Journal of Network and Computer Applications 52 (2015) 11–25

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Review

A survey on virtual machine migration and server consolidation frameworks for cloud data centers

Raja Wasim Ahmad [a,*], Abdullah Gani [a], Siti Hafizah Ab. Hamid [a], Muhammad Shiraz [a], Abdullah Yousafzai [a], Feng Xia [b]

[a] Center for Mobile Cloud Computing (C4MCC), FSKTM, University of Malaya, Malaysia
[b] School of Software, Dalian University of Technology, China

ARTICLE INFO

Article history:
Received 21 August 2014
Received in revised form
17 December 2014
Accepted 10 February 2015
Available online 5 March 2015

Keywords:
Virtualization
Server consolidation
VM migration
Data center
QoS

ABSTRACT

Modern Cloud Data Centers exploit virtualization for efficient resource management to reduce cloud computational cost and energy budget. Virtualization empowered by virtual machine (VM) migration meets the ever increasing demands of dynamic workload by relocating VMs within Cloud Data Centers. VM migration helps successfully achieve various resource management objectives such as load balancing, power management, fault tolerance, and system maintenance. However, being resource-intensive, the VM migration process rigorously affects application performance unless attended by smart optimization methods. Furthermore, a Cloud Data Centre exploits server consolidation and DVFS methods to optimize energy consumption. This paper reviews state-of-the-art bandwidth optimization schemes, server consolidation frameworks, DVFS-enabled power optimization, and storage optimization methods over WAN links. Through a meticulous literature review of state-of-the-art live VM migration schemes, thematic taxonomies are proposed to categorize the reported literature. The critical aspects of virtual machine migration schemes are investigated through a comprehensive analysis of the existing schemes. The commonalties and differences among existing VM migration schemes are highlighted through a set of parameters derived from the literature. Finally, open research issues and trends in the VM migration domain that necessitate further consideration to develop optimal VM migration schemes are highlighted.

Contents

* Corresponding author. Tel.: +60 107891697; fax: +60 379579249.
E-mail addresses: wasimraja@siswa.um.edu.my (R.W. Ahmad), abdullah@um.edu.my (A. Gani), sitihafizah@um.edu.my (S.H.Ab. Hamid), muh_shiraz@yahoo.com (M. Shiraz), abdullahyousafzai@siswa.um.edu.my (A. Yousafzai), f.xia@acm.org (F. Xia).

http://dx.doi.org/10.1016/j.jnca.2015.02.002
1084-8045/

1. Introduction

Inefficient resource management policies poorly exploit system resources within Cloud Data Centers (CDC). CDCs are normally over-provisioned to assure high service availability and application quality of service (QoS) (Beloglazov and Buyya, 2013). On an average, 30% of cloud servers exploit 10–15% of resource capacity most of the time (Uddin et al., 2013). Circumscribed resource utilization results in astonishingly high CDC operational cost and energy usage. Google Data Centers are estimated to have consumed 260 million Watts of energy (0.01% of the world's energy) in 2013 (Koomey, 2011; Server (2013)). Having adequate processing power, today's CDC servers host plenty of applications for efficient resource management to optimize energy consumption. To scale a CDC, virtualization exploits VM migration to relocate VMs both within and across CDCs to achieve various resource management objectives like server maintenance provisioning, power reduction, load balancing, and fault tolerance (Koomey, 2008; Voorsluys et al., 2009). Further, a CDC provides the foundation for cloud computing (CloudCom) and is crucial to its economic growth.

CloudCom is a distributed computing model that offers highly reliable and scalable services to subscribers. Emerging technologies, including Vehicular Adhoc Network (VANET) (Whaiduzzaman et al., 2013), Wireless Sensor Networks (WSN), and mobile computing applications (e.g., online games, bio-medical image processing, etc.) (Khan et al., 2013) use cloud-hosted services (e.g. infrastructure as a service (IaaS), platform as a service (PaaS) (Kremer; Mell and Grance, 2011), and Software as a service (SaaS)) to improve and extend functionalities. For instance, Vehicular cloud computing (VCC) merges VANET and CloudCom to assist vehicle drivers to minimize traffic congestion, accidents, and travel time (Huang et al., 2014; Whaiduzzaman et al., 2013). Similarly, a sensor cloud merges WSN and CloudCom to improve remote healthcare, vehicular transport systems (VTS), and environmental monitoring (Kim et al., 2014; Wang and Fan, 2014, Whaiduzzamanet al., 2013) by exploiting cloud services. Virtualization technology, the backbone of CloudCom, proactively offers scalable services to customers

Virtualization employs a hypervisor to proficiently manage several VMs running on a single physical server and to efficiently utilize cloud resources (Barham et al., 2003, Bugnion et al., 2012; Tao et al., 2012; Younge et al., 2011). However, co-hosting multiple VMs degrades application performance due to high resource contention (Asberg et al., 2011; Habib, 2008; Hu et al., 2013; Nathan et al., 2013; Younge et al., 2011). To improve application performance, the migration daemon migrates VM(s) to a resource-rich server in order to reduce the degree of resource contention (Jeong et al., 2013; Shuja et al., 2012; Mishra and Jaiswal, 2012; Moura Silva et al., 2007; Pop et al., 2012; Yao et al., 2014). However, since VM migration is a resource-intensive process, application performance is significantly affected during migration (Barhamet al., 2003; Clark et al., 2005; Xu et al., 2014). Bandwidth optimization techniques such as deduplication, compression, write-throttling, and dynamic rate-limiting optimize bandwidth utilization efficiency to enhance application performance (Deshpande et al., 2012; Gerofi et al., 2011, Hirofuchi et al., 2010; Sahni and Varma, 2012; Svärd et al., 2011). Furthermore, VM migration techniques migrate VMs either within LAN or WAN boundaries. However, while VMs migrate over WAN links, the migration daemon migrates storage besides VM memory. To optimize power consumption, VM migration technology uses server consolidation frameworks to switch off unnecessary servers (Deshpande et al., 2012).

