stage 1: Reservation
Initialize a container on the target host
Overhead due to memory page copying
Stage 2: Iterative Pre-copy
Enable shadow paging
Copy dirty pages in successive rounds
Downtime (VM out of service)
Stage 3: Stop and copy
Suspend VM on host A
Generate ARP to redirect traffic to Host B
Synchronize all remaining VM state to Host B
Stage 4: Commitment
VM state on Host A is released
VM running normally on target host B
Stage 5: Activation
VM starts on Host B
Connects to local devices
Resumes normal operation

Stage 4 (Commitment): Host B notifies host A that it has a consistent VM image. Upon receipt, host A sends the acknowledgment message indicating the commitment of the total migration transaction. After this point, the original VM at host A can be abandoned and host B is considered as the primary host of the VM.

Stage 5 (Activation): Host B activates the migrated VM. The post-migration code runs in order to reattach the device drivers at host B and advertise the moved IP addresses.

Data Center Network Architectures

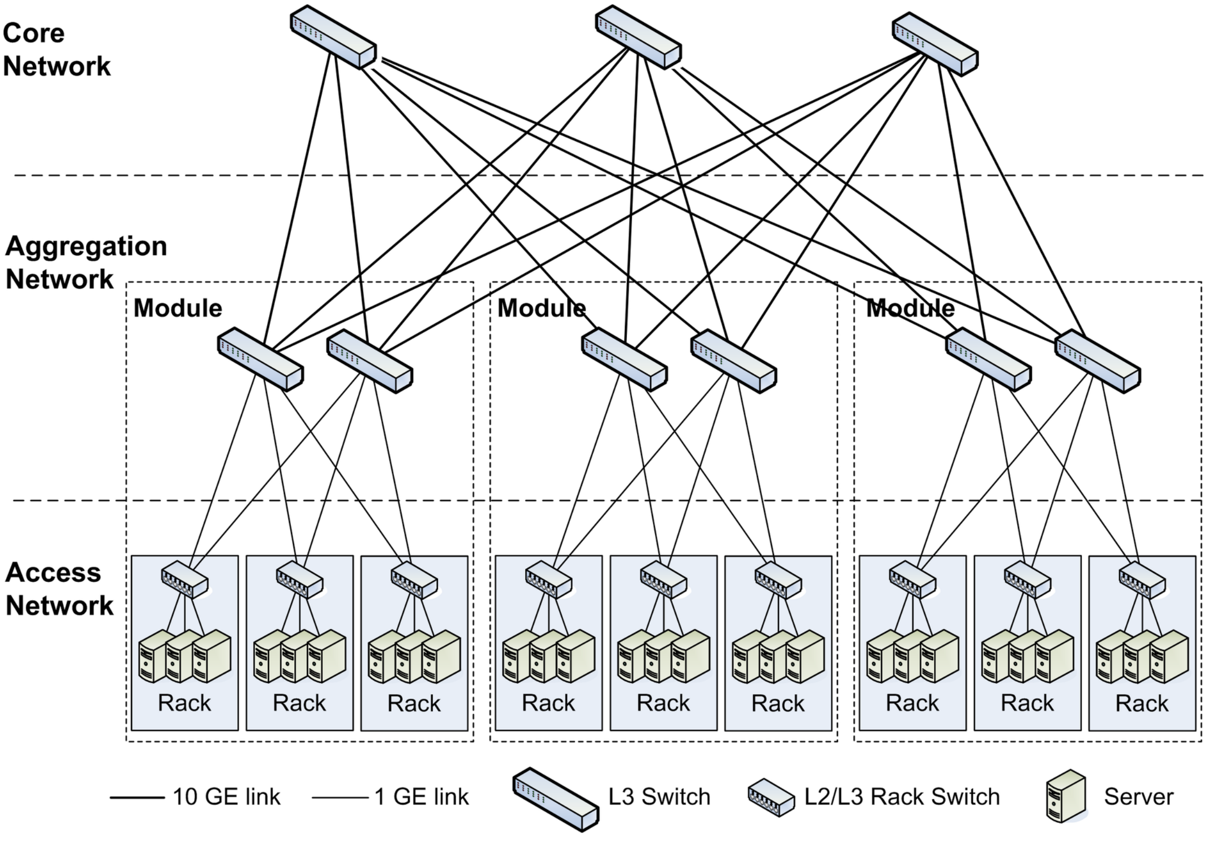
Modern data centers are built primarily according to the generic multi-tier architecture (“Cisco Data Center Infrastructure 2.5 Design Guide”, 2014). The most common network topologies follow the three-tier architecture (Figure 6), where each tier has specific responsibility and goal in the design and traffic handling. In the bottom tier, known as the *Access Tier* every physical server is connected to one or two (in case of redundancy to increase reliability) access switches, in the *Aggregation Tier*, each access switch is connected to one or two aggregation switches, and in the *Core Tier* each aggregation switch is connected to more than one core switches. The access switches provide the servers connectivity to other servers and to the upper tiers, the aggregate switches interconnects between the access switches and enables localization of traffic among the servers, and finally, the core switches connects the aggregation switches in such a way that there exists connectivity among each pair of servers and also includes gateways for the traffic to communicate outside the data center.

In three-tier network architectures, the access tier links are normally 1 Gigabit Ethernet (GE) links. Although 10 GE transceivers are available in the commodity market, they are not used for the following reasons:

1. Very high price,and
2. Bandwidth capacity is much more than needed by the physical servers.

Servers in data centers are normally grouped in ranks and rack connectivity is achieved through the use of not-so-expensive Top-of-Rack (ToR) switches. Typically, such ToR switches have two 10 GE uplinks with 48 GE links that interconnects the servers within the rack. *Oversubscription Ratio* of a switch is defined the difference between the downlink and uplink capacities of the switch and in this case it is 48:20 or 2.4: 1. As a result, though each access link has 1 GE capacity, under full load, only 416 Mb/s will be available to each server (Kliazovich, Bouvry, & Khan, 2013). At the aggregation and core tier, the racks are organized in *modules* with a couple of

Figure 6. The three-tier network architecture



aggregation switches and oversubscription ratio for these switches is around 1.5:1. Therefore, the available bandwidth for each server is reduced to 277 Mb/s.

Though such network architectures have multi-rooted forest topology at the physical level, because of the extensive use of *Virtual LANs* (VLANs) and *Spanning Tree* algorithm the network packets are forwarded according to the logical layer-2 topology. Such layer-2 logical topology always takes the form of a tree, normally rooted at one of the core switches.

Scalability issue of three-tier architecture is normally addressed through scaling up each individual switches by increasing their fan-outs, not by the scaling out of the network topology. For example, according to the Cisco Data Center Infrastructure 2.5 Design Guide, the core tier can have a maximum of 8 switches. Because of such scalability issues regarding topology scaling, high oversubscription ratio, as well as requirement for flat address space, several recent research endeavors produced complex network architectures for the large scale modern data centers and among these, the following are considered as the standard-de-facto solutions:

1. **Fat-Tree:** This is a three-tier architecture based on bipartite graphs (Al-Fares et al., 2008) and basic building block of this topology is called pods which are collections of access and aggregation switches connected in a complete bipartite graph. Every pod is connected to all the core switches; however links that connect pods to core switches are uniformly distributed between the aggregation switches contained within the pods. Such connection pattern results in a new bipartite graph between aggregation and core switches. In this topology, all the switches need to have same number of ports. The primary advantage of fat-tree topology is that $N^2/4$ paths are available to route the traffic between any two servers.
2. **VL2:** Somewhat similar to fat-tree, VL2 (Greenberg et al., 2009) is also a three-tier topology having a complete bipartite graph between core and aggregation switches, rather than between access and aggregation switches. Moreover, access switch traffic is forwarded through the aggregation and core switches using valiant load balancing techniques that forwards the traffic first to a randomly selected core switch and then back to the actual destination switch. The advantage of such routing is that when traffic is unpredictable, the best way to balance load across all the available network links is to forward the packets to a randomly selected core switch as an intermediate destination.
3. **PortLand:** This is also a three-tier architecture that shares the same bipartite graph feature with VL2, however at different levels (Mysore et al., 2009). It makes use of fat-tree topologies (Leiserson, 1985) and uses the concept of pods. Such pods are collections of access and aggregations switches that form complete bipartite graphs. Furthermore, each pod is connected to all the core switches, by uniformly distributing the up-links between the aggregation switches of the pod. As a result, another level of bipartite graph is formed between the pods and the core switches. Portland requires that the number of ports of all the switches is same. The number of ports per switch is the only parameter that determines the total number of pods in the topology, and consequently the total number of switches and hosts machines.
4. **BCube:** It is a multi-level network architecture for the data center defined in a recursive fashion (Guo et al., 2009). Host machines are considered as part of the network architecture and they forward packets on behalf of other host machines. It is based on the generalized hypercube architecture (Bhuyan & Agrawal, 1984) with the main difference that the neighboring hosts instead of forming a full

mesh network with each other, they connect through switches. In a BCube topology, the total number of connected hosts machines and the total number of required switches is a function of the total number of ports of each switch.

Cloud Applications and Data Center Traffic Patterns

With the increasing popularity of Cloud hosting platforms (e.g., Amazon AWS and Microsoft Azure) due to the benefits of pay-as-you-go business model, high availability and reliability, as well as extensive computing and storage services, Cloud platforms are enjoying deployment of a wide variety of composite applications, including scientific applications, social networks, video streaming, medical services, search engines and web browsing, various content delivery applications, and so on (Chen et al., 2011; Huang, Yang, Zhang, & Wu, 2012; Vaquero, Rodero-Merino, Caceres, & Lindner, 2008). Such composite applications are generally composed of multiple compute VMs backed by huge amount of data. As more and more communication-intensive applications are being deployed in data centers, the amount of inter-VM traffic is increasing with rapid pace. Based on the dynamics on computational and communicational requirements, the commonly deployed Cloud application workloads are categories into the following three groups (Kliazovich et al., 2013):

1. **Data-Intensive Workloads:** Such workloads require less computational resources, but cause heavy data transfers. For example, video file sharing where each user request generates a new video streaming process. For such applications, it is the interconnection network that can be a bottleneck rather than the computing power. In order to maintain the application performance and respect the SLAs, a continuous feedback mechanism need to be present between the network devices (e.g. switches) and the centralized workload scheduler or placement manager. Based on feedbacks, the scheduler will decide the placement of the workloads with consideration of the run-time network status and congestion levels of communication links. In this way, placement of workloads over congested network links can be avoided even though corresponding servers have enough computing capacity to accommodate the workloads. As a result, data center traffic demands can be distributed over the network in a balanced way and minimize network latency and average task completion time.
2. **Computationally Intensive Workloads:** CIWs represent the High Performance Computing (HPC) applications that are used to solve advanced and computationally expensive problems. These applications require very high amount of computing capacity, but causes little data transfer over the communication network. Such applications can be grouped together and placed in a minimum number of computing servers through VM consolidation mechanisms in order to save energy. Because of low data traffic among the VMs, there is very less probability of network congestion and most of network switches can be turned into lower power states (e.g., in sleep mode) and thus help reducing energy consumption in the data center.
3. **Balanced Workloads:** Applications that require both computing power and data transfer among the computing nodes (i.e. VMs) as represented by BWs. For example, Geographic Information Systems (GISs) need to transfer large volume of graphical data as well as huge computing resources to process these data. With this type of workloads, the average compute server load is proportional to the amount of data volume transferred over the communication

networks. VM placement and scheduling policies for such application need to account for both current state of compute servers' load and traffic loads on the network switches and links.

Since Cloud data centers host heterogeneous services and application, communication patterns exhibit wide spectrum of variations, ranging from one-to-one and all-to-all traffic matrixes. Based on trace analysis of network usage from production data centers, the following trends of network traffic are found to be pre-dominant (Ersoz, Yousif, & Das, 2007; Kandula, Sengupta, Greenberg, Patel, & Chaiken, 2009; Meng et al., 2010):

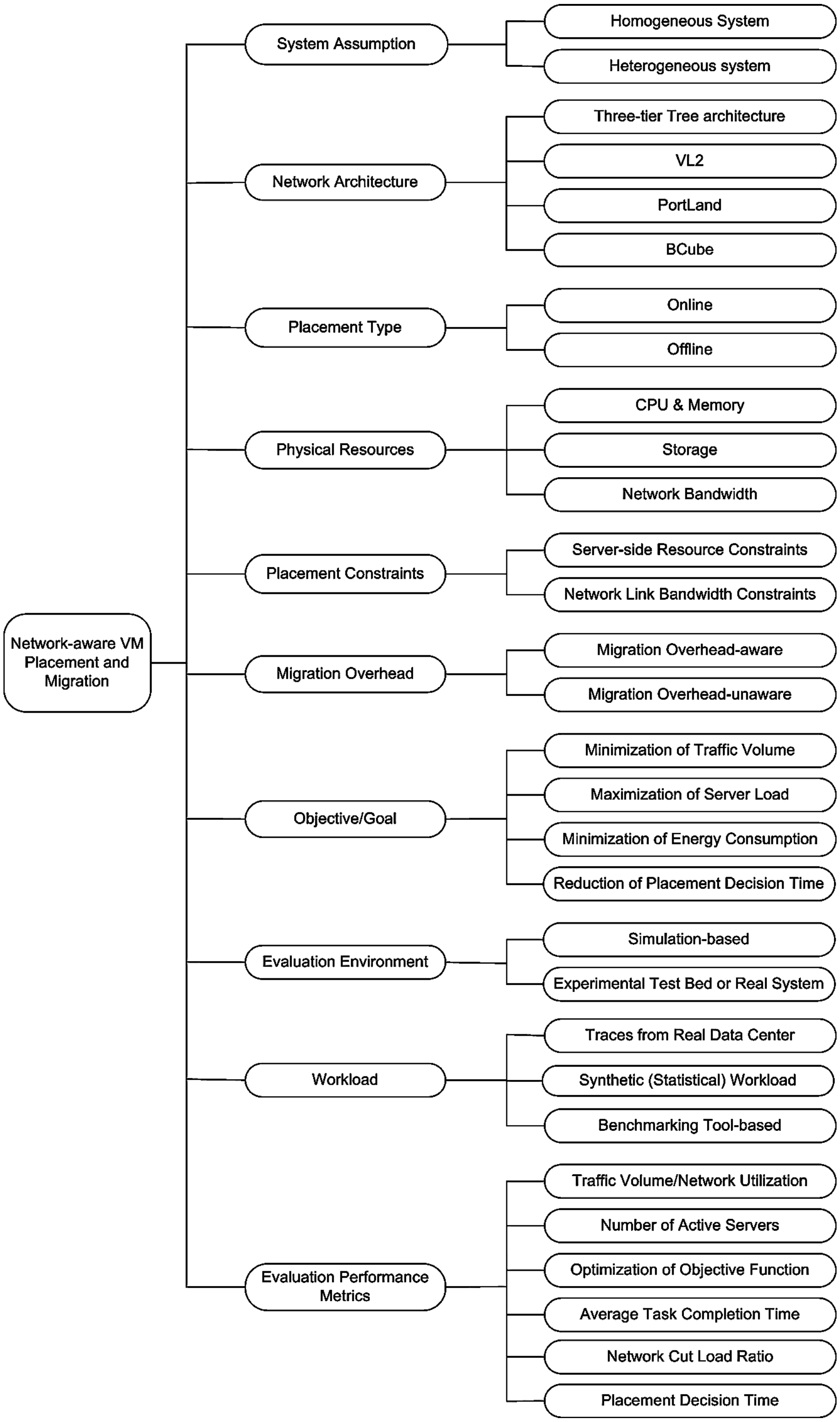
1. **Highly Non-Uniform Distribution of Traffic Volume Among VMs:** VMs running on servers exhibit uneven traffic volume among themselves across different VMs. The trace analysis reports show that 80% of the VMs have relatively low traffic rate (800Kbyte/min) over a period of two-weeks, 4% of the VMs have a rate ten times higher. This concludes that the inter-VM traffic rate varies significantly and it is quite hard for the data center administration to estimate the amount of inter-VM traffic accurately and consistently.
2. **Stable Inter-VM Traffic Volume:** For a long duration, the average inter-VM traffic rate is found to be relatively stable in spite of the highly skewed traffic rate among VMs. The work of Meng et al. (2010) shows that for the majority of the VMs, the standard deviation of their traffic rates is less than the double of the mean of the traffic rates. This consistent traffic volume among VMs implies that the run-time communication patterns among the VMs can be estimated and known a priory from the users deploying the VMs in the Clouds.
3. **Weak Correlation between Traffic Rate and Network Latency:** It is further reported from the measurement-based study that there is no any dependency or relationship between inter-VM traffic volume and the network distance between the servers hosting the VMs. That means VM pairs with high traffic rate do not necessarily correspond to low latency and vice versa.