This paper comprehensively reviews current VM migration schemes in cloud data centers and identifies challenges with migrating VMs across CDCs. The issues are thoroughly investigated while consolidating servers, and optimizing network bandwidth, storage, and dynamic voltage frequency scaling (DVFS)-enabled power consumption. We meticulously review VM migration schemes and underline their strengths, weaknesses, and issues requiring further research. Novel thematic taxonomies for VM migration approaches for server consolidation and bandwidth optimization are proposed to classify existing literature. The critical aspects and significant features of existing VM migration techniques are inspected through qualitative and quantitative enquiries. We drive critical parameters from the literature to compare VM migration schemes for server consolidation, bandwidth, storage, and DVFS-enabled power optimizations methods. Finally, open research issues and trends in the VM migration domain necessitating further exploration to develop optimal techniques for VM migration in cloud data centers are discussed. The main contribution of this article lies in the categorization of frameworks/schemes based on thematic taxonomies, analysis of existing migration schemes by discussing implications and critical aspects, identifying issues in existing solutions, and highlighting recent trends in the VM migration domain.

The rest of this paper is structured as follows. Section 2 discusses CloudCom, VM migration, DVFS technology, and server consolidation method. Section 3 presents a thematic taxonomy for the classification of server consolidation frameworks, existing server consolidation frameworks, and comparisons of existing frameworks based on parameters selected from literature. Section 4 presents a thematic taxonomy on bandwidth optimization schemes, and discusses state-of-the-art bandwidth, storage, and DVFS-enabled power optimization, followed by a detailed discussion on comparisons of existing schemes. Section 5 briefly discusses the research issues and trends in the VM migration domain. Section 6 concludes the paper with a discussion on potential future research directions.

2. Background

This section briefly overviews cloud computing, virtual machine migration, server consolidation, and DVFS enabled VM migration process.

2.1. Cloud computing

A Data Centre (DC) consists of interconnected servers organized into racks that represent a tree-based model for efficient resource management (Mahajan and Singh, 2013). A DC provides the foundation for implementing CloudCom services to offer subscribers elastic services (Armbrust et al., 2010; Niranjan Mysore et al., 2009). DC operators offer cloud services (Mahajan and Singh, 2013) based on the “Pay as you go” service model, to drive professionals (e.g., researchers, businessmen, etc.) to profoundly exploit cloud services (Buyya et al., 2010). The SaaS service model

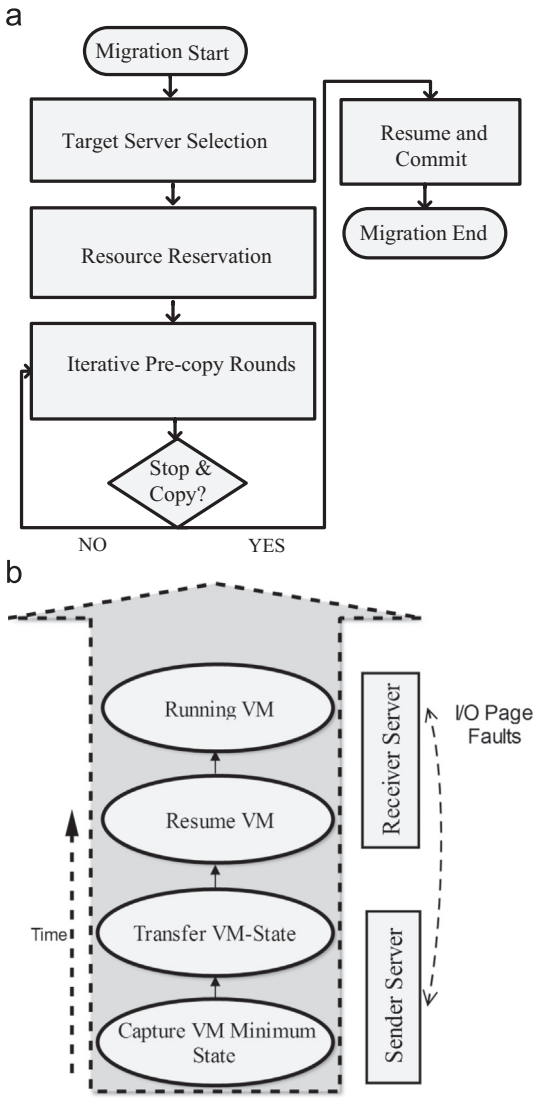

Fig. 1. VM migration phases, (a) pre-copy, (b) post-copy.

(Mell and Grance, 2011) offers cloud-hosted applications, including Google Drive and Google Docs to customers. Similarly, the PaaS service model delivers tools and services like Google's App Engine without the need for customers to manage the assigned resources (Khan et al., 2014). Furthermore, the IaaS service model (e.g., Amazon EC2) offers hardware resources such as storage and computational devices for remote storage and processing (Mahajan and Singh, 2013; Xing and Zhan, 2012). However, DC-based applications consume enormous amounts of energy that surge total DC operational cost and carbon footprint (Mell and Grance, 2011).

Inefficient resource utilization within the cloud significantly escalates power consumption by CDC equipment while increasing operational cost and carbon emission rates (Marston et al., 2011). Regarding CDC, rapidly growing energy demand of information and communication technology (ICT) equipment is a major cause of global warming due to the exponential CO_2 emissions growth (Ali et al., 2013). Virtualizing (KREMER; Wang et al., 2010c) ICT equipment reduces CDC energy demand through efficient resource utilization (Webb, 2008). Virtualization runs numerous VMs on a single hardware resource while respecting the application's privacy by implementing strict isolation policies (server virtualization) (Shuja et al., 2012). Resources including CPU, memory, network bandwidth, and system cache are dynamically provisioned to a VM