3. TAXONOMY AND SURVEY OF THE NETWORK-AWARE VM PLACEMENT AND MIGRATION TECHNIQUES

With the various intricacies of virtualization technologies, enormous scale of modern data centers, and wide spectrum of hosted applications and services, different VM placement strategies and algorithms are proposed with various assumptions and objectives. Figure 7 presents a full taxonomy of the various aspects of network-aware VM placements and migrations. A brief description of the identified aspects of the research works used in the course of taxonomy is given below:

1. **System Assumption:** Physical servers and network resources in data centers or IT infrastructures are primarily modeled as homogeneous, and often times as heterogeneous as well. Homogeneous cluster of servers normally represent servers with same capacity for certain fixed types of resources (e.g., CPU, memory, and storage), whereas heterogeneous cluster of servers can either mean servers having different capacities of resources or different types of resources (e.g., virtualized servers powered by Xen or VMware hypervisor, and servers with Graphics Processing Units or GPUs). In practice, commercial data centers evolve over time and thus different parts of the data center can have devices with different capabilities and properties. It is quite common

Figure 7. Taxonomy of network-aware VM placement and migration


Network-aware VM Placement and Migration
System Assumption
Homogeneous System
Heterogeneous system
Network Architecture
Three-tier Tree architecture
VL2
PortLand
BCube
Placement Type
Online
Offline
Physical Resources
CPU & Memory
Storage
Network Bandwidth
Placement Constraints
Server-side Resource Constraints
Network Link Bandwidth Constraints
Migration Overhead
Migration Overhead-aware
Migration Overhead-unaware
Objective/Goal
Minimization of Traffic Volume
Maximization of Server Load
Minimization of Energy Consumption
Reduction of Placement Decision Time
Evaluation Environment
Simulation-based
Experimental Test Bed or Real System
Workload
Traces from Real Data Center
Synthetic (Statistical) Workload
Benchmarking Tool-based
Evaluation Performance Metrics
Traffic Volume/Network Utilization
Number of Active Servers
Optimization of Objective Function
Average Task Completion Time
Network Cut Load Ratio
Placement Decision Time

that a recent server installed in a data center would have much higher computing power compared to the old ones; similarly a network switch can be more recent than others and thus can have lower network latency and higher I/O throughput capacity. Moreover, recently there is growing trends towards deploying multi-purpose hardware devices that increase the degree of heterogeneity in data centers. Example of such devices can be some storage devices, such as IBM DS8000 series that have built-in compute capability (POWER5 logical portioning LPAR) that can host applications (Adra et al., 2004; Korupolu, Singh, & Bamba, 2009) and network switches, such as Cisco MDS 9000 switches ("Cisco MDS 9000 SANTap", 2014) that have additional x86 processors capable of executing applications. Efficiency and effectiveness of VM placement and migration strategies are highly dependent on the assumed system assumptions and properties. VM placement techniques that consider the heterogeneity of the devices in data centers can efficiently utilize various capabilities of the divergent resources and optimize the placements, and thus can reduce the traffic burden and energy consumption.

2. **Network Architecture/Topology:** With the variety of proposed data center network architectures and intricacies of traffic patterns, different VM placement approaches are proved to be efficient for different types of network topologies and inter-VM traffic patterns. Such effectiveness is sometimes subject to the specific analytic or modeling technique used in the proposed placement and migration schemes. Since different network topologies are designed independently focusing on different objectives (e.g., VL2 is good for effective load balancing while BCube has higher degree of connectivity and network distances among hosts), different VM placement techniques see different levels performance gains for existing network topologies. For example, the TVMPP (Traffic-aware VM Placement Problem) optimization technique (Meng et al., 2010) gains better performance for multi-layer architecture such as BCube, compared to VL2.
3. **Placement Types:** VM placement problems can be broadly categorized into two groups: online VM placement and offline VM placement. Online VM placement, including VM migrations indicate VM placement and migration actions during the run-time of the data centers where different production applications and services are active and customers are continuously requesting services (Shrivastava, Zerfos, Lee, Jamjoom, Liu, & Banerjee, 2011; Song, Huang, Zhou, & You, 2012; Takouna, Rojas-Cessa, Sachs, & Meinel, 2013; Zhang, Qian, Huang, Li, & Lu, 2012). On the other hand, offline VM placements normally indicate initial VM placements that will be actively running in subsequent phases of the system administration (Biran et al., 2012; Georgiou, Tsakalozos, & Delis, 2013; Korupolu et al., 2009; Piao & Yan, 2010; Zhang, Qian, Huang, Li, & Lu, 2012). One very important difference between online and offline VM placements is the fact that online VM placements require potential VM live migrations and large amount of extra network traffic due performing the VM migrations and can have detrimental effects on the hosted applications performance SLAs subject to the VM downtime and types of hosted services.
4. **Modeling Technique:** Effectiveness and applicability of different VM placement and migration schemes are highly contingent on the applied analytic and modeling approaches. Since different models have specific system assumptions and objectives, VM placement problems are presented using various optimizations modeling techniques, such as Quadratic Assignment Problem (QAP) (Meng et al., 2010), Convex Optimization

Problem (Huang, Gao, Song, Yang, & Zhang, 2013), Knapsack Problem (Korupolu et al., 2009), Integer Quadratic Programming (Biran et al., 2012), and so on.

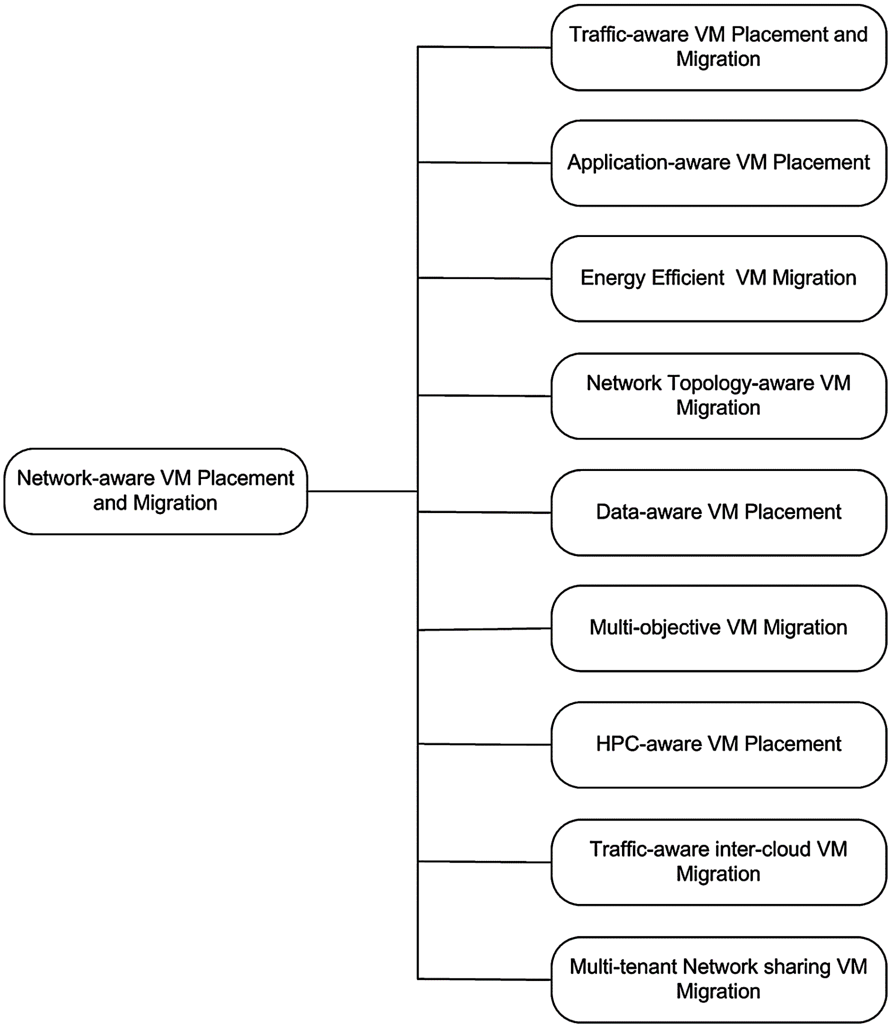
5. **Physical Resources:** Generally, optimization across different ranges of resources (i.e. CPU, memory, network I/O, storage, etc.) is harder than single resource optimization. Often various mean estimators (such as L1 norm, vector algebra, etc.) are used to compute equivalent scalar estimation while trying to optimize across multiple types of server resources. Inter-VM communication requirement is often modeled as Virtual Links (VL) that is characterized by the bandwidth demand. VM cluster forming an application environment with mutual traffic demand is represented as graph with VMs denoting vertices and VLs denoting edges of the graph.
6. **VM Placement Constraints:** Individual VM placement feasibility or practicality involves a server resource capacity constraint which means that the remaining resource (e.g., CPU cycles, memory, and storage) capacities of the hosting servers need to be enough in order to accommodate the VM. Similarly, while placing two VMs with mutual communication requirement, the bandwidth demand of the VL connecting the two VMs need to match with the remaining bandwidth capacities of the corresponding physical network links connecting the two hosting servers.
7. **Migration Overhead-Awareness:** During VM live migration process, additional network traffic is generated during the whole migration period since hypervisor need to transfer in-memory states of the running VM to the target machine. Furthermore, VM migration causes unavailability of hosted applications due to the VM downtime factor. As a consequence VM living migration is identified as an expensive data center operation that should not be triggered very often (Mann, Gupta, Dutta, Vishnoi, Bhattacharya, Poddar, & Iyer, 2012). Therefore, efficiency of a VM migration policy also dependents on the number of required VM migration commands issued. While network-aware VM migration strategies opt for optimizing overall network usage and reduce the inter-VM communication delays through migrating communicating VMs into nearly hosts, most of the strategies do not consider the associated VM migration overheads and resulting application performance degradation.
8. **Goal/Objective:** Network-aware VM placement and migration policies primarily target on minimizing overall network traffic overhead within the data center. The obvious way to achieve such goal is to place VMs with large amount of traffic communication in neighboring servers with minimum network delays and enough available bandwidth, most preferably in the same server where the VMs can communicate through memory rather than network links. With this goal in mind, VM placement and migration problem is generally modeled as mathematical optimization framework with minimization objective function. Such objective function can be a measure of total amount of network traffic transferred with the data center, or network utilization of the switches at the different tiers of the network architecture. Since VM placement and migration decision needs to be taken during run-time, reduction of the placement decision time (i.e. problem solving time or algorithm execution time) is also considered as an objective.
9. **Algorithm/Solution Approach:** Given the above mentioned placement constraints, VM placement problem is in fact an NP-complete problem since it requires combinatorial optimization to achieve the goals. As a consequence, most of the research works attempt to solve the problem through heuristic methods so that the algorithms terminate in a reasonable amount

of time. Such heuristics are not guaranteed to produce optimal placement decision; however from time constraints perspective exhaustive search methods that guarantee the generation of optimal solutions are not practical, especially considering the scale of modern data centers. Several metaheuristic-based approaches such as Ant Colony Optimization (ACO), Genetic Algorithms (GA), and Simulated Annealing (SA) have been proven to be effective in the area of VM consolidation. Nevertheless, adaptation and utilization of these problem solving techniques are still open to explore to address the network-aware VM placement and migration problem.

10. **Evaluation/Experimental Platform:** Most of the proposed works presentation evaluation based on simulation based experimentation. This, however, makes sense given the complexity and scale of modern data centers and the hosted applications. Several works have attempted to validate their proposed placement policies through testbed-based experiments and have reported various run-time dynamics across different performance metrics, that is otherwise would be impossible to report though simulation-based evaluations. However, such evaluations are performed on small scale testbeds with 10 to 20 physical machines (or PMs) and thus do not necessarily forecast the potential behavior and performance for large scale data centers.
11. **Competitor Approaches:** Comparison of the performance results among the various competitor placement approaches highly depends on the goals of the competitor approaches. Since network-aware VM placement is a relatively new area of research, proposed approaches are often compared to other placement approaches that are agnostic to network traffic and network topologies and have different goals set in the underlying algorithms (e.g., power consumption minimization or SLA violation reduction).
12. **Workload/VM Cluster:** Because of the lack of enough VM workload data sets from large scale Cloud data centers or other production data centers due to their proprietary nature, statistical distribution-based VM load (compute resource and network bandwidth demands) generation is the most common approach adopted in the simulation-based evaluations. Among others, normal, uniform, and exponential distributions are usually used most. Such synthetic workload data characterize randomness based on particular trend (e.g., through setting mean and variance in case of normal distribution). Subject to accessibility, workload traces from real data centers of often used to feed data to the simulation based evaluation to imply the effectiveness of the proposed approaches in real workload data. Furthermore, testbed-based evaluations often use various benchmarking tools to generate and feed runtime workload data to the algorithms under evaluation.
13. **Evaluation Performance Metrics:** Depending of the goals of the VM placement solutions, various performance metrics are reported in proposed research works. Most common performance metric used is the overall network traffic in the data center. Placement schemes that have multiple objectives, often try to balance between network performance gain and energy consumption reduction, and report evaluations based on both traffic volume reduction and number of active servers. From energy savings point of view, minimization of the number of active servers in data center through VM consolidation is always an attractive choice.

Figure 8 provides a categorization of the various published research works based on the addressed and analyzed subareas of the VM placement problem and the ultimate objectives of the VM placement and migration strategies.

Figure 8. Categorization of network-aware VM placement and migration approaches


Network-aware VM Placement and Migration
Traffic-aware VM Placement and Migration
Application-aware VM Placement
Energy Efficient VM Migration
Network Topology-aware VM Migration
Data-aware VM Placement
Multi-objective VM Migration
HPC-aware VM Placement
Traffic-aware inter-cloud VM Migration
Multi-tenant Network sharing VM Migration

The next four subsections are dedicated to thorough review, analysis, and remarks on the recent prominent research works.

Traffic-Aware VM Placement and Migration Techniques

Network Topology-Aware VM Cluster Placement in IaaS Clouds

Georgiou et al. (2013) have investigated the benefits of user-provided hints regarding inter-VM communication patterns and bandwidth demands during the actual VM placement decisions phase. The authors have proposed two offline VM-cluster placement algorithms with the objective to minimize the network utilization at physical layer, provided that the physical server resource capacity constraints are met. VM deployment request is modeled as Virtual Infrastructure (VI) with specification of the number and resource configuration (CPU core, memory, and storage demands) of VMs, bandwidth demands of inter-VM communication within the VI, modeled as Virtual Links (VLs), as well as possible anti-colocation constraint for pairs of VMs. The underlying physical infrastructure is modeled as a homogeneous cluster of servers organized according to the PortLand (Mysore et al., 2009) network architecture. The authors have argued that conventional tree-like network topologies often suffer from over-subscription and network resource contention primarily at the core top-levels, leading to bottlenecks and delays in services, PortLand network architecture can play a significant role in effective management of computational resources and network bandwidth in Cloud data centers.

The authors have also presented a framework comprising of two layers: physical infrastructure consisting of homogeneous servers organized as PortLand network topology and a middleware on top of the infrastructure. The middleware layer is composed of the following two main components: Planner and Deployer. As input, the Planner gets VM deployment request as VI specification (in XML format), and possible suggestions regarding desired features in VI from user as well as the current resource state information of the infrastructure layer, executes the VM placement algorithms to determine the VM-to-PM and VL-to-physical link mappings, and finally passes over the placement decision to the Deployer. The Deployer can be a third-party provided component that takes care of the VMs deployment on the physical layer components.

With the goal of minimizing network utilization of the physical layer during the VI deployment decision, the authors have proposed two algorithms based on greedy approach. The first algorithm, Virtual Infrastructure Opportunistic fit (VIO) tries to place the communicating VMs near to each other in the physical network. Starting with a sorted list of VLs (in decreasing order of their bandwidth demands) connecting the VMs, the VIO picks up the front VL from the list and attempts to place the VMs connected by the VL in the nearest possible physical nodes (preferably in the same node when anti-colocation is not set), provided that physical node resource capacity constraints, network link bandwidth capacity constraints, as well as user provided constraints are met. In case VIO reaches a dead-end state where the VL at hand cannot be placed on any physical link, VIO employs a backtracking process where VLs and corresponding VMs are reverted back to unassigned state. Such VL placement inability can occur due to three reasons:

1. No physical node with enough resource is found to host a VM of the VL,
2. No physical path with enough bandwidth is found to be allocated for the VL, and
3. Anti-colocation constraint is violated.

Backtracking process involves de-allocation of both server resource and network bandwidth of physical links. In order to limit the number of reverts for a VL and terminate the algorithm with a reasonable amount of time, a revert counter is set for each VL. When the maximum amount of reverts has been reached for a VL, the VI placement request is rejected and the VIO terminates gracefully. The second algorithm, Vicinity-BasEd Search (VIBES) based on the PortLand network architecture characteristics, tries to detect an appropriate PortLand neighborhood to accommodate all the VMs and VLs of the requested VI, and afterward applies VIO within this neighborhood. In order to identify fitting neighborhood, VIBES exploits PortLand's architectural feature of pods (cluster of physical nodes under the same edge-level switch). The authors also presented formula for ranking all neighborhoods based on the available resources in the servers and bandwidth of the physical links within each neighborhood. VIBES starts with the pod with the most available resources and invokes VIO. Upon rejection from VIO, VIBES expands the neighborhood further by progressively merging the next most available pod to the set of already selected pods. The search for a large enough neighborhood proceeds until a neighborhood with enough available resources is found or the search window is growing beyond a customizable maximize size in which case the VI placement request is rejected.

Performance evaluation of VIO and VIBES is conducted through simulation of physical infrastructures and compared against network-agnostic First Fit Decreasing (FFD) algorithm. Online VI deployment and removal is simulated using three different data flow topologies: Pipeline, Data Aggregation, and Epigenomics (Bharathi et al., 2008). The simulation results show that the proposed algorithms outperforms FFD with respect to network usage: VIO trims down the network traffic routed through the top-layer core switches in the PortLand architecture by up to 75% and incorporation of VIBES attains a further 20% improvement. The authors have also suggested future research directions such as optimization of the power usage of network switches through exploitation of reduced network utilization, testing VIO and VIBES for other network topologies such as BCube (Guo et al., 2009) and VL2 (Greenberg et al., 2009).

Stable Network-Aware VM Placement for Cloud Systems

With focus on communication pattern and dynamic traffic variations of modern Cloud applications, as well as non-trivial data center network topologies, Biran et al. (2012) have addressed the problem of VM placement with the objective to minimize the maximum ratio of bandwidth demand and capacity across all network cuts and thus maximize unused capacity of network links to accommodate sudden traffic bursts. The authors have identified several important observations regarding network traffic and architectures:

1. Due to several factors such as time-of-day effects and periodic service load spikes, runtime traffic patterns undergo high degree of variations,
2. Modern data centers are architected following non-trivial topologies (e.g., Fat-tree (Al-Fares, Loukissas, & Vahdat, 2008) and VL2 (Greenberg et al., 2009)) and employ various adaptations of dynamic multi-path routing protocols.

Considering the above mentioned points, the authors presented two VM placement algorithms that strive to satisfy the forecasted communication requirements as well as be resistant to dynamic traffic variations.

The authors have introduced the Min Cut Ratio-aware VM Placement (MCRVMP) problem and formally formulated using the Integer Quadratic Programming model considering both the server side resource capacity constraints and network resource constraints evolving from complex network topologies and dynamic routing schemas. Since the MCRVMP problem definition works only for tree topology, the authors have also proposed graph transformation techniques so that MCRVMP can be applied to other complex network topologies, for example VL2 and Fat-tree. Considering the fact the MCRVMP is a NP-hard problem, the authors have proposed two separate heuristic algorithms for solving the placement problem and compared these against optimal and random placements.

Both the proposed VM placement heuristic algorithms utilize the concept of Connected Components (CCs) of the running VMs in the data center. Such a CC is formed by the VMs that exchange data only between themselves or with the external gateway (e.g., VMs comprising a multi-tier application) and thus clustering VMs in this way helps minimize the complexity of the problem. First algorithm, termed 2-Phase Connected Component-based Recursive Split (2PCCRS) is a recursive, integer programming technique-based algorithm that utilizes the tree network topology to define and solve small problem instances on one-level trees. By adopting a two-phase approach, 2PCCRS places the CCs in the network and then expands them to place the actual VMs on the servers. Thus, 2PCCRS reduces the larger MCRVMP problem into smaller sub-problems and solves them using mixed integer programming solver in both the phases. Second algorithm, called Greedy Heuristic (GH) entirely avoids using mathematical programming and greedy places each VM individually. Similar to 2PCCRS, GH works in two phases. In the first phase, GH sorts all the traffic demands in decreasing values and sorts all CCs in decreasing order based on the accumulated traffic demands among the VMs within a CC. In the second phase, GH iteratively processes the ordered traffic demands by placing each VM on the physical server that results in minimum value of the maximum cut load values.

The efficiency of the proposed algorithms is evaluated in two phases. In the first phase, 2PCCRS and GH algorithms were compared to random and optimal placement approaches with focus on placement quality in terms of worst and average cut load ratio and solution computation time. As reported by the authors, for small problem instances both 2PCCRS and GH reach worst case and average cut load ratio very close to optimal algorithm with nearly zero solving time; whereas for larger problem sizes, 2PCCRS significantly outperforms GH, while requiring much higher solving time due its use of mathematical programming techniques. In the second phase, the authors have validated the resilience of MCRVMP-based placements under time-varying traffic demands with NS2-based simulations focusing on the percentage of dropped packets and average packet delivery delay. Simulation results show that with no dropped packets, both 2PCCRS and GH can absorb traffic demands up to three times the nominal values. Furthermore, placements produced by the 2PCCRS algorithm have average packet delivery delays lower than GH-based ones due to the less loaded network cuts.

The authors have also remarked that the proposed MCRVMP problem formulation is not meant for online VM placement where new VM requests are served for data center having already placed VMs. In addition, the authors have ignored the potential VM migration costs entirely.

As per future works, the authors have indicated potential extension of MCRVMP by incorporating traffic demand correlation among VMs to further cut down the amount of dropped packets and by preventing MCRCMP to produce solutions with very high local compute-resource overhead due to inter-memory communications.

Scalability Improvement of Data Center Networks with Traffic-Aware VM Placement

Meng et al. (2010) have addressed the scalability problem of modern data center networks and proposed solution approaches through optimization of VM placement on physical servers. Different from existing solutions that suggest changing of network architecture and routing protocols, the authors have argued that scalability of network infrastructures can be improved by reducing the network distance of communicating VMs. In order to observe the dominant trend of data center traffic-patterns, the authors have claimed to have conducted a measurement study in operational data centers resulting with the following insights:

1. There exists low correlation between average pairwise traffic rate and the end-to-end communication cost,
2. Highly uneven traffic distribution for individual VMs, and
3. VM pairs with relatively heavier traffic rate tend to constantly exhibit the higher rate and VM pairs with low traffic rate tend to exhibit the low rate.

The authors have formally defined the Traffic-aware VM Placement Problem (TVMPP) as a combinatorial optimization problem belonging to the family of Quadratic Assignment Problems (Loiola, de Abreu, Boaventura-Netto, Hahn, & Querido, 2007) and proved its computational complexity to be NP-hard. TVMPP takes the traffic matrix among VMs and communication cost matrix among physical servers as input, and its optimal solution would produce VM-to-PM mappings that would result in minimum aggregate traffic rates at each network switch. The cost between any two communicating VMs is defined as the number of switches or hops on the routing path of the VM pair. The authors have also introduced a concept of slot to refer to one CPU/memory allocation on physical server where multiple such slots can reside on the same server and each slot can be allocated to any VM.

Since TVMPP is NP-hard and existing exact solutions cannot scale to the size of current data centers, the authors have proposed two-tier approximate algorithm Cluster-and-Cut based on two design principles:

1. Finding solution of TVMPP is equivalent to finding VM-to-PM mappings such that VM pairs with high mutual traffic are placed on PM pairs with low-cost physical links and
2. Application of the divide-and-conquer strategy.

The Cluster-and-Cut heuristic is composed of two major components: SlotClustering and VMMinKcut. SlotClustering partitions a total of *n* slots in the data center into *k* clusters using the cost between slots as the partition criterion. This component produces a set of slot-clusters sorted in decreasing order of their total outgoing and incoming cost. The VMMinKcut partitions a total of *n* VMs into *k* VM-clusters such that VM pairs with high mutual traffic rate are placed within the same VM-cluster and inter-cluster traffic is minimized. This component uses the minimum k-cut graph algorithm (Saran & Vazirani, 1995) partition method and produces *k* clusters with the same set of size as the previous *k* slot-clusters. Afterwards, Cluster-and-Cut maps each VM-cluster to a slot-cluster and for each VM-cluster and slot-cluster pair, it maps VMs to slots by solving the much smaller sized TVMPP problem. Furthermore, the authors have shown that the computational complexity of SlotClustering and VMMinKcut are $O(nk)$ and $O(n^4)$, respectively, with total complexity of $O(n^4)$.

The performances evaluation of Cluster-and-Cut heuristic is performed through trace-driven simulation using hybrid traffic model on inter-VM traffic rates (aggregated incoming and outgoing) collected from production data centers. The results show that Cluster-and-Cut produces solution with objective function value 10% lower than its competitors across different network topologies and the solution computation time is halved.

However, the proposed approach considers some assumptions that cannot be hold in the context of real data centers. TVMPP does not incorporate the link capacity constraints that can lead to VM placement decisions with congested links into the data center (Biran et al., 2012). Furthermore, Cluster-and-Cut algorithm places only one VM per server that can result in high amount of resource wastage. Additionally, it is assumed that static layer 2 and 3 routing protocols are deployed in the data center. Finally, VM migration overhead incurred due to the offline VM shuffling is not considered.

Through discussion the authors have indicated the potential benefit of combining the goals of both network resource optimization and server resource optimization (such as power consumption or CPU utilization) during the VM placement decision phase. They also emphasized that reduction of total energy consumption in a data center requires combined optimization of the above mentioned resources. The authors have also mentioned potential of performance improvement by employing dynamic routing and VM migration, rather than using simple static routing.

Network-Aware Energy-Efficient VM Placement and Migration Approaches

Multi-Objective Virtual Machine Migration in Virtualized Data Center Environments

Huang et al. (2013) have addressed the problem of overloaded VM migration in data centers having inter-VM communication dependencies. Indicating the fact that most of the existing works on VM migrations focus primarily on the server-side resource constraints with the goal of consolidating VMs on minimum number of servers and thus improving overall resource utilization and reducing energy-consumption, the authors have argued that VMs of modern applications have mutual communication dependencies and traffic patterns. As a result, online VM migration strategies need to be multi-objective focusing both on maximizing resource utilization and minimizing data center traffic overhead.