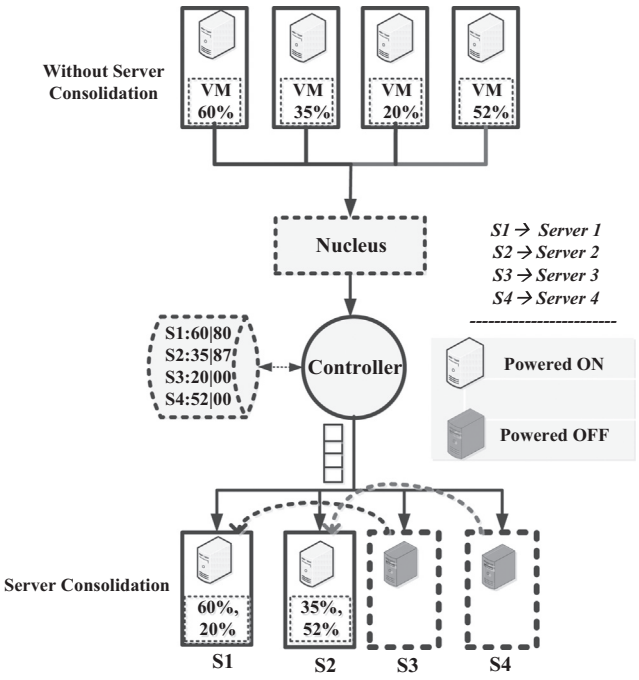

Fig. 2. An abstract-overview of server consolidation.

based on the performance requirements of VM-hosted workloads (Ali and Jing, 2013; Armbrust et al., 2010; Niranjan et al., 2009; Xing and Zhan, 2012). Virtualization exploits VM migration technology to consolidate server workloads to improve the CDC's energy consumption (due to improved system resource utilization) (Webb, 2008).

2.2. Virtual machine migration

VM migration practice exploits live or non-live patterns to move a complete virtual server across physical machines to successfully attain load balancing (Shiraz et al., 2013b, Yao et al., 2014), power efficiency (Dong et al., 2013; Mishra et al., 2012; Shrivastava et al., 2011), fault tolerance (Nagarajan et al., 2007, NGUYEN et al., 2013; Thein and Park, 2009), and system maintenance (Asberg et al., 2011; Liu et al., 2011; Wu et al., 2013; Zhou et al., 2013) within a DC. A live VM migration pattern guarantees continuous service provisioning to the hosted applications during the VM memory transfer process (Kapil et al., 2013; Shribman and Hudzia, 2013), whereas non-live VM migration (Kozuch and Satyanarayanan, 2002; Wang et al., 2010a) suspends application execution prior to memory image transfer (Aikema et al., 2012; Glazer and Tropper, 1993; Kozuch and Satyanarayanan, 2002, Kozuch et al., 2002; Milojičić et al., 2000). Live VM migration is classified into pre-copy or post-copy methods. A pre-copy VM migration pattern (Clark et al., 2005; Wang et al., 2012) transfers complete memory image before resuming VM at the target server. Alternatively, a post-copy VM migration approach captures and transfers the VM's minimum system state (e.g., processor registers, I/O device's states, etc.) to the target server before the VM resume phase.

The life-cycle of pre-copy and post-copy migration patterns is depicted in Fig. 1. The pre-copy approach follows target server selection, resource reservation (at the target server), iterative pre-copy rounds, stop and copy, commit and resume phase (Fig. 1 (a)). In the pre-copy stages, the migration manager selects an appropriate server (with sufficient resources), reserves required system resources for VM, and iteratively copies dirtied memory pages to the target server until a stop and copy (final round) phase is

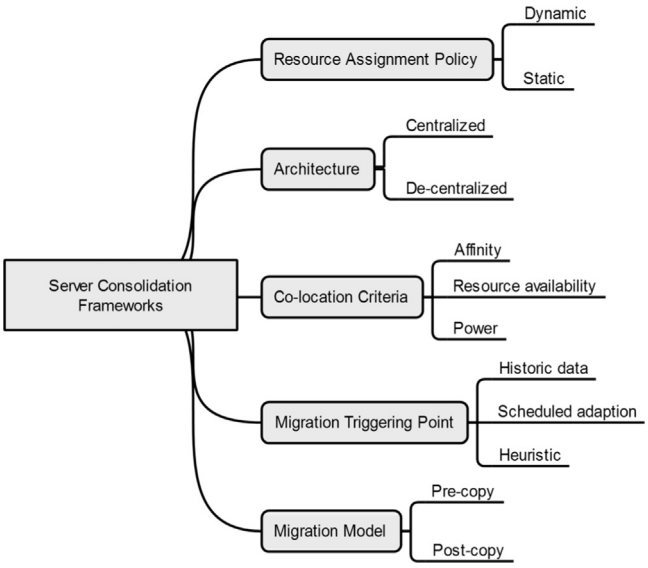

Fig. 3. Server consolidation frameworks taxonomy.

triggered to halt and transfer remaining dirty pages. Once VMs are synchronized, the commit and resume phase starts the VM at the target server and terminates the connection with the source server. The post-copy method (Clark et al., 2005; Hines et al., 2009; Shribman and Hudzia, 2013) starts with capturing the VM's minimum state (e.g., CPU register's state, I/O state), transferring the captured state, creating and resuming VM at the target server, and fetching memory pages from the source server until both VMs are completely synchronized (Fig. 1 (b)).

2.3. Server consolidation

Resource underutilization leads to substantial rise in DC operational cost and power consumption (Clark et al., 2005). Server consolidation (Ferreto et al., 2011; Wood et al., 2009) is among the methods that improve DC resource utilization efficiency by multiplexing several workloads on a few servers to reduce hardware cost, carbon footprint, and power consumption within a DC (Vogels, 2008). Server consolidation helps the DC operator avoid server sprawls by eliminating underutilized servers (Pop et al., 2012). Fig. 2 depicts an overview of server consolidation wherein the workload from an under-loaded server is shifted to underutilized servers in order to increase resource usage efficiency. Prior to server consolidation, servers use a fraction of their total capacity, e.g. server 1 uses 60% of its peak capacity. The migration controller accesses the servers' usage statistics (CPU, memory, network, etc.) using a nucleus module (a stub program) running within the physical servers. The controller triggers VM migration from under-loaded servers to underutilized servers to switch off the former and optimize the CDC power budget (Liu et al., 2013).