Following a similar approach as in (Huang et al., 2012), the authors have presented three stages of the joint optimization framework:

1. Based on the dominant resource share and max-min fairness model, the first optimization framework tries to maximize the total utilities of the physical servers; in order words tries to minimize the number of used servers and thus reduce power consumption,
2. Considering the complete application context with inter-VM traffic dependencies, the second optimization framework strives to minimize the total communication costs among VM after necessary VM migrations, and
3. Based on the above two frameworks, the third optimization framework combines the above goals subject to the constraints that the allocated resources from each server is not exceeded its capacity and the aggregated communication weight of a server is lower or equal to its bandwidth capacity.

The authors have further proposed a two-stage greedy heuristic algorithm to solve the defined optimization problem: Base Algorithm and Extension Algorithm. The Base Algorithm takes as input the set of VMs, set of servers, and the dominant resource share of user servers, and the set of overloaded VMs. Then, it sorts the overloaded VMs in decreasing order of their dominant resource share before migration. After incorporation of application dependencies (i.e. inter-VM communication dependencies), the Extension Algorithm selects candidate destination server for migration to the server with the minimum dominant resource share and application-dependent inter-VM traffic. The VM migration effect is computed as the impact based on both distance effect and inter-VM traffic pattern-based network cost after migration. For each overloaded VM, the total communication weight is computed as the sum of all related inter-VM communication weights and the overloaded VM is migrated to the server with minimum migration impact.

The authors have shown simulation-based evaluation of the proposed multi-objective VM placement approach with comparison to AppAware (Shrivastava et al., 2011) application-aware VM migration policy. The following four different network topologies are used as data center network architecture: Tree, Fat-Tree (Al-Fares et al., 2008), VL2 (Greenberg et al., 2009), and BCube (Guo et al., 2009). Data center server capacity, VM resource demand, and inter-VM traffic volume is generated synthetically based on normal distribution with varying mean. The results show that the achieved mean reduction in traffic of the proposed algorithm is higher for BCube compared to Tree topology. Compared to AppAware, the proposed algorithm can achieve larger reduction in data center network traffic volume, by generating migrations that decreased traffic volume transported by the network up to 82.6% (for small number of VMs). As per average impact of migration, it decreases with the increase of server resource capacity. It is attributed that since the multiplier factor in the migration impact formulation includes dominant resource share of the migrating VM and it is decreased after migration. However, with the increase of VM resource demands, the average impact of migration is increased. This is attributed for the fact that the demand of VMs has a direct impact on the inter-dependencies among the VMs of multi-tier applications. Finally, with the increase of inter-VM communication weights, the average impact of migration increases since communication weights influence the cross-traffic burden between network switches.

Communication Traffic Minimization with Power-Aware VM Placement in Data Centers

Zhang et al. (2012) have addressed the problem of static greedy allocations of resources to VMs, regardless of the footprints of resource usage of both VMs and PMs. The authors have suggested that VMs with high communication traffic can be consolidated into minimum number of servers so as to reduce the external traffic of the host since co-located VMs can communicate using memory copy. With goal of minimizing communication traffic within a data center, the authors have defined dynamic VM placement as an optimization problem. The solution of the problem would be a mapping between VMs and servers, and such a problem is presented to be reduced from a minimum k-cut problem (Xu & Wunsch, 2005) that is already proved to be NP-hard. Since an idle server uses more than two-third of the total power when the machine is fully utilized (Kusic et al., 2009), the authors set power-consumption minimization as a second objective of their proposed VM placement scheme.

The authors have provided formal presentation of the optimization problem using mathematical framework that is set to minimize the total communication traffic in the data center, provided that various server-side resource constraints should be satisfied. Such problem can be solved by partitioning the VMs into clusters in such a way that VMs with heavy communication can be placed in the same server. As a solution, the author proposed the use of K-means clustering algorithm (Xu & Wunsch, 2005) that would generate VM-to-server placement mappings. Utilizing the K-means clustering approach, the authors proposed a greedy heuristic named K-means Clustering for VM consolidation that starts by considering each server as a cluster. Such cluster definition has got some benefits:

1. The number *K* and the initial clusters can be fixed to minimize the negative impact from randomization,
2. There is an upper-bound for each cluster that corresponds to the capacity constraints of each server, and
3. Fixed clusters can reduce the number of migrations.

In each iteration of the K-means Clustering for the VM Consolidation algorithm, the distance between a selected VM and a server is determined. Using this, the VM is placed in the server with minimum distance. This step is repeated until every VM has a fixed placement on its destination server. The authors have further reported that the greedy algorithm has a polynomial complexity of $O(tmn)$, where t is the number of iterations, n is the number of VMs, and m is the number of servers in the data centers. The authors have further presented algorithms for computing the distance between a VM and a cluster, and for online scenarios where greedy heuristic handles new VM requests.

Performance evaluation based on simulation and synthetic data center load characteristics is reported with superior performance gain by the proposed algorithm compared to its three competitors:

1. Random placement,
2. Simple greedy approach (puts the VM on the server which communicates most with current VM), and
3. First Fit (FF) heuristic.

Both the random placement and FF heuristics are unaware of inter-VM communication. The results show that the proposed greedy algorithm achieved better performance for both performance metrics:

1. Total communication traffic in data center, and
2. Number of used server (in other words, measure of power cost) after consolidation.

For the online VM deployment scenario, the clustering algorithm is compared against greedy algorithm and it is reported that the greedy algorithm can perform very close to the clustering method where the number of migrations is significantly larger than the greedy method and the greedy method can deploy new VM requests rapidly without affecting other nodes.

As for future work directions, the authors expressed plan to introduce the SLA to approach a better solution where the data center can provide better performance for the applications because of less communication traffic. This metric would be included in the cost model. Furthermore, the migration cost would be taken as a metric of the proposed distance model.

Energy-Aware Virtual Machine Placement in Data Centers

Huang et al. (2012) have presented a joint physical server and network device energy consumption problem for modern data centers hosting communication-intensive applications. The authors have staged several data center facts in order to signify the importance of multi-objective VM placement:

1. Increasing deployment of wide spectrum of composite applications consisting of multiple VMs with large amount of inter-VM data transfers,
2. Continuous growth in the size of data centers,
3. Existing VM placement strategies lack multiple optimizations, and
4. Rise of electricity cost.

In response to the above issues, the authors have investigated the balance between server energy consumption and energy consumption of data center transmission and switching network.

The multi-objective VM placement problem is modeled as an optimization problem in three stages. Considering server resource capacities (CPU, memory, and storage) and VM resource demands, the first optimization framework is targeted on VM placement decisions that would maximize server resource utilizations and eventually reduce energy consumption (by turning idle servers to lower power state, e.g., standby) following proportional fairness and without considering inter-VM communication pattern. The second optimization framework considers inter-VM data traffic patterns and server-side bandwidth capacity constraints, and is modeled as a Convex Programming Problem that tries to minimize the total aggregated communication costs among VMs. Finally, the energy-aware joint VM placement problem is modeled using fuzzy-logic system with trade-off between the first two objectives that can be in conflict when combined together. The authors have further proposed a prototype implementation approach for the joint VM placement following a two-level control architecture with local controllers installed in every VM and a global controller at the data center level responsible to determining VM placement and resource allocations.

As solution approach, the authors have put forward two algorithmic steps: VMGrouping and SlotGrouping. VMGrouping finds VM-to-server mappings such that VM pairs with high traffic communication are mapped to server pairs with low cost physical link. Such VM-to-server mappings are modeled as Balanced Minimum K-cut Problem (Saran & Vazirani, 1995) and a k-cut with minimum weight is identified so that the VMs can be partitioned into *k* disjoint subsets of different sizes. Afterwards, SlotGrouping maps each VM group to appropriate servers in closest neighborhood respecting the server side resource constraints.

The authors have validated the proposed multi-objective VM placement approach using simulation-based evaluation under varying traffic demands, and load characteristics of VMs and physical servers using normal distribution under different means as well as for different network architectures (e.g., Tree (Al-Fares et al., 2008), VL2 (Greenberg et al., 2009), Fat-tree (Guo et al., 2008), and BCube (Guo et al., 2009)). Focusing on the formulated objective function value and total data center traffic volume as performance metrics, the proposed joint VM placement policy is compared against random placement and First Fit Decreasing (FFD) heuristic-based placement policies. The results show that the joint VM placement achieves higher objective values and much reduced traffic flow (up to 50% to 81%) compared to other approaches, resulting in lower communication cost and resource wastage. In order to assess performance from energy-consumption reduction point of view, the proposed placement approach is compared against Grouping Genetic Algorithm (GGA) (Agrawal, Bose, & Sundarrajan, 2009), FFD, two-stage heuristic algorithm (Gupta, Bose, Sundarrajan, Chebiyam, & Chakrabarti, 2008), random placement, and optimal placement considering the number of used PMs as performance metric. It is reported that the proposed energy-aware joint placement method achieves better performance over random placement, GGA, and the two stage heuristic algorithm, and inferior performance over FFD and optimal placement. Such performance pattern is rationalized by the trade-offs between multiple objectives (i.e. minimizing both resource wastage and traffic volume simultaneously) that the joint VM placement policy strives to achieve.

In this research work, the authors have brought about a very timely issue of balancing both energy- and network-awareness while VM placement decisions are made. Most of the existing works focus on either one of the objectives, not both at the same time. However, this work has not considered the impact of the necessary VM live migrations and reconfigurations on both the network links and hosted applications performance, which can have substantially detrimental effects on both applications SLAs and network performance given that the new VM placement decision requires large number of VM migrations.

Network- and Data-Aware VM Placement and Migration Mechanisms

Coupled Placement in Modern Data Centers

Korupolu et al. (2009) have addressed the problem of placing both computation and data components of applications among the physical compute and storage nodes of modern virtualized data centers. The authors have presented several aspects that introduce heterogeneity in modern data centers and thus make the optimization problem of compute-data pairwise placement non-trivial:

1. Enterprise data centers evolve over time and different parts of the data center can have performance variations (e.g., one network switch can be more recent than others and have lower latency and greater I/O throughput),
2. Wide spread use of multi-purpose hardwire devices (e.g., storage devices with built-in compute resources), and
3. Large variance of the I/O rates between compute and data components of modern applications.

Taking into considering the above factors, the Coupled Placement Problem (CPP) is formally defined as an optimization problem with the goal of minimizing the total cost over all applications, provided that compute server and storage node capacity constraints are satisfied. The cost function can be any user defined function and the idea behind it is that it captures the network cost that is incurred due placing the application computation component (e.g., VM) in a certain compute node and the data component (e.g., data block or file system) in a certain storage node. One obvious cost function can be the I/O rate between compute and data components of application multiplied by the corresponding network distance between the compute and storage nodes.

After proving the CPP as a NP-hard problem, the authors proposed three different heuristic algorithms to solve it:

1. Individual Greedy Placement (INDV-GR), following greedy approach, tries to place the application data storages sorted by their I/O rate per unit of data storage where storage nodes are ordered by the minimum distances to any connected compute node. Thus, INDV-GR algorithm places highest throughput applications on storage nodes having the closest compute nodes.
2. Another greedy algorithm, Pairwise Greedy Placement (PAIR-GR) considers the compute-storage node affinities and tries to place both compute and data components of each application simultaneously by assigning applications sorted by their I/O rate normalized by their CPU and data storage requirements on storage-compute node pairs sorted by the network distance between the node pairs.
3. Finally, in order to avoid early sub-optimal placement decisions resulting due to the greedy nature of the first two algorithms, the authors proposed Coupled Placement Algorithm (CPA) where CPP is shown to have properties very similar to the Knapsack Problem (Pisinger, 1997) and the Stable-Marriage Problem (McVitie & Wilson, 1971). Solving both the Knapsack and the Stable-Marriage Problem, the CPA algorithm iteratively refines placement decisions to solve the CPP problem in three phases:
    a. CPA-Stg phase where data storage placement decision is made,
    b. CPA-Compute phase where computation component placement decision is taken provided the current storage placements, and
    c. CPA-Swap phase that looks for pairs of applications for which swapping their storage-compute node pairs improves the cost function and performs the swap.

The performance of INDV-GR, PAIR-GR, and CPA is compared against the optimal solutions through simulation-based experimentations. The authors have used CPLEX ILP solver for small problem instances and MINOS solver based on LP-relaxation for larger problems. Cost function values and placement computation times are considered as performance metrics and the experiments are carried out across four different dimensions:

1. Problem size/complexity through variations in simulated data center size,
2. Tightness of fit through variations of mean application compute and data demands,
3. Variance of application compute and data demands, and
4. Physical network link distance factor.

Through elaborate analysis of results and discussion, the proposed CPA algorithm is demonstrated to be scalable both in optimization quality and placement computation time, as well as robust with varying workload characteristics. On average, CPA is shown to produce placements within 4% of the optimal lower bounds obtained by LP formulations.

However, the optimization framework takes some simplistic view of the application models and resource capacity constraints. Firstly, the CPP has considered each application as having one compute and one data storage components whereas modern applications usually have composite view with multiple compute components with communications among themselves as well as communication with multiple data storage components. Secondly, on the part of compute resource demand, only CPU is considered whereas memory and other OS-dependent features make the problem multi-dimensional (Ferdaus, Murshed, Calheiros, & Buyya, 2014). Thirdly, no assumption is made regarding the overhead or cost of reconfiguration due to the new placement decision, in which VM migrations and data movement would be dominating factors. Finally, no network link bandwidth capacity constraint is no taken into account during the CPP formulation.

Nonetheless, the authors have pointed out couple of future research outlooks: inclusion of multi-dimensional resource capacity constraints and other cost models focusing on different data center objectives like and energy utilization.

Network and Data Location-Aware VM Placement and Migration Approach in Cloud Computing

Piao et al. (2010) have addressed the problem of achieving and maintaining expected performance level of data-intensive Cloud applications that need frequent data transmission from storage blocks. The studied scenario is focused on modern Cloud data centers comprising of both compute Clouds (e.g., Amazon EC2) and storage Clouds (e.g., Amazon S3) where hosted applications access the associated data across the Internet or Intranet over communication links that can either be physical or logical. Moreover, the authors have suggested that under current VM allocation policy, the data can be stored arbitrarily and distributed across single storage Cloud or even over several storage Clouds. Furthermore, the brokers allocate the applications without consideration of the data access time. As a consequence, such placement decisions can lead to data access over unnecessary distance.

In order to overcome the above mentioned problem, the authors have proposed two algorithms based on exhaustive search: VM placement approach and VM migration approach. For both the solutions, the per application data is modeled as a set of data blocks distributed across different physical storage nodes with varying distances (either logical or physical) from physical compute nodes. Network speed between physical compute node and storage node is modeled using *Speed(s, Δt)* function that depends on the size of the data *s* and packet transfer time slot *Δt*. Finally, for each physical compute node, the corresponding data access time is formulated as the sum of product of each data block size and the inverse of the corresponding network speed value. The VM placement algorithm handles each new application deployment request and performs an exhaustive search over all the feasible compute nodes to find the one with minimum data access time for the corresponding data blocks for the submitted VM, subject to the compute node resource capacity constraints are satisfied. The VM migration algorithm is triggered when the application execution time exceeds the SLA specified threshold. In such a situation, a similar exhaustive search over all the feasible compute nodes is performed to find the one with minimum data access time for the corresponding data blocks for the migrating VM, subject to the compute node resource capacity constraints as satisfied.

The efficacy of the proposed algorithms is validated through simulation based on the CloudSim (Buyya, Ranjan, & Calheiros, 2009) simulation toolkit. The evaluation is focused on the average task completion time and the proposed algorithms are compared against the default VM placement policy implemented in CloudSim 2.0, namely VMAllocationPolicySimple that allocates the VM on the least utilized host following a load-balancing approach. The simulation is setup with small scale data centers comprising of 3 VMs, 3 data blocks, 2 storage nodes, and 3 compute nodes with fixed resource capacities. It is shown that the proposed approaches needed shorter average task completion time, which is emphasized as due to the optimized location of hosted VMs. In order to trigger the proposed VM migration algorithm, the network status matrix is changed and as a consequence some of the VMs are migrated to hosts that resulted in lower average task completion time.

Besides considering very simplistic view of federated Cloud data centers, the proposed VM placement and migration algorithms take an exhaustive search approach that may not scale for very large data centers. Moreover, the experimental evaluation is performed in a tiny scale and compared with a VM placement that is fully network-agnostic. Furthermore, VM migration or reconfiguration overhead is not considered in the problem formulation or solution schemes.

As for future work directions, the authors suggested inclusion of negotiation between service provider and user in terms of data access time to guarantee SLA enforcement. In order to avoid some users' tasks always occupying a faster network link, priority-based scheduling policy is recommended through extension of the payment mechanisms.

Application-Aware VM Placement and Migration Strategies

Communication-Aware Scheduling for Parallel Applications in Virtualized Data Centers

Takouna et al. (2013) have introduced the problem of scheduling VMs that are part of HPC applications and communicate through shared memory bus (when placed in the same server) and shared networks (when placed in different servers). The authors have identified some limitations of existing VM placement and migration approaches with regards to the HPC and parallel applications:

1. VM placement approaches that optimize server-side resources (e.g., CPU and memory) are unaware of the inter-VM communication patterns, and as a result are less efficient from network utilization and ultimately from application performance point of view, and
2. Recent network-aware VM placement approaches focus on optimal initial VM placement and overlook the real-time communication patterns and traffic demands, and thus are not reactive to changes.

In order to address the above shortcomings, the authors have proposed communication-aware and energy-efficient VM scheduling technique focusing on parallel applications that use different programming models for inter-VM communication (e.g. OpenMP and Message Passing Interface (MPI)). The proposed technique determines the run-time inter-VM bandwidth requirements and communication patterns and upon detection of inefficient placement, reschedules the VM placement through VM live migrations.

In order to handle potential VM migration requests, the authors have presented a brief overview of the system framework consisting of VMs with peer-VM information (i.e. VMs that have mutual communication) and a central Migration Manager (MM). HPC jobs are executed in individual VMs and each VM have a list of its peer-VMs at run-time. It is the responsibility of the MM to determine the communication pattern of the whole parallel application. It is further assumed that each physical server have enough free resources (10% to 20% of CPU) to handle potential VM migration. The authors have further proposed an iterative greedy algorithm, namely Peer VMs Aggregation (PVA) that would be run by the MM upon getting migration requests from VMs. The ultimate goal of the PVA algorithm is to aggregate the communicating VMs with mutual traffic into the same server so that they can communicate through the shared memory bus, so as to reduce the inter-VM traffic flow in the network. This would both localize the traffic (and thus reduce network utilization) and minimize the communication delays among VMs with mutual communication dependencies (and thus improving application performance). The PVA algorithm is composed of the following four parts:

1. **Sort:** The MM ranks the VMs that are requesting migration in a decreasing order based on the number of input/output traffic flows while ignoring the requests of VMs assigned on the same server),
2. **Select:** MM selects the highest ranked VM to be migrated to the destination server where its peer VMs are assigned,

3. **Check:** MM examines the feasibility of VM migrations to the destination servers in terms of server resource (CPU, memory, and network I/O) capacity constraints, and
4. **Migrate:** If MM finds the server suitable for the migrating VM, it directly migrates the selected VM to that server; otherwise the MM tries to migrate a VM from the destination server to free enough resources for the selected VM to be placed in the same server of its peer VMs (in that case the selected VM should also be suitable to be migrated). However, if the destination server does not host any VM, the MM can assign the selected VM on a server that shares the same edge switch with the server of its peer VMs.

The PVA approach is reported to minimize the total data center traffic significantly by reducing the network utilization traffic by 25%. The authors have claimed to have implemented the network topology and memory subsystem on the popular CloudSim simulation toolkit (Calheiros, Ranjan, Beloglazov, De Rose, & Buyya, 2011) and used the NAS Parallel Benchmarks (NPB) as HPC application which is divided into two groups: kernel benchmarks and pseudo-applications (Takouna, Dawoud, & Meinel, 2012). While compared to CPU utilization-based random placement algorithm, PVA is reported to have aggregated all the VMs belonging to an application into the same server and thus produced perfect VM placement after determining the traffic pattern of the communicating VMs. Moreover, the proposed approach have been shown to have outperformed the CPU-based placement in terms of reducing network link utilization through transferring inter-VM communication from shared network to shared memory by aggregating communicating VMs. In addition, the application performance degradation is computed and compared against the ideal execution time of the individual jobs and it is reported that 18% of the VMs suffer performance degradation while using PVA, whereas 20% performance degradation is experienced in the case of CPU-based placements.

Though PVA approach mentions where to migrate a VM, it does not make it clear when a VM requests for migration. Moreover, the associated VM migration overhead is not taken into account. Furthermore, it would not be always the case that all the VMs consisting of a parallel/HPC application can be aggregated into a single server. Finally, the evaluation lacks the reporting of the energy-efficiency aspect of the proposed approach.

The authors have presented a few future research work directions: 1) performance evaluation using different number of VMs for each application and 2) comparison with communication- and topology-aware VM placement approaches.

Application-Aware VM Placement in Data Centers

Song et al. (2012) have presented an application-aware VM placement problem focusing on energy-efficiency and scalability of modern data centers. The authors have pointed out several factors of modern data center management:

1. Increasing use of large-scale data processing services deployed in data centers,
2. Due to the rise of inter-VM bandwidth demands of modern applications, several recent network architecture scalability research works have been conducted with the goal of minimizing data center network costs by increasing the degree of network connectivity and adopting dynamic routing schemes,
3. Focusing on energy- and power-consumption minimization, several other recent works proposed mechanisms to improve server resource utilization and turning inactive servers to lower power states to save energy, and
4. Existing VM placement tools (e.g., VMware Capacity Planner (“VMware Capacity Planner”, 2014) and Novell PlateSpin

Recon ("Novell PlateSpin Recon", 2014)) are unaware of inter-VM traffic patterns, and thus can lead to placement decisions where heavily communicating VMs can be placed in physical servers with long distance network communication.

Similar to the work by Huang et al. (2012), Song et al. (2012) have expounded a VM placement problem based on proportional fairness and convex optimization to address the combined problem of reducing energy-consumption and data center traffic volume in order to improve scalability. During the problem formulation, both server-side resource capacity constraints and application-level inter-VM traffic demands are considered. However, given the problem definition, no algorithm or placement mechanism is presented in the work in order to solve the problem. Furthermore, simulation-based evaluation is presented and it is claimed that the combined VM placement algorithm outperforms random and FFD-based VM placement algorithms.

Application-Aware VM Migration in Data Centers

Shrivastava et al. (2011) have addressed the load balancing problem in virtualized data centers trough migration of overloaded VMs to underloaded physical servers such that the migration would be network-aware. The authors have argued that when VMs (part of multi-tier applications) are migrated to remove hot spots in data centers can introduce additional network overhead due to the inherent coupling between VMs based on communication, especially when moved to servers that are distant in terms of network distance. With the goal of finding destination servers for overloaded VMs that would result in minimum network traffic after the migration, the authors have formulated the VM migration as an optimization problem and proposed a network topology-aware greedy heuristic.

The proposed optimization problem is called application-aware since the complete application context running on top of the overloaded VM is considered during the migration decision. A view of the interconnections of the VMs comprising a multi-tier application is modeled as a dependency graph consisting of VMs as vertices and inter-VM communications as edges of the graph. The authors have also modeled the network cost function as a product of traffic demand of edge and network distance of the corresponding host servers, where such network distance can be defined as latency, delay, or number of hops between any two servers. Furthermore, server-side resource capacity constraint is also included in the problem formulation.

Since such optimization problem is NP-complete, the authors have proposed a greedy approximate solution named AppAware that attempts to reduce the cost during each migration decision step while considering both application-level inter-VM dependencies and underlying network topology. AppAware has the following four stages:

1. **Base Algorithm:** for each overloaded VM in the system, the total communication weight is computed and based on this the overloaded VMs are sorted in decreasing order, and then for each feasible destination server, the migration impact factor is computed. The impact factor gives a measure of the migration overhead based on the defined cost function due to the potential migration. Finally, the base algorithm selects the destination host for which the migration impact factor is the minimum, provided that the destination host has enough resources to accommodate the migrating VM.
2. **Incorporation of Application Dependency:** this part of AppAware computes the total cost to migrate a VM to a destination server as the sum of its individual cost corresponding to each of its peer VM that the migrating VM has communication.

3. **Topology Information and Server Load:** this part of AppAware considers network topology and neighboring server load while making migration decisions since a physical server that is close (in terms of topological distance) to other lightly loaded servers would be of higher preference as destination for a VM due to its potential for being capable of accommodating it dependent VMs to nearly servers.
4. **Iterative Refinements:** AppAware is further improved by incorporating two extensions to minimize the data center traffic. The first extension computes multiple values of the migration impact over multiple iterations of the AppAware base algorithm and the second extension further refines upon the previous extension by considering expected migration impact of future mappings of other VMs for a given candidate destination server at each iteration.

Based on numerical simulations, the authors have reported performance evaluation of AppAware by comparing with the optimal solution and Sandpiper black-box and grey-box migration scheme (Wood, Shenoy, Venkataramani, & Yousif, 2007). Run-time server-side remaining resource capacity (CPU, memory, and storage) and VM resource demands are generated using normal distribution, whereas inter-VM communication dependencies are generated using normal, exponential, and uniform distributions with varying mean and variance. Since the formulated migration problem is NP-hard, the performance of AppAware and Sandpiper are compared with optimal migration decisions only for small scale data centers (with 10 servers) and AppAware is reported to have produced solutions that are very close to the optimal solutions. For large data centers (with 100 servers), AppAware is compared against Sandpiper and it is reported that AppAware outperformed Sandpiper consistently by producing migration decisions that decreased traffic volume transported by the network by up to 81%. Moreover, in order to assess the suitability of AppAware against various network topologies, AppAware is compared to optimal placement decisions for Tree and VL2 network topologies. It is reported that AppAware performs close to optimal placement for Tree topology, whereas the gap is increased for VL2.

AppAware considered server-side resource capacity constraints during VM migration, but it does not consider the physical link bandwidth capacity constraints. As a consequence, subsequent VM migrations can cause network links of low distance to get congested.

4. COMPARATIVE ANALYSIS OF THE VM PLACEMENT AND MIGRATION TECHNIQUES

Besides resource capacity constraints on the physical computer servers, scalability and performance of data centers also depends on the efficient network resource allocations. With the growing complexity of the hosted applications and rapid rise in the volume of data associated to the application tasks, network traffic rates among the VMs running inside the data centers are increasing sharply. Such inter-VM data traffic exhibits non-uniform patterns and can change dynamically. As a result, this can cause bottlenecks and congestions in the underlying communication infrastructure. Network-aware VM placement and migration decisions have been considered as an effective tool to address this problem by assigning VMs to PMs with consideration of different data center characteristics and features, as well as traffic demands and patterns among the VMs.

The existing VM placement and migration techniques proposed by both academia and industry consider various system assumptions, problem modeling techniques and the features of the data centers and applications, as well as different solution and evaluation approaches. As

Table 1. Comparative analysis of the traffic-aware VM placement and migration techniques

Project	Network Topology-Aware VM Cluster Placement in IaaS Clouds
Salient Features	● VMs deployment as composite virtual infrastructure. ● Physical server resource capacity constraints. ● User provided prospective traffic patterns and bandwidth requirements among VMs in the form of XML configuration. ● Possible anti-colocation condition among VMs. ● Physical infrastructure interconnection following PortLand network topology. ● Two-layered framework: physical infrastructure and middleware.
Advantages	● Suggested VIBES algorithm incrementally searches for a neighborhood by utilizing PortLand's topological features with sufficient physical resources and VIO places the virtual infrastructure within the neighborhood. This approach has the advantage that all the VMs of the whole virtual infrastructure are placed in near proximity within the network topology. ● Use of greedy heuristics ensures fast placement decisions. ● Placements of VMs with higher inter-VM traffic demands in topologically near physical servers suggests lower network utilization and possible accommodation of higher number of VMs.
Drawbacks	● VM Placement decisions focusing on network utilization may result in significant compute resource wastage and less energy efficient. ● Expected inter-VM traffic demands may not always be readily available to Cloud users and dynamic traffic patterns can different from the initial estimation. ● In a dynamic data center, VMs are deployed and terminated at runtime and the initial traffic-aware VM placement decisions may not remain network efficient as time passes. Such approaches can be complemented through the use of dynamic (periodic or event triggered) VM migration and reconfiguration decisions.
Project	**Stable Network-Aware VM Placement for Cloud Systems**
Salient Features	● Graph transformation techniques to convert complex network topologies (e.g., Fat-tree and VL2) to plain tree topology. ● Minimization of the ratio between the inter-VM bandwidth requirements and physical link bandwidth capacities. ● Integer Quadratic Programming model-based Min Cut Ratio-aware VM Placement (MCRVMP) problem definition with server and network resource capacities constraints. ● Grouping of communicating VMs in data center as connected components and dynamic relocation of the connected components in order to minimize network overhead on physical network infrastructure. ● Two VM placement heuristic algorithms: o Integer Programming-based recursive algorithm, and o Iteration-based greedy placement algorithm.
Advantages	● Grouping of communicating VMs into smaller-sized connected components ensure faster VM placement decision. ● Though the proposed VM placement algorithms works on tree topology, by the use of topology conversion techniques the algorithms can be applied for much complex network architectures. ● As reported by the experimental evaluation using NS2 network simulator, the proposed VM placement techniques experience zero dropped packets and can absorb time-varying traffic demands up to three times the nominal values.
Drawbacks	● Cost or overhead of necessary VM migrations are not considered in the problem formulation and solution techniques. ● The quality of the VM placement solutions were compared to random and optimal solutions only for small problems and not evaluated against other placement techniques for larger data centers.

continued on following page

a consequence, comparative analysis in a uniform fashion of such techniques becomes quite tricky. Moreover, VM placement and migration is a broad area of research with various optimization and objectives. Some of the techniques strive for single-objective optimization, while others try to incorporate multiple objectives while making VM placement and relocation decisions. Taking into account the various aspects and features considered and proposed in the network-aware VM placement and migration strategies, detailed comparative analyses are presented in Tables 1, 2, 3, and 4 grouped by the subdomains they are categorized in.