2.4. Dynamic voltage frequency scaling

DVFS decreases voltage or frequency scales to adjust the power a system consumes. However, decreasing clock frequency only applies to CPU power (Verma et al., 2008; Von Laszewski et al., 2009). Most DVFS-based power reduction schemes are based on advanced configuration and power interface (ACPI) (Shuja et al., 2014) specifications. ACPI defines four states for a server, including working (G_0), sleeping (G_1), soft-off (G_2), and mechanical-off (G_3). Working and mechanical-off states represent switched-on and switched-off states, respectively (Verma et al., 2008). Mechanical-off state means the system is not in running state, and OS must be restarted in order to resume the working state. Sleeping and soft-off are further divided into sub-states based on how the components are powered. During soft-state, computer components consume minimal power, and both user and system mode codes are not running. Also, the system consumes minimal power while in sleeping state as it appears switched-off. Compared to soft-state when a system is in sleeping state, there is no need to restart OS to switch to working state (Intel Corporation H-PC, 2005;Shuja et al., 2012).

3. Server consolidation

This section presents the taxonomy on server consolidation frameworks, a review of state-of-the-art server consolidation frameworks, and a comparison of existing frameworks based on parameters selected from literature.

3.1. Taxonomy of server consolidation frameworks

This section highlights and presents a thematic taxonomy for the classification of current server consolidation frameworks. Server consolidation frameworks are categorized based on five common characteristics among server consolidation frameworks, including Resource assignment policy, Architecture, Co-location criteria, Migration triggering point, and Migration model (Fig. 3).

Resource assignment policy describes the method chosen to grant resources to VM within a DC. Resource assignment policy attributes are either static or dynamic. Dynamic attributes proactively re-configure the VM based on workload demand. The static server consolidation method pre-assigns maximum resources to the VM upon its creation. The architecture parameter attribute describes server consolidation framework design. A decentralized architecture makes DC scalable as there is no risk of a single failure point. However, centralized server consolidation frameworks are prone to single failure point and are unreliable. The co-location criteria attribute defines the criterion opted to co-host multiple VMs within a server. VM co-location criteria can be defined in terms of shared memory, communication bandwidth between VMs, power efficiency, and sufficient resource availability. The migration triggering point attribute labels the condition (When to migrate?) to decide on the appropriate time to migrate a VM. The migration triggering point attributes include heuristic-based migration triggering, historical data-based migration, intelligent-learned QoS, or scheduled adoption. A migration model describes the migration pattern chosen to migrate the VM among servers. During server consolidation, VMs are migrated either using pre-copy migration pattern or post copy method.

3.2. A review of server consolidation frameworks

The VM placement problem has been studied (Kakadia et al., 2013) to identify VM clusters based on communication cost between VMs in order to improve the performance of I/O and non-I/O intensive applications. Communication cost is a factor of communication rate and end-to-end network delay. The proposed framework exploits an access matrix, representing communication cost between different VMs (calculated offline); to identify bandwidth intensive VMs in order to form a VM cluster. The cost tree representing the communication cost between VMs serves to optimally place VMs according to the communication distance between VMs while traversed in bottom-up fashion. The framework suppresses unnecessary VM migrations to reduce the likelihood of SLA violation. However, the framework lacks in considering the effect of CPU and memory-intensive workloads during VM placement. For instance, co-hosting memory-intensive workloads harms system performance due to resource contention.

To avoid SLA violation due to aggressive server consolidation, an adaptive threshold-based dynamic server consolidation framework was proposed (Beloglazov and Buyya, 2010a). The framework uses CPU usage distribution and inverse cumulative probability functions for t-distribution to calculate lower and upper CPU utilization thresholds, which are adjusted based on workload nature. VMs are selected for migration according to the hosted workload behavior against dynamic upper and lower thresholds. To optimally place VMs, a bin packing model is exploited to sort the servers (based on remaining CPU capacity) using a modified best fit decreasing method in offline mode. Furthermore, the proposed framework considers CPU resource usage and power efficiency to place VM (s) on the selected servers. The power model used estimates power usage based on CPU usage, the nature of workload hosted within the VM, and power consumption when the server is idle. However, the proposed model lacks in considering the effect of inter-VM bandwidth behavior during workload consolidation.

Page sharing models and a sharing-aware VM co-hosting framework were proposed in Sindelar et al. (2011) to highlight the significance of VM page sharing in reducing memory footprint. The proposed framework exploited tree-based and cluster tree models to emphasize the page sharing potential among VMs by using OS type, OS versions, or software libraries (common) as hint functions. Additionally, the proposed framework has optimized the VM maximization problem that defines the most profitable set of VMs that can be co-hosted using dynamic server consolidation. Moreover, to formulate VM packing, the greedy approach is used to reduce the number of servers while respecting server capacity. As both these problems are NP-hard, the time complexity for solving these problems is very high. Also, the page sharing models and sharing-aware VM co-location based framework only consider memory sharing capability as a VM placement parameter, which is insufficient and can lead to aggressive VM migrations.

Memory buddies (Wood et al., 2009) are a memory sharing-aware server consolidation framework that uses a fingerprinting system (hashing (Hu et al., 2013) and balloon filter (Medina and García, 2014)) to efficiently determine the sharing potential of different VMs located within heterogeneous servers. Memory buddies have a smart module embedded called “Nucleus” within VMM to calculate and transfer (to control plane module) memory fingerprints. The control plane module is responsible for VM placement and hotspot detection. To consolidate VMs, memory buddies selects candidate servers by analyzing the memory usage level with a lower set threshold level. Upon server selection, sorted VMs are subsequently migrated to sorted servers and are placed based on sufficient resource availability besides high sharing potential. However, this framework does not address the method of determining lower threshold values. Memory buddies has focused memory usage to trigger the migration process, which alone is insufficient; as VMs similarity is also an important factor that needs to be considered.

A reactive post-copy enabled server consolidation framework is discussed in Hirofuchi et al. (2011) to optimize VM placement during random load changes. The proposed framework consists of three modules: load monitor, relocation planner, and VM controller. The load monitor collects system resource statistics such as CPU usage, disk I/O usage, network I/O, etc., to periodically update the database. The relocation planner decides whether the shared server is overloaded or under-loaded based on resource usage statistics analysis (via the load monitor) prior to calculating the relocation plans. The VM controller is responsible for executing migration and suspending/resuming the servers based on the relocation planner's command. Alternatively, based on resource usage statistics, servers are switched on and off in the DC layer to handle the load. However, this framework focused only on CPU resources for deciding VM migration, and it does not consider power efficiency, workload nature and other resources (e.g., memory similarity between VMs).