Table 1. Continued

Project	Scalability Improvement of Data Center Networks with Traffic-Aware VM Placement
Salient Features	• Three observed dominant trends of data center traffic patterns: o Low correlation between mean traffic rates of VM pairs and the corresponding end-to-end physical communication distance/cost. o Highly non-uniform traffic distribution for individual VMs. o Traffic rates between VM pairs tend to remain relatively constant. • Definition of the traffic-aware VM placement problem as a NP-hard combinatorial optimization problem belonging to the family of the Quadratic Assignment Problems. • The goal of the defined problem is minimization of aggregate traffic rates at each network switch. • The cost of placing any two VMs with traffic flows is defined as the number of hops or switches on the routing path of the VM pairs. • A concept of slot is incorporated to represent one CPU/memory allocation on physical server. Multiple such slots can reside on the same server and each slot can be allocated to any VM.
Advantages	• Adaptation of divide-and-conquer strategy to group all the slots based on the cost among the slots. This approach helps reduce the problem space into smaller sub-problems. • The proposed Cluster-and-Cut algorithm finds VM-to-PM assignment decisions to place VM pairs with high mutual traffic on PM pairs with low cost communication links. • Trace-driven simulation using global and partitioned traffic model, as well as hybrid traffic model combining real traces from production data centers with classical Gravity model.
Drawbacks	• The formulated Traffic-aware VM Placement Problem does not consider the physical link capacity constraints. • It is assumed that static layer 2 and 3 routing protocols are deployed in the data center. • VM migration overhead incurred due to the offline VM shuffling is not considered. • The proposed Cluster-and-Cut algorithm places only one VM per server that can result in high amount of resource wastage.

Finally, Table 5 illustrates the most significant aspects of the reviewed research projects that are highly relevant to network-aware VM placement and migration techniques.

5. FUTURE RESEARCH DIRECTIONS

VM consolidation and resource reallocation through VM migrations with focus on both energy-awareness and network overhead is yet another area of research that requires much attention. VM placement decisions focusing primarily on server resource utilization and energy consumption reduction can produce data center configurations that are not traffic-aware or network optimized, and thus can lead to higher SLA violations. As a consequence, VM placement strategies utilizing both VM resource requirements information and inter-VM traffic load can come up with placement decisions that are more realistic and efficient.

Cloud environments allow their consumers to deploy any kind of applications in an on-demand fashion, ranging from compute intensive applications such as HPC and scientific applications, to network and disk I/O intensive applications like video streaming and file sharing applications. Co-locating similar kinds of applications in the same physical server can lead to resource contentions for some types of resources while leaving other types underutilized. Moreover, such resource contention will have adverse effects on application performance, thus leading to SLA violations and profit minimization. Therefore, it is important to understand the behavior and resource usage patterns of the hosted applications in order to efficiently place VMs and allocate resources to the applications. Utilization of historical workload data and application of appropriate load prediction mechanisms need to be integrated with VM consolidation techniques to minimize resource contentions among applications and increase resource utilization and energy efficiency of data centers.

Table 2. Comparative analysis of the network-aware energy-efficient VM placement and migration techniques

Project	**Multi-Objective Virtual Machine Migration in Virtualized Data Center Environments**
Salient Features	● Definition of VM migration problem as multi-objective optimization with the goal of maximization of resource utilization and minimization of network traffic. ● Three levels of joint optimization framework: o **Server Consolidation:** Minimization of the number of active physical servers and reduce energy consumption. o Minimization of the total communication cost after necessary VM migrations. o Combined goal of minimizing energy consumption and total communication costs. ● Two-staged greedy heuristic solution to compute overloaded VM migration decisions: o Application of dominant resource share of servers. o Selection of destination server for migration with minimum dominant resource share and communication traffic among VMs.
Advantages	● VM migration decisions consider minimum migration impact of overloaded VMs. ● Combined optimization of energy consumption and network traffic.
Drawbacks	● Exhaustive search-based solution generation.
Project	**Communication Traffic Minimization with Power-Aware VM Placement in Data Centers**
Salient Features	● VMs located in same server would communicate using memory copy rather than network links, thus reduce total network traffic. ● Definition of dynamic VM placement problem as a reduced minimum k-cut problem (NP-hard). ● Two-fold objectives of minimizing total network traffic and energy consumption through VM consolidation. ● Server side resource capacity constraints as VM placement constraints. ● Solution approach utilizes K-means clustering algorithm with following distinguishing features: o Minimization of the negative impact of placement randomization o Reduction of the number of migration ● Method for computing the communication distance between a VM and a cluster.
Advantages	● Suggested solutions address both online dynamic VM migration and offline deployment of new VM requests. ● Evaluation using workload traces from production data centers. ● Multiple goals of reducing power consumption and network traffic.
Drawbacks	● Most of the compared VM placement approaches are network-agnostic.
Project	**Energy-Aware Virtual Machine Placement in Data Centers**
Salient Features	● Balanced optimization between server power consumption and network-infrastructure power consumption. ● Definition of three-phased optimization framework: o Maximization of server resource utilization and reduction of power consumption. o Minimization of total aggregated communication costs. o Fuzzy-logic system-based energy-aware joint VM placement with trade-off between the above two optimizations. ● Clustering of VMs and PMs based on the amount of communication traffic and network distances. ● Broad range of experimental evaluation comparing with multiple existing VM placement approaches using different network topologies.
Advantages	● Multiple objectives focusing on optimizations of resource utilization, data center power consumption, and network resource utilization. ● Partitioning of VMs into disjoint sets helps reduce the problem space and find solutions in reduced time.
Drawbacks	● Impacts of necessary VM migrations and reconfigurations are not considered in the modeled problem and proposed solution approaches: o Increased traffic due to required VM migrations could impose overhead in network communication. o VM migrations can have detrimental effects on hosted applications SLA due to VM download time.

Table 3. Comparative analysis of the network- and data-aware VM placement and migration techniques

Project	Coupled Placement in Modern Data Centers
Salient Features	• Network-focused joint (pair-wise) compute and data component placement. • Heterogeneous data center comprised of storage and network devices with built-in compute facilities and diversified performance footprints. • User defined network cost function. • Joint compute and data component placement problem modeled as Knapsack Problem and Stable-Marriage Problem. • Proposed Couple Placement Algorithm based on iterative refinement using pair-wise swap of application compute and storage components.
Advantages	• Incorporation of data components associated with application compute components and the corresponding traffic rates in application placement. • Incorporation of physical storage nodes and the corresponding network distances to the compute servers in cost definition. • Featured advanced properties and features of modern data center devices.
Drawbacks	• Compared to modern Cloud applications (composite and multi-tiered), the proposed Couple Placement Problem (CPP) assumes simplistic view of the application having only one compute and one data component. • CPP considers the server side resource capacity constraint as single dimensional (only CPU-based), whereas this is in fact a multi-dimensional problem (Ferdaus et al., 2014). • Network link bandwidth capacity is not considered. • VM and data components reconfiguration and relocation overhead is not considered in the problem formulation.
Project	**Network- and Data Location-Aware VM Placement and Migration Approach in Cloud Computing**
Salient Features	• Cloud applications with associated data components spread across one or more storage Clouds. • Single VM placement (initial) and overloaded VM migration decisions. • Initial fixed location of data components. • Modeled network link speed depends on both the size of the data transmitted and the packet transfer time. • Allocations of application compute components (i.e. VMs) with consideration of the associated data access time.
Advantages	• Consideration of data location during VM placement and migration decisions.
Drawbacks	• Over simplified view of federated Cloud data centers. • Exhaustive search-based solution approaches that can be highly costly as data center size increases. • VM migration and reconfiguration overheads are not considered. • Over simplified and small scale evaluation of the proposed VM placement and migration algorithms comparing with network-agnostic VM placement algorithm of CloudSim simulation toolkit.

Centralized VM consolidation and placement mechanisms can suffer from the problems of scalability and single-point-of-failure, especially for Cloud data centers. One possible solution approach would be replication of VM consolidation managers; however such decentralized approach is non-trivial since VMs in the date centers are created and terminated dynamically through on-demand requests of Cloud consumers, and as a consequence consolidation managers need to have updated information about the data center. As initial solution, servers can be clustered and assigned to the respective consolidation managers and appropriate communication and synchronization among the managers need to be ensured to avoid possible race conditions.

VM migration and reconfiguration overhead can have adverse effect on the scalability and bandwidth utilization of data centers, as well as application performance. As a consequence, VM placement and scheduling techniques that are

Table 4. Comparative analysis of the application-aware VM placement and migration techniques

Project	Communication-Aware Scheduling for Parallel Applications in Virtualized Data Centers
Salient Features	● Network-aware VM placement with focused on Parallel and HPC applications. ● Dynamic VM reconfiguration through VM migrations based on communication patterns with peer-VMs of HPC applications. ● Proposed approach iteratively refines the VMs placement through VM migrations with the goal of accumulating VMs (with traffic dependencies) of the same HPC application in the same server. ● VM migration follows a ranking system based on the total number of input/output traffic flows.
Advantages	● Reactive VM scheduling approach to dynamic (run-time) changes of the inter-VM communication patterns. ● Multiple objectives to optimization communication overhead and delay, as well as energy consumption.
Drawbacks	● It is unclear when a VM triggers it migration request. ● Associated VM migration overhead is not considered in the problem statement. ● Depending on the size of the HPC applications and the resource capacities of the physical servers, it is not guaranteed that all the VMs of a HPC application can be placed in a single server. ● The reported experimental evaluation does not show improvement in terms of energy consumption.
Project	**Application-Aware VM Placement in Data Centers**
Salient Features	● Combined optimization of data center power consumption and network traffic volume. ● Proposed modeling considers server-side resource capacity constraints and application-level communication dependencies among the VMs.
Advantages	● Multiple optimizations of both network traffic and power consumption.
Drawbacks	● Presented work lacks sufficient information regarding VM placement algorithm or scheduling. ● Simulation-based evaluation considers network-agnostic competitors.
Project	**Application-Aware VM Migration in Data Centers**
Salient Features	● Load balancing through network-aware migration of overloaded VMs. ● VM migration decisions considers complete application context in terms of peer VMs with communication dependencies. ● Network cost is modeled as a product of traffic demands and network distance. ● Server side resource capacity constraints are considered during VM migration decisions.
Advantages	● Network topology-aware VM migration decisions. ● Iterative improvement is suggested to minimize data center traffic volume.
Drawbacks	● Physical link capacity constraints are not considered while mapping overloaded VMs to underloaded physical servers.

unaware of VM migration and reconfiguration overhead can effectively congest the network and cause SLA violations unbeknown. Incorporation of the estimated migration overhead with the placement strategies and optimization of VM placement and migration through balancing the utilization of network resources, migration overhead, and energy consumption are yet to explore areas of data center virtual resource management. With various trade-offs and balancing tools, data center administrators can have the freedom of tuning the performance indicators for their data centers.

6. CONCLUSION

Cloud Computing is quite a new computing paradigm and from the very beginning it has been growing rapidly in terms of scale, reliability, and availability. Because of its flexible pay-as-you-go business model, virtually infinite pool of on-demand resources, guaranteed QoS, and almost perfect reliability, consumer base of Cloud Computing is increasing day-by-day. As a result, Cloud providers are deploying large data centers across the globe. Such data centers extensively use virtualization technologies in order to utilize the

Table 5. Aspects of the notable research works on network-aware VM placement and migration