The problem of VM consolidation was empirically examined in Beloglazov and Buyya (2010c) with focus on guaranteeing SLA and energy efficiency of the entire data center. The proposed framework handled the trade-offs between system performance and energy efficiency by eliminating hotspots detected within a server. A hotspot is detected when a server becomes heat-up due to resource-overloading that affects hosted application's QoS. Several metrics considered in the study include current resource utilization, communication bandwidth, and system temperature for server consolidation. In the proposed system architecture, a local manger (residing on the server layer) chooses and suggests (to the global manager in an upper level in a tree) VM migration whenever system resource usage reaches a peak resource utilization level, the temperature exceeds a threshold value, or VM has extensive network communication with another node, to minimize the performance-energy trade-off. The server placement problem is resolved using semi-online multidimensional bin packing. However, the authors considered CPU resource usage as migration triggering point but ignored other factors like device temperature, affinity, and network traffic.

In Mi et al. (2010), an online self-reconfiguration based relocation framework was proposed to handle dynamic cloud-hosted workload demand. Brown's quadratic exponential smoothing formula is employed to predict future demand of the running workload. Furthermore, genetic optimization is used to efficiently find an optimal reconfiguration policy. The proposed method, called genetic algorithm-based approach (GABA), finds the optimum reconfiguration for a set of VMs. By considering the count and locations of VMs, the number of PM (physical machines) is reduced significantly, and the algorithm converges within reasonable time. The advantages of GABA are the flexibility to self-reconfigure and efficient online searching to optimize the solution through diverse and complex conceivable reconfiguration spaces. However, GABA considers only one resource (CPU) for VM resource capacity calculation and migration-decision making. Others factors including the server's component temperature, affinity among VMs, and network traffic patterns are not considered for decision-making during VM migration. These parameters affect the application QoS as well as result in migration stability.

A controlled VM migration system is discussed in Ferreto et al. (2011) to use linear programing (LP) formulation and heuristics to prioritize stable migration during server consolidation. LP formulation opts for physical server minimization as an objective function while considering the host's peak resource usage and each VM mapping-guarantee as constraints for both static and dynamic consolidation. To avoid SLA violation, the proposed framework does not migrate VMs with steady usage workload; however, VMs with variable capacity can be migrated to reduce the number of physical servers. The reason why steady workload is prioritized is that varying VM capacity does not affect it. Furthermore, selected VMs are packed on a few servers (already sorted) using best fit decreasing, worst fit decreasing, and almost worst fit decreasing heuristics in offline mode when the overloading/under-loading condition is met. For static consolidation, mapping is performed only once based on workload peak resource demand compared to dynamic consolidation where mapping is done adaptively.

To optimize server consolidation frameworks, a SLA violation decision algorithm (Cao and Dong, 2014) was proposed to identify overloaded (SLA violation) hosts based on interquartile range (IQR), local regression (LR), robust local regression (LRR), and median absolute deviation (MAD) heuristics. For minimum power (MP) policy implementation, it selects a server for VM placement while considering the least energy consumption as an objective parameter. The proposed framework also decides SLA violations based on CPU

Table 1
Server consolidation framework's comparison.

Framework (Ref.)	Resource assignment policy	Architecture	Co-location criteria	SLA violation	Migration triggering point	Migration model
Net-aware Kakadia et al., 2013	DYNAMIC	CENTRALIZED	Affinity/IVMCM	YES	AGH	N/A
Adapt-threshold (Beloglazov and Buyya, 2010a)	DYNAMIC	DE-CENTRALIZED	Sufficient power & CPU resource	YES	CHB	N/A
Memory-share (Sindelar et al., 2011)	DYNAMIC	CENTRALIZED	Shared memory	NO	BS	N/A
Memory-buddies (Wood et al., 2009)	DYNAMIC	CENTRALIZED	Shared memory	NO	CHB	N/A
Reactive-cons(Hirofuchi et al., 2011)	DYNAMIC	CENTRALIZED	Sufficient resources	NO	AGH	Post-copy
Energy-thermal (Beloglazov and Buyya, 2010c)	DYNAMIC	DE-CENTRALIZED	Affinity/IVMCM	YES	CHB	N/A
Self-reconfig.Mi et al., 2010	DYNAMIC	CENTRALIZED	N/A	NO	AGH	N/A
LP-formulation (Ferreto et al., 2011)	STATIC/ DYNAMIC	N/A	N/A	YES	CHB	N/A
Min-Ener(Cao and Dong, 2014)	DYNAMIC	N/A	Power efficiency	YES	AGH	N/A
MECS(Huang et al., 2013)	DYNAMIC	CENTRALIZED	CPU and memory	YES	CHB	pre-copy

CHB: Heuristic based approach QB: Adaptive learned QoS based threshold AGH: Analysis of gathered Historic data BS: Scheduled Adaption, MM: Migration Method IVMCM: inter VM communication.

utilization statistics, as the CPU consumes a major share of energy within a DC. During the initial migration round, potential servers leading to SLA violation are identified. In the next round, VMs from these selected servers are iteratively migrated until all servers come out of the SLA violation zone. Similarly, to migrate VMs, it selects VMs with the highest CPU usage in the VM-to-migrate list to migrate them on the energy-efficient server. A VM-to-migrate list names all VMs that are migration candidates. Subsequently, the remaining VMs are migrated until a non-critical situation is attained.

A migration-based elastic consolidation framework (MECS) (Huang et al., 2013) consisting of four modules, including node agent, system monitor, resource demand predictor, and elastic scheduler, considers the dynamic workload nature and VM migration overhead while consolidating workloads. The system monitor and node agent supervise system resource usage to be exploited by the resource demand predictor modules for predicting CPU and memory usage by the application during the next “k” time interval in online mode. “K” is an arbitrary integer set that represents the prediction-usages of the system resources (e.g., CPU, memory). The resource conflict prediction phase of MECS identifies a set of VMs that cannot participate in any VM migration activity (using an ARIMA resource predictor) to avoid SLA violation. Also, in the resource consolidation phase, MECS sorts VMs and investigates the feasibility of VM migration (available in to-be-migrated list) on available (under-loaded server) hosts while eliminating chances of SLA violation prior to VM placement on sorted physical machines (sorted based on capacity). The critical aspect is that this framework does not consider inter-VM communication cost while executing server consolidation VM placement.

3.3. Comparison of server consolidation frameworks

Table 1 compares server consolidation frameworks based on resource assignment policy, architecture, co-location criteria, SLA violation, migration triggering point, and migration model parameters selected from the literature.

The resource assignment policy attributes describe the method selected to assign system resources to VMs. These attributes are either static (Ferreto et al., 2011) or dynamic (Beloglazov and Buyya, 2010c; Mi et al., 2010; Wood et al., 2009). Dynamic resource assignment policy attributes (Beloglazov and Buyya, 2010c; Miet al., 2010; Wood et al., 2009) follow “resource on demand” mode to assign resources to the hosted VMs. A dynamic assignment policy (Beloglazov and Buyya, 2010c, 2013; Hirofuchi et al., 2011, Kakadia et al., 2013; Mi et al., 2010) evolves by calculating the workload's resource demands for each VM to re-configure it. Alternatively, a static (Ferreto et al., 2011) resource assignment policy configures VMs according to the workload's peak resource demand in a single step (Ferreto et al., 2011). However, static assignment policies bind VMs to using only designated resources. Static attributes lead to application QoS degradation if the hosted application's (dynamic workloads) resource demand grows substantially to satisfy SLA (Ferreto et al., 2011). Furthermore, static resource assignment exploits cloud resources inefficiently as resources remain idle most of the time. However, when using static resource assignment (Ferreto et al., 2011), fewer VMs can be packed onto one server compared to dynamic assignment (Nakada et al., 2009). Consequently, energy efficiency is lower for static resource assignment than dynamic assignment (Ferreto et al., 2011, Kakadia et al., 2013).

The attributes of server consolidation architecture describe the server consolidation framework design in order to achieve energy efficiency within a DC (Ferretoet al., 2011). Server consolidation framework architecture is categorized as either centralized (Hirofuchiet al., 2011, Kakadiaet al., 2013, Mi et al., 2010, Sindelar et al. Sitaraman, 2011, Woodet al., 2009) or decentralized (Beloglazov and Buyya, 2010a, 2010c, 2013). Centralized architectural design is unreliable due to a single point of failure (Ferreto et al., 2011). Server consolidation frameworks exploit a consolidation scheduler (Mi et al., 2010) to collect resource usage statistics using a nucleus (a stub process) embedded within each host's hypervisor. Centralized (Hirofuchi et al., 2011) architecture-based framework design uses a control plan layer to process the gathered resource usage statistics (Kakadia et al., 2013, Mi et al., 2010, Wood et al., 2009). However, if the control plan is compromised, availability and reliability are severely affected. In contrast, decentralized architecture (Beloglazov and Buyya, 2010a, 2010c, 2013) uses a global controller to control a set of servers (usually a cluster) within a DC. Similarly, a local controller manages a set of VMs residing on the same server by exploiting nucleus-generated resource usage statistics. The local manager reports the server's health condition (resource usage and system components heat) to the global controller/manager to migrate VM(s) for efficient resource utilization (Beloglazov and Buyya, 2010a, 2010c, 2013).

The migration triggering point attributes define “when to migrate” a VM to consolidate servers. An overloaded server affects residing and co-hosted applications' performance; thus, optimal migration triggering point selection is crucial (Beloglazov and Buyya, 2013; Mishra et al., 2012). Several server consolidation frameworks reactively trigger VM migration using a predefined threshold (resource utilization level) (Beloglazov and Buyya, 2010a, 2010b, 2010c; Ferreto et al., 2011; Wood et al., 2009), whereas some frameworks (Hirofuchi et al., 2011; Mi et al., 2010) are proactive and decide on migration after analyzing workload behavior for a specified time period. Server consolidation frameworks select migration triggering points based on heuristic methods (Beloglazov and Buyya, 2010a, 2010b, 2010c, Ferreto et al., 2011; Wood et al., 2009),

historical data analysis (Hirofuchi et al., 2011; Kakadia et al., 2013; Mi et al., 2010), learned-intelligent QoS-based threshold (Beloglazov and Buyya, 2013), and scheduled adoption (Sindelar et al., 2011). Deciding on migration triggering based on heuristics approaches (Beloglazov and Buyya, 2010b; Ferreto et al., 2011; Huang et al., 2013) leads to unnecessary migration if not managed properly. Besides, numerous frameworks (Hirofuchi et al. 2011, Kakadia et al., 2013, Mi etal., 2010) exploit resource usage history curves (by VMs) and prediction models (Beloglazov and Buyya, 2010a, 2010c, 2013; Ferretoet al., 2011, Kakadia et al., 2013) to optimize migration triggering and improve application QoS. However, a limited set of frameworks (Beloglazov and Buyya, 2013; Sindelar et al. 2011) considered workload-independent QoS-based threshold methods for strict SLA violation avoidance and controlled migration (Beloglazov and Buyya, 2013). Some frameworks (Sindelar et al., 2011) follow a predefined schedule to migrate VMs, e.g. migrating a VM at midnight/12.00AM.

The co-location criteria parameter defines criteria for co-hosting VMs to optimize power consumption. The co-location criteria attributes include shared memory (Sindelar et al., 2011; Wood et al., 2009), communication bandwidth between VMs (Beloglazov and Buyya, 2010c; Kakadia et al., 2013), power efficiency (Beloglazov and Buyya, 2010a, 2010b), and system resource usage statistics. Shared memory attributes co-host VMs based on VMs' memory sharing capacity (e.g., similar workload, OS, hypervisor). By using the shared memory (Sindelar et al., 2011; Wood etal., 2009) attribute, co-located VMs communicate with each other efficiently and memory footprint is reduced. Numerous server consolidation frameworks consider a memory-sharing capability to co-locate VMs (Sindelar et al., 2011; Wood etal., 2009). Furthermore, high memory sharing decreases physical servers' memory size (Sindelar et al., 2011; Wood et al., 2009). Likewise, frequently communicating VMs (Beloglazov and Buyya, 2010c; Kakadia et al., 2013) can be co-hosted to relieve the network link. Co-hosting frequently communicating VMs significantly reduces the probability of SLA violation due to shortened communication distances (Beloglazov and Buyya, 2010c; Hemalatha, 2013; Kakadia et al., 2013). VMs with lower power demand are co-hosted to reduce the possibility of system heat-up. System heat-up issues affect durability, stability, and availability (Beloglazov and Buyya, 2010a, 2010b). Co-location criteria attributes affect the SLA violation parameter attributes. Moreover, during server consolidation, the workload characteristics affect application performance due to memory sharing (Beloglazov and Buyya, 2013).