Research Project	System Assumption	Net. Arch/ Topology	Placement Type	Modeling/ Analysis Technique	Physical Resources	VM Placement Constraints	Objective/ Goal	Solution Approach/ Algorithm	Evaluation/ Experimental Platform	Competitor Approaches	Workload/ VM-Cluster in Experiments/ Evaluation	Evaluation Performance Metrics
Multi-objective Virtual Machine Migration in Virtualized Data Center Environments (Huang et al., 2013)	Homogeneous data center	Tree, VL2, Fat-tree, BCube	Online	Max-min fairness and convex optimization framework	CPU, memory, and storage	Server resource capacity and inter-VM bandwidth requirement	Maximize utilization of physical servers and minimize data center network traffic	Two-staged greedy heuristic (exhaustive search based)	Simulation based on synthetic data center and load characteristics	AppAware: Application-aware VM Migration (Shrivastava et al., 2011)	Normal distribution-based load characteristics for VMs and servers, and inter-VM traffic demands	Reduction in network traffic and average impact of migration
Communica-tion Traffic Minimiza-tion with Power-aware VM Placement in Data Centers (Zhang et al., 2012)	Homogeneous data center	N/A	Online and Offline	Mathematical optimization (minimization)	CPU, memory, network, and I/O	Server resource capacity	Minimization of communi-cation traffic and power cost	Greedy heuristic	Simulation based on synthetic data center and load characteristics	Random placement, simple greedy, and First Fit (FF)	Workload traces from production data center	Total communica-tion traffic and number of active servers
Energy-aware Virtual Machine Placement in Data Centers (Huang et al, 2012)	Homogeneous data center	Tree, VL2, Fat-tree, BCube	Online	Mathematical Optimization Framework [Proportional Fairness and (Convex Optimization)]	CPU, memory, and storage	Server resource capacity and inter-VM bandwidth requirement	Reduction of data Transmission and minimization of energy consumption of server and network device	Greedy approach	Simulation based on synthetic data center and load characteristics	Random placement, First Fit Decreasing (FFD), Grouping Genetic Algorithm (GGA), two stage heuristic algorithm, optimal placement	Normal distribution-based load characteristics for VMs and servers, and inter-VM traffic demands	Objective function value, reduction rate of traffic volume, and number of used servers
Network-aware VM Placement and Migration Approach in Cloud Computing (Piao & Yan, 2010)	IaaS Compute and Storage Data Center	Federated and Distribu-ted Cloud Data Centers	Offline and Online		CPU and Memory	Server resource capacity constraints	Minimization of data transfer time consumption	Exhaustive search	Fixed simulation scenario implemented in CloudSim (Buyya, Ranjan, & Calheiros, 2009)	Default VM placement policy of CloudSim	Small-scale fixed valued workload data	Average task completion time
Coupled Placement in Modern Data Centers (Korupolu et al., 2009)	Heterogeneous data center	SAN Data Center using Core-Edge design pattern	Offline	Knapsack Problem and Stable Marriage Problem	CPU and Storage	Server node CPU and storage node capacity	Minimization of total network cost across all application communication links	Individual and pairwise Greedy heuristic, and iterative refinement-based heuristic	Heteroge-neous SAN data center and synthetic workload-based simulation	LP-based optimal placement	Normal distribution-based compute and storage resource demands and network I/O rates	Defined network cost function value and placement computation time
Network Topology-aware VM Cluster Placement in IaaS Clouds (Georgiou et al., 2013)	IaaS Cloud Data Center, Homogeneous	PortLand	Offline	Graph search	CPU, Memory, and Storage	VM Anti-colocation, server resource capacity constraints	Minimization of network utilization, Low decision time overhead regardless of infrastructure size	Greedy Approach, Recursive, Backtrack-ing	Simulation (JgraphT lib)	First Fit Decreasing (FFD)	Workflow structures (Pipeline, Data Aggregation, and Epigenomics)	Network utilization,

continued on following page

Table 5. Continued

Research Project	System Assumption	Net. Arch/ Topology	Placement Type	Modeling/ Analysis Technique	Physical Resources	VM Placement Constraints	Objective/ Goal	Solution Approach/ Algorithm	Evaluation/ Experimental Platform	Competitor Approaches	Workload/ VM-Cluster in Experiments/ Evaluation	Evaluation Performance Metrics
Improving the Scalability of Data Center Networks with Traffic-aware VM Placement (Meng et al., 2010)	N/A	Tree, VL2, Fat-tree, BCube	Both Offline and Online	NP-hard Combinatorial Optimization Problem, instance of Quadratic Assignment Problem (QAP)	CPU and Memory	Maximum placement of one VM per PM	Improvement of scalability by minimizing the aggregated traffic rates at each network switch	Greedy approximate heuristic employing divide-and-conquer strategy and minimum k-cut graph algorithm	Trace-driven simulation using global and partitioned traffic model, as well as hybrid traffic model combining real traces with classical Gravity model	Local Optimal Pairwise Interchange (LOPI) (Armour & Buffa, 1963) and Simulated Annealing (Burkard & Rendl, 1984)	Inter-VM traffic rates (aggregated incoming and outgoing) collected from production data centers.	
Stable Network-aware VM Placement for Cloud Systems (Biran et al., 2012)	Homogeneous Data Center	Tree, Fat-tree, VL2	Offline	NP-hard Integer Quadratic Programming	CPU and Memory	Server resource capacity constraints, physical link bandwidth capacity constraints	Minimization of the maximum ratio of the demand and capacity across all network cuts	Integer Programming Techniques employing divide-and-concur strategy and Greedy heuristics	Simulation-based using IBM ILOG CPLEX mixed integer mathematical solver.	Random and optimal placement	Gaussian distribution based inter-VM and VM-gateway traffic demands. Equal server resource capacity and VM resource demand.	Worst case and average network cut load ratio (utilization), placement solving time, % of dropped packets, and avg. packet delivery delay
Communication-Aware and Energy-Efficient Scheduling for Parallel Applications in Virtualized Data Centers (Takouna et al., 2013)	Homogeneous data center	Tree topology based on core-aggregation-edge model	Online	Simple peer-based inter-VM communication pattern	CPU, memory, and network I/O	Server resource capacity and inter-VM bandwidth requirement	Minimization of energy-consumption by servers and network components, as well as average network utilization	Iterative greedy that ranks VMs based on the number of in/out traffic flow	Simulation-based (network and memory subsystem implemented on CloudSim (Calheiros et al., 2011))	Simple CPU utilization-based random VM placement	NPB parallel application benchmark used as HPC application	Uniformity of VM placement on servers, average utilization of network links, and application performance degradation
Application-aware VM Placement in Data Centers (Song et al., 2012)	Homogeneous data center	Tree, VL2, Fat-tree, BCube	Online	Proportional Fairness and Convex Optimization	CPU, memory, and storage	Inter-VM bandwidth requirement and server resource capacity	Reduction of data transmission and energy consumption	N/A	Simulation based on synthetic data center and load characteristics	Random placement and First Fit Decreasing (FFD)	Normal distribution-based load characteristics for VMs and servers, and inter-VM traffic demands	Objective function value and reduction rate of traffic volume
Application-aware VM Migration in Data Centers (Shrivastava et al., 2011)	Homogeneous data center	Tree and VL2	Online	Mathematical optimization, multiple knapsack problem	CPU, memory, and storage	Server resource capacity	Minimization of network overhead due to VM migration	Greedy heuristic (exhaustive)	Simulation based on synthetic data center and load characteristics	Optimal placement (CPLEX solver) and Sandpiper VM migration scheme (Wood et al., 2007)	Normal distribution-based server resource and VM demands and, Normal, exponential, and uniform distribution-based inter-VM traffic demands	Objective function value and reduction in data center traffic

underlying effectively and with much higher reliability. With increasing deployment of data- and communication-intensive composite applications in the virtualized data centers, traffic volume transferred through the network devices and links are also increasing rapidly. Performance of these applications is highly dependent on the communication latencies and thus can have tremendous effects on the agreed SLA guarantees. Since SLA violations result in direct revenue reduction for the Cloud data center providers, efficient utilization of the network resources is highly important. Intelligent VM placement and migration is one of the key tools to maximize utilization of data center network resources. When coupled with effective prediction mechanism of inter-VM communication pattern, VM placement strategies can be utilized to localize bulk of the intra-data center traffic. This localization would further help in reducing packet switching and forwarding load in the higher level switches, which will be helpful in reducing energy consumption of the data center network devices.

This chapter has presented the motivation and background knowledge related to the network-aware VM placement and migration in data centers. Afterwards, a detailed taxonomy and characterization on the existing techniques and strategies have been expounded followed by an elaborate survey on the most notable recent research works. A comprehensive comparative analysis highlighting the significant features, benefits, and limitations of the techniques has been put forward, followed by a discussion on the future research outlooks.

REFERENCES

Adra, B., Blank, A., Gieparda, M., Haust, J., Stadler, O., & Szerdi, D. (2004). Advanced power virtualization on ibm eserver p5 servers: Introduction and basic configuration. IBM Corp.

Agrawal, S., Bose, S. K., & Sundarrajan, S. (2009). Grouping genetic algorithm for solving the server consolidation problem with conflicts. In *Proceedings of the First ACM/SIGEVO Summit on Genetic and Evolutionary Computation*. ACM. doi:10.1145/1543834.1543836

Al-Fares, M., Loukissas, A., & Vahdat, A. (2008). A scalable, commodity data center network architecture. *Computer Communication Review*, *38*(4), 63–74. doi:10.1145/1402946.1402967

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., & Konwinski, A. (2010). A view of cloud computing. *Communications of the ACM*, *53*(4), 50–58. doi:10.1145/1721654.1721672

Armour, G. C., & Buffa, E. S. (1963). A heuristic algorithm and simulation approach to relative location of facilities. *Management Science*, *9*(2), 294–309. doi:10.1287/mnsc.9.2.294

Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., & Warfield, A. et al. (2003). Xen and the art of virtualization. *Operating Systems Review*, *37*(5), 164–177. doi:10.1145/1165389.945462

Bharathi, S., Chervenak, A., Deelman, E., Mehta, G., Su, M.-H., & Vahi, K. (2008). Characterization of scientific workflows. In *Proceedings of Third Workshop on Workflows in Support of Large-Scale Science*. Academic Press.

Bhuyan, L. N., & Agrawal, D. P. (1984). Generalized hypercube and hyperbus structures for a computer network. *IEEE Transactions on Computers*, *100*(4), 323–333.

Biran, O., Corradi, A., Fanelli, M., Foschini, L., Nus, A., Raz, D., & Silvera, E. (2012). A stable network-aware VM placement for cloud systems. In *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing* (CCGRID 2012) (pp 498-506). IEEE. doi:10.1109/CCGrid.2012.119

Burkard, R. E., & Rendl, F. (1984). A thermodynamically motivated simulation procedure for combinatorial optimization problems. *European Journal of Operational Research*, *17*(2), 169–174. doi:10.1016/0377-2217(84)90231-5

Buyya, R., Broberg, J., & Goscinski, A. M. (2010). *Cloud computing: Principles and paradigms* (Vol. 87). John Wiley & Sons.

Buyya, R., Ranjan, R., & Calheiros, R. N. (2009). Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: Challenges and opportunities. In *Proceedings of High Performance Computing & Simulation*. Academic Press.

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software, Practice & Experience*, *41*(1), 23–50. doi:10.1002/spe.995

Chen, M., Zhang, H., Su, Y.-Y., Wang, X., Jiang, G., & Yoshihira, K. (2011). Effective vm sizing in virtualized data centers. In *Proceedings of Integrated Network Management* (pp. 594-601). Academic Press. doi:10.1109/INM.2011.5990564

Cisco, *M. D. S. 9000 SANTap*. (2014). Retrieved from http://www.cisco.com/c/en/us/products/collateral/storage-networking/mds-9000-santap/data_sheet_c78-568960.html

Cisco Data Center Infrastructure 2.5 Design Guide. (2014). Retrieved from http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/DC_Infra2_5/DCI_SRND_2_5a_book/DCInfra_1a.html

Clark, C., Fraser, K., Hand, S., Hansen, J. G., Jul, E., Limpach, C., . . . Warfield, A. (2005). Live migration of virtual machines. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation* (vol. 2, pp. 273-286). Academic Press.

Crosby, S., & Brown, D. (2006). The virtualization reality. *Queue*, *4*(10), 34–41. doi:10.1145/1189276.1189289

Ersoz, D., Yousif, M. S., & Das, C. R. (2007). Characterizing network traffic in a cluster-based, multi-tier data center. In *Proceedings of Distributed Computing Systems* (pp. 59-59). Academic Press. doi:10.1109/ICDCS.2007.90

Ferdaus, M. H., Murshed, M., Calheiros, R. N., & Buyya, R. (2014). Virtual machine consolidation in cloud data centers using ACO metaheuristic. In Proceedings of Euro-Par 2014 Parallel Processing (pp. 306-317). Springer. doi:10.1007/978-3-319-09873-9_26

Georgiou, S., Tsakalozos, K., & Delis, A. (2013). Exploiting network-topology awareness for VM placement in IaaS clouds. In *Proceedings of Cloud and Green Computing* (CGC) (pp. 151-158). Academic Press. doi:10.1109/CGC.2013.30

Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., & Sengupta, S. et al. (2009). VL2: A scalable and flexible data center network. *Computer Communication Review*, *39*(4), 51–62. doi:10.1145/1594977.1592576

Guo, C., Lu, G., Li, D., Wu, H., Zhang, X., Shi, Y., & Lu, S. et al. (2009). BCube: A high performance, server-centric network architecture for modular data centers. *Computer Communication Review*, *39*(4), 63–74. doi:10.1145/1594977.1592577

Guo, C., Wu, H., Tan, K., Shi, L., Zhang, Y., & Lu, S. (2008). Dcell: A scalable and fault-tolerant network structure for data centers. *Computer Communication Review*, *38*(4), 75–86. doi:10.1145/1402946.1402968

Gupta, R., Bose, S. K., Sundarrajan, S., Chebiyam, M., & Chakrabarti, A. (2008). A two stage heuristic algorithm for solving the server consolidation problem with item-item and bin-item incompatibility constraints. In *Proceedings of Services Computing* (Vol. 2, pp. 39-46). IEEE. doi:10.1109/SCC.2008.39

Hines, M. R., Deshpande, U., & Gopalan, K. (2009). Post-copy live migration of virtual machines. *Operating Systems Review*, *43*(3), 14–26. doi:10.1145/1618525.1618528

Huang, D., Gao, Y., Song, F., Yang, D., & Zhang, H. (2013). Multi-objective virtual machine migration in virtualized data center environments. In *Proceedings of Communications* (ICC) (pp. 3699-3704). IEEE. doi:10.1109/ICC.2013.6655129

Huang, D., Yang, D., Zhang, H., & Wu, L. (2012). Energy-aware virtual machine placement in data centers. In *Proceedings of Global Communications Conference* (GLOBECOM). IEEE.

Kandula, S., Sengupta, S., Greenberg, A., Patel, P., & Chaiken, R. (2009). The nature of data center traffic: Measurements & analysis. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement* (pp. 202-208). ACM. doi:10.1145/1644893.1644918

Kivity, A., Kamay, Y., Laor, D., Lublin, U., & Liguori, A. (2007). KVM: The Linux virtual machine monitor. In *Proceedings of the Linux Symposium* (*Vol. 1,* pp. 225-230). Academic Press.

Kliazovich, D., Bouvry, P., & Khan, S. U. (2013). DENS: Data center energy-efficient network-aware scheduling. *Cluster Computing*, *16*(1), 65–75. doi:10.1007/s10586-011-0177-4

Korupolu, M., Singh, A., & Bamba, B. (2009). Coupled placement in modern data centers. In *Proceedings of Parallel & Distributed Processing* (pp. 1-12). IEEE. doi:10.1109/IPDPS.2009.5161067

Kusic, D., Kephart, J. O., Hanson, J. E., Kandasamy, N., & Jiang, G. (2009). Power and performance management of virtualized computing environments via lookahead control. *Cluster Computing*, *12*(1), 1–15. doi:10.1007/s10586-008-0070-y

Leiserson, C. E. (1985). Fat-trees: Universal networks for hardware-efficient supercomputing. *IEEE Transactions on Computers*, *100*(10), 892–901.