VM migration is an expensive operation as it significantly exploits cloud resources for transferring VM memory pages across servers (Hirofuchi etal., 2011; Mi et al. 2010;Sahni and Varma, 2012; Sindelar et al., 2011, Wood et al., 2009). A number of server consolidation frameworks (Beloglazov and Buyya, 2010a, 2010b, 2010c, 2013; Ferretoet al., 2011, Kakadia et al., 2013) consider SLA violation by (a) optimizing the threshold (migration triggering point) using workload-independent QoS-based threshold (Beloglazov and Buyya, 2013); (b) injecting control parameters inside the consolidation scheduler (Beloglazov and Buyya, 2013) to avoid unnecessary migration; and (c) accurately predicting resource demands of hosted workloads (Beloglazov and Buyya, 2010a, 2010b, 2010c, 2013; Kakadia et al., 2013). The migration model's attributes define the migration method selected by the server consolidation frameworks to migrate VMs across servers. Server consolidation frameworks (Beloglazov and Buyya, 2010a, 2010c, 2013; Ferreto et al., 2011; Kakadia et al., 2013) either consider pre-copy or post-copy migration patterns to move VMs across servers.

4. Virtual machine migration optimization

During VM migration process, memory pages are iteratively moved across servers while intensely using underlying system resources. VM migration within LAN exploits network-attached storage (NAS) architecture to share the storage between communicating servers, whereas migrating a VM across WAN boundaries requires migrating large-sized storage (in addition to VM memory) over intermittent links. Further, VM migration schemes use DVFS methods to optimize the power efficiency within a server as shown in Fig. 4. This section presents and compares VM migration optimization schemes that consider bandwidth, DVFS-enabled power, and storage optimization to lessen the side-effects of VM migration process.

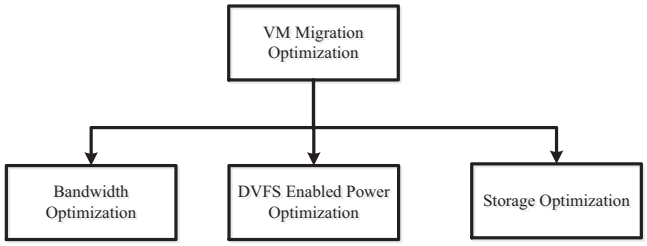

Fig. 4. VM migration schemes.

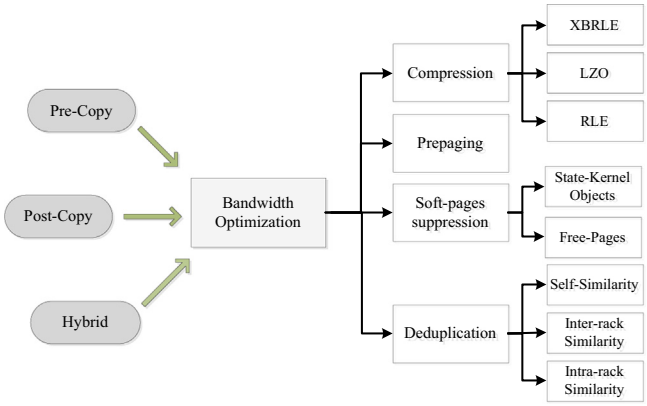

Fig. 5. Taxonomy of bandwidth optimization VM migration schemes.

4.1. Bandwidth optimization

This section presents a thematic taxonomy, review of existing schemes, and comparisons among bandwidth optimization schemes, to effectively utilize limited network capacity in order to improve application performance during the VM migration process.

4.1.1. Taxonomy of bandwidth optimization schemes

Live VM migration schemes exploit pre-copy, post-copy and hybrid patterns while effectively utilizing network bandwidth capacity. Existing schemes are classified based on optimization heuristics they exploit to efficiently utilize bandwidth capacity while migrating VMs across servers. The migration pattern optimizes bandwidth capacity by exploiting different methods as shown in Fig. 5.

Compression reduces VM memory size by using redundant features within the memory contents. Bandwidth optimization schemes use XBRLE, LZO, and RLE compression algorithms to compress/decompress memory contents. Pre-paging uses actively pushed memory pages as reference to predict the memory access pattern in order to transfer memory pages to the target server prior to page fault (post-copy case). Soft pages represent memory pages unnecessary for the system to work correctly after VM migration completion. Soft pages include state kernel objects and

free pages. Free pages are automatically produced at the target side when VMs are initialized. Alternatively, state kernel objects include a cache for storage blocks and cache for kernel resource managers that can be reproduced using disk data (e.g., NAS) and original objects (slab cache). Deduplication eliminates duplicate memory pages while migrating the VM memory image and exploits self-similarity, inter-rack similarity, and intra-rack VM similarity to optimize network bandwidth. Self-similarity deduplication eliminates duplicate memory pages within the VM memory. Alternatively, inter-rack deduplication suppresses the transfer of identical memory pages available within the rack. Similarly, intra-rack level deduplication identifies and avoids transfer of similar memory pages located within the servers of different racks.

4.1.2. Review of bandwidth optimization schemes

Inter-racked live migration (IRLM) (Deshpande et al., 2012) employs deduplication to improve bandwidth utilization efficiency while migrating a group of VMs. Deduplication identifies similar memory pages and transfers only one copy using any VM. The proposed framework exploits a controller to find similar memory pages both within and athwart servers using QEMU/KVM thread. Further, to identity memory pages at the rack level, an index server hosts memory page hashes (source side). The index server is inquired prior to memory page transfer to access the memory page status. For previously transferred memory pages, QEMU/KVM pushes a page identifier only during VM migration. The receiver module of the IRLM framework uses a controller to accept and deliver memory pages to legitimate target servers. However, the proposed framework is very complex and computationally expensive due to large calculations involved (e.g., 160 bit hash value). The acceptability of designed framework is limited to servers hosting identical VMs and workloads.