Lo, J. (2005). *VMware and CPU virtualization technology*. World Wide Web Electronic Publication.

Loiola, E. M., de Abreu, N. M. M., Boaventura-Netto, P. O., Hahn, P., & Querido, T. (2007). A survey for the quadratic assignment problem. *European Journal of Operational Research*, *176*(2), 657–690. doi:10.1016/j.ejor.2005.09.032

Mann, V., Gupta, A., Dutta, P., Vishnoi, A., Bhattacharya, P., Poddar, R., & Iyer, A. (2012). Remedy: Network-aware steady state VM management for data centers. In Proceedings of Networking 2012 (pp. 190-204). Springer.

McVitie, D. G., & Wilson, L. B. (1971). The stable marriage problem. *Communications of the ACM*, *14*(7), 486–490. doi:10.1145/362619.362631

Meng, X., Isci, C., Kephart, J., Zhang, L., Bouillet, E., & Pendarakis, D. (2010). Efficient resource provisioning in compute clouds via VM multiplexing. In *Proceedings of the 7th International Conference on Autonomic Computing* (pp. 11-20). Academic Press. doi:10.1145/1809049.1809052

Meng, X., Pappas, V., & Zhang, L. (2010). Improving the scalability of data center networks with traffic-aware virtual machine placement. In Proceedings of INFOCOM (pp. 1-9). IEEE. doi:10.1109/INFCOM.2010.5461930

Mysore, R. N., Pamboris, A., Farrington, N., Huang, N., Miri, P., Radhakrishnan, S., & Vahdat, A. et al. (2009). Portland: A scalable fault-tolerant layer 2 data center network fabric. *Computer Communication Review*, *39*(4), 39–50. doi:10.1145/1594977.1592575

Nelson, M., Lim, B.-H., Hutchins, G., & Associates. (2005). Fast transparent migration for virtual machines. In *Proceedings of USENIX Annual Technical Conference, General Track* (pp. 391-394). Academic Press.

Nimbus is Cloud Computing for Science. (2014). Retrieved from http://www.nimbusproject.org/

Novell PlateSpin Recon. (2014). Retrieved from https://www.netiq.com/products/recon/

Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., & Zagorodnov, D. (2009). The eucalyptus open-source cloud-computing system. In *Proceedings of Cluster Computing and the Grid* (pp. 124-131). IEEE. doi:10.1109/CCGRID.2009.93

Piao, J. T., & Yan, J. (2010). A network-aware virtual machine placement and migration approach in cloud computing. In *Proceedings of Grid and Cooperative Computing* (GCC) (pp. 87-92). Academic Press. doi:10.1109/GCC.2010.29

Pisinger, D. (1997). A minimal algorithm for the 0-1 knapsack problem. *Operations Research*, *45*(5), 758–767. doi:10.1287/opre.45.5.758

Sapuntzakis, C. P., Chandra, R., Pfaff, B., Chow, J., Lam, M. S., & Rosenblum, M. (2002). Optimizing the migration of virtual computers. *ACM SIGOPS Operating Systems Review, 36*(SI), 377-390.

Saran, H., & Vazirani, V. V. (1995). Finding k cuts within twice the optimal. *SIAM Journal on Computing*, *24*(1), 101–108. doi:10.1137/S0097539792251730

Shrivastava, V., Zerfos, P., Lee, K.-W., Jamjoom, H., Liu, Y.-H., & Banerjee, S. (2011). Application-aware virtual machine migration in data centers. In Proceedings of INFOCOM, 2011 (pp. 66–70). IEEE. doi:10.1109/INFCOM.2011.5935247

Smith, J., & Nair, R. (2005). *Virtual machines: Versatile platforms for systems and processes*. Elsevier.

Song, F., Huang, D., Zhou, H., & You, I. (2012). Application-aware virtual machine placement in data centers. In *Proceedings of Innovative Mobile and Internet Services in Ubiquitous Computing* (IMIS) (pp. 191-196). Academic Press. doi:10.1109/IMIS.2012.119

Sotomayor, B., Montero, R. S., Llorente, I. M., & Foster, I. (2009). Virtual infrastructure management in private and hybrid clouds. *IEEE Internet Computing*, *13*(5), 14–22. doi:10.1109/MIC.2009.119

Takemura, C. & Crawford, L. S. (2009). *The book of Xen: A practical guide for the system administrator*. No Starch Press.

Takouna, I., Dawoud, W., & Meinel, C. (2012). Analysis and simulation of HPC applications in virtualized data centers. In *Proceedings of Green Computing and Communications* (GreenCom) (pp. 498-507). IEEE. doi:10.1109/GreenCom.2012.80

Takouna, I., Rojas-Cessa, R., Sachs, K., & Meinel, C. (2013). Communication-aware and energy-efficient scheduling for parallel applications in virtualized data centers. In *Proceedings of Utility and Cloud Computing* (UCC) (pp. 251-255). IEEE. doi:10.1109/UCC.2013.50

Vaquero, L., Rodero-Merino, L., Caceres, J., & Lindner, M. (2008). A break in the clouds: Towards a cloud definition. *Computer Communication Review*, *39*(1), 50–55. doi:10.1145/1496091.1496100

VMware Capacity Planner. (2014). Retrieved from http://www.vmware.com/products/capacity-planner

Wood, T., Shenoy, P. J., Venkataramani, A., & Yousif, M. S. (2007). *Black-box and gray-box strategies for virtual machine migration*. NSDI.

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, *16*(3), 645–678.

Zayas, E. (1987). Attacking the process migration bottleneck. *Operating Systems Review*, *21*(5), 13–24. doi:10.1145/37499.37503

Zhang, B., Qian, Z., Huang, W., Li, X., & Lu, S. (2012). Minimizing communication traffic in data centers with power-aware VM placement. In *Proceedings of Innovative Mobile and Internet Services in Ubiquitous Computing* (IMIS) (pp. 280-285). Academic Press. doi:10.1109/IMIS.2012.71

Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: State-of-the-art and research challenges. *Journal of Internet Services and Applications, 1*(1), 7-18.

ADDITIONAL READING

Al-Fares, M., Radhakrishnan, S., Raghavan, B., Huang, N., & Vahdat, A. (2010). Hedera: Dynamic Flow Scheduling for Data Center Networks. In NSDI (Vol. 10, pp. 19-19).

Ballani, H., Jang, K., Karagiannis, T., Kim, C., Gunawardena, D., & O'Shea, G. (2013). Chatty Tenants and the Cloud Network Sharing Problem. In NSDI (pp. 171-184).

Bansal, N., Lee, K. W., Nagarajan, V., & Zafer, M. (2011). Minimum congestion mapping in a cloud. In *Proceedings of the 30th annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing* (pp. 267-276). ACM. doi:10.1145/1993806.1993854

Benson, T., Akella, A., Shaikh, A., & Sahu, S. (2011). CloudNaaS: a cloud networking platform for enterprise applications. In *Proceedings of the 2nd ACM Symposium on Cloud Computing* (p. 8). ACM. doi:10.1145/2038916.2038924

Bose, S. K., & Sundarrajan, S. (2009). Optimizing migration of virtual machines across datacenters. In Parallel Processing Workshops, 2009. ICPPW'09. International Conference on (pp. 306-313). IEEE. doi:10.1109/ICPPW.2009.39

Calcavecchia, N. M., Biran, O., Hadad, E., & Moatti, Y. (2012). VM placement strategies for cloud scenarios. In Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on (pp. 852-859). IEEE. doi:10.1109/CLOUD.2012.113

Chaisiri, S., Lee, B. S., & Niyato, D. (2009). Optimal virtual machine placement across multiple cloud providers. In Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific (pp. 103-110). IEEE. doi:10.1109/APSCC.2009.5394134

Cruz, J., & Park, K. (2001). Towards communication-sensitive load balancing. In Distributed Computing Systems, 2001. 21st International Conference on. (pp. 731-734). IEEE. doi:10.1109/ICDSC.2001.919011

Fan, P., Chen, Z., Wang, J., Zheng, Z., & Lyu, M. R. (2012). Topology-aware deployment of scientific applications in cloud computing. In Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on (pp. 319-326). IEEE. doi:10.1109/CLOUD.2012.70

Ferreto, T. C., Netto, M. A., Calheiros, R. N., & De Rose, C. A. (2011). Server consolidation with migration control for virtualized data centers. *Future Generation Computer Systems*, *27*(8), 1027–1034. doi:10.1016/j.future.2011.04.016

Gupta, A., Kalé, L. V., Milojicic, D., Faraboschi, P., & Balle, S. M. (2013). HPC-Aware VM Placement in Infrastructure Clouds. In Cloud Engineering (IC2E), 2013 IEEE International Conference on (pp. 11-20). IEEE.

Gupta, A., Milojicic, D., & Kalé, L. V. (2012). Optimizing VM Placement for HPC in the Cloud. In *Proceedings of the 2012 workshop on Cloud services, federation, and the 8th open cirrus summit* (pp. 1-6). ACM. doi:10.1145/2378975.2378977

Hyser, C., Mckee, B., Gardner, R., & Watson, B. J. (2007). Autonomic virtual machine placement in the data center. Hewlett Packard Laboratories, Tech. Rep. HPL-2007-189, 2007-189.

Jung, G., Hiltunen, M. A., Joshi, K. R., Schlichting, R. D., & Pu, C. (2010). Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on (pp. 62-73). IEEE.

Kozuch, M. A., Ryan, M. P., Gass, R., Schlosser, S. W., O'Hallaron, D., Cipar, J., & Ganger, G. R. (2009). Tashi: location-aware cluster management. In *Proceedings of the 1st workshop on Automated control for datacenters and clouds* (pp. 43-48). ACM. doi:10.1145/1555271.1555282

Machida, F., Kim, D. S., Park, J. S., & Trivedi, K. S. (2008). Toward optimal virtual machine placement and rejuvenation scheduling in a virtualized data center. In Software Reliability Engineering Workshops, 2008. ISSRE Wksp 2008. IEEE International Conference on (pp. 1-3). IEEE. doi:10.1109/ISSREW.2008.5355515

Mann, V., Gupta, A., Dutta, P., Vishnoi, A., Bhattacharya, P., Poddar, R., & Iyer, A. (2012). Remedy: Network-aware steady state VM management for data centers. In NETWORKING 2012 (pp. 190-204). Springer Berlin Heidelberg.

Mann, V., Kumar, A., Dutta, P., & Kalyanaraman, S. (2011). VMFlow: leveraging VM mobility to reduce network power costs in data centers. In *NETWORKING 2011* (pp. 198–211). Springer Berlin Heidelberg. doi:10.1007/978-3-642-20757-0_16

Mogul, J. C., & Popa, L. (2012). What we talk about when we talk about cloud network performance. *Computer Communication Review*, *42*(5), 44–48. doi:10.1145/2378956.2378964

Nakada, H., Hirofuchi, T., Ogawa, H., & Itoh, S. (2009). Toward virtual machine packing optimization based on genetic algorithm. In Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living (pp. 651-654). Springer Berlin Heidelberg. doi:10.1007/978-3-642-02481-8_96

Rodrigues, H., Santos, J. R., Turner, Y., Soares, P., & Guedes, D. (2011). Gatekeeper: Supporting Bandwidth Guarantees for Multi-tenant Datacenter Networks. In WIOV.

Shieh, A., Kandula, S., Greenberg, A. G., Kim, C., & Saha, B. (2011). Sharing the Data Center Network. In NSDI.

Sonnek, J., Greensky, J., Reutiman, R., & Chandra, A. (2010). Starling: Minimizing communication overhead in virtualized computing platforms using decentralized affinity-aware migration. In Parallel Processing (ICPP), 2010 39th International Conference on (pp. 228-237). IEEE.

Stage, A., & Setzer, T. (2009). Network-aware migration control and scheduling of differentiated virtual machine workloads. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing* (pp. 9-14). IEEE Computer Society. doi:10.1109/CLOUD.2009.5071527

Tsakalozos, K., Roussopoulos, M., & Delis, A. (2011). VM placement in non-homogeneous IaaS-Clouds. In *Service-Oriented Computing* (pp. 172–187). Springer Berlin Heidelberg. doi:10.1007/978-3-642-25535-9_12

Wang, S. H., Huang, P. P. W., Wen, C. H. P., & Wang, L. C. (2014). EQVMP: Energy-efficient and QoS-aware virtual machine placement for software defined datacenter networks. In Information Networking (ICOIN), 2014 International Conference on (pp. 220-225). IEEE.

Xu, J., & Fortes, J. (2011). A multi-objective approach to virtual machine management in datacenters. In *Proceedings of the 8th ACM international conference on Autonomic computing* (pp. 225-234). ACM. doi:10.1145/1998582.1998636

Zhang, Y., Su, A. J., & Jiang, G. (2011). Understanding data center network architectures in virtualized environments: A view from multi-tier applications. *Computer Networks*, *55*(9), 2196–2208. doi:10.1016/j.comnet.2011.03.001

KEY TERMS AND DEFINITIONS

Cloud Computing: A computing paradigm that enables on-demand, ubiquitous, convenient network access to a shared pool of configurable and highly reliable computing resources (such as servers, storage, networks, platforms, applications, and services) that can be readily provisioned and released with minimal management effort or service provider interaction.

Data Center: An infrastructure or facility (either physical or virtual) that accommodates servers, storage devices, networking systems, power and cooling systems, and other associated IT resources that facilitates the storing, processing, and serving of large amounts of mission-critical data to the users.

Network Topology: Physical or logical arrangement of various computing and communication elements (nodes such as servers, storage devices, network switches/routers, and network links). It defines how the nodes are interconnected with each other (physical topology); alternately, it defines how data is transmitted among the nodes (logical topology).

Virtual Machine: A software computer (emulation of physical machine) that is comprised of a set of specification and configuration files backed by the physical resources of a host machine and runs an operating system and applications. A Virtual Machine has virtual devices with similar functionality as the underlying physical devices having additional advantages in relation to manageability, security, and portability.

Virtualization: The creation, management, and termination of virtual version of a resource or device (such as computer hardware, storage device, operating system, or computer network) where the framework partitions the resource into one or more virtual execution environments.

VM Live Migration: The process of moving a running VM from one host machine to another with little downtime of the services hosted by the VM. It enables server maintenance, upgrade, and resource optimization without subjecting the service users to downtime.

VM Placement: The selection process that identifies of the most suitable physical machine during the VM deployment in a data center. During placement, hosts are ranked based on their resource conditions and the VM's resource requirements and additional deployment conditions. VM placement decisions also consider the placement objectives such as maximization of physical compute-network resource utilization, energy efficiency, and load balancing.