A detailed analysis of the execution log indicates that the same portion of VM RAM is often dirtied repeatedly in successive pre-copy iterations. Delta-based compression works excellently in such scenarios. The proposed scheme (Svärd et al., 2011) applies binary XOR-based RLE (XBRLE) delta compression to improve VM migration performance. A Delta page is derived by performing a XoR operation between current and previous dirty pages (in case of cache hit). For further improvement, the delta page is compressed using a light weight compression algorithm (RLE). The critical aspect of the proposed method is the demand for larger cache memory and CPU cycles for compressing memory-intensive applications. XBRLE assumes that co-hosted VMs host similar workload and OS, which is not always true. In another study (Zhang et al., 2010), hash-based fingerprints use VM self-similarity ratio prior to compressing pages using a lightweight compression (RLE) algorithm in order to enhance bandwidth utilization efficiency.

Sonic migration (Koto et al., 2012) scheme tracks and avoids the transfer of soft pages during the VM migration process. Soft pages include free pages and soft-kernel state objects, which are already available at the target server. Prior to triggering migration, a guest kernel conveys soft page addresses to the VMM. Sonic migration creates a shared memory between VMM and guest kernel to communicate efficiently with little intervention of CPU resource. Prior to stop-and-copy phase initiation, VMM generates a signal for VM to update the shared memory. Resultantly, VM generates a hyper call for the hypervisor to trigger the stop-and-copy phase. The proposed scheme has the advantage that it does not influence hosted applications' throughput by decreasing the total migration time. However, the proposed approach induces extra overhead on CPU and memory resources, affecting the application's QoS.

Recently, the cloning trend efficiently manages DC. Cloning technology requires the code and data between different hosts to be strictly synchronized. A hash-based fingerprinting approach, MV-motion, was presented (Zhang et al., 2013) to identify a similarity matrix among VM memory content. MV-motion generates metadata at both the sender and receiver servers, representing pages' hash values to avoid duplicate data transfer. Furthermore, the sender communicates to the receiver metadata used by the receiver to generate a list of memory pages already available at the receiver. The copy-on-write (COW) method ensures metadata consistency at the receiver host. The critical aspect of MV-motion is the time complexity (preparation phase) and poor performance for dissimilar workloads. Also, cloning-based technology occupies massive bandwidth capacity owing to synchronization. The heavy network bandwidth usage by the VM migration daemon affects the hosted and co-hosted applications due to bandwidth contention.

A lightweight extension of KVM was proposed (Hirofuchi and Nakada, 2010) for VM migration within LAN links. The proposed scheme has an additional special device driver within the guest OS to transparently handle “on-demand” memory access queries. This scheme progresses by transferring CPU registers and device states to the receiver host prior to VM memory content migration using QEMU. Based on migrated device states, the receiver uses a QEMU process to resume VM with the received state (at receiver). To service the application's I/O request, a kernel triggers a page fault handler process of the “VMEM” driver to pull the contents from the sender if memory pages are not already available. The VMEM driver is responsible for page fault handler implementation. Based on request, the VMEM module pulls and updates the shared memory at the receiver. In addition, QEMU also initiates a background thread to actively push memory pages (to the target). The advantage of VMEM-based migration is the freedom to work independently with no demand for a special driver to execute VM migration.

The proposed scheme in Sahni and Varma (2012) exploits a hybrid method (pre-copy, post-copy) to migrate VMs between servers communicating through Ethernet links. The entire VM migration process is divided into the preparation stage, down state, and resumes state. In preparation step, “access bit scanning” serves to identify VM working set. Access bit scanning identifies the most frequently accessed memory pages (called working sets). Next, the working set is transferred along the system minimum state to the destination. The third stage resumes the VM and pulls memory pages from the source server based on the application's I/O requests. For network resource optimization, Lempel–Ziv–Oberhumer (LZO) (Van Bockhaven, 2014) compresses memory pages prior to page transfer. The proposed approach has significantly improved application downtime, total migration duration, and total data transfer. Advantages include successful reduction of total migration time and downtime.

In Hines et al. (2009), an optimized post-copy VM migration scheme was proposed that exploits on-demand paging, active push, pre-paging, and dynamic self-ballooning optimizations to pre-fetch memory pages at the receiver host. Active push transfers memory pages to the target server and ensures that every page is sent exactly once from the source server. The proposed scheme significantly reduces I/O network page faults by growing bubbles around the current page fault (pre-paging) based on spatial locality heuristics. Similarly, dynamic self-ballooning does not transfer free pages to the target server to efficiently utilize available network bandwidth. However, the performance of this approach is severely affected by prediction accuracy (e.g., spatial locality) selected to transfer neighboring memory pages prior to page fault occurrence. Besides, growing bubbles around the pivot memory page to transfer neighboring memory pages does not always improve VM migration performance, especially when write-intensive applications are hosted within migrated VMs.

VMFlock (Al-Kiswany et al., 2011) is a distributed migration approach that exploits deduplication within a group of VMs hosting identical workloads to reduce memory size and improve network performance. Hashing and distributed content addressing helps to execute deduplication both within and across DC servers. Moreover, the authors introduced a novel approach design to boot VM at the destination rapidly by logging and prioritizing the transfer of booting-related blocks. VMFlock contains a VM profiler and VM launch pad to identify the VM memory area necessary to boot the VM image and launch the application, respectively. The VM launch pad asks the source server for any memory page required for booting that is not available at the receiver server due to slightly different boot start-up methods at various CDCs. However, VMFlock requires sufficient CPU cycles to complete the whole migration process and locate booting-related memory pages. VMFlock presents reasonably reduced migration time and duration.

A three-stage copy approach (Yin et al., 2014) was proposed to ensure that memory pages can be transferred at most twice during the complete migration process. To reduce total migration time, the proposed scheme ensures that memory page faults occur within a small part of dirty pages. Three-stage copy follows full memory copy, dirty bitmap, and dirty page copy phases during VM migration. The full memory copy phase moves all the memory from the source to the target server while continuously servicing running applications in parallel to dirty page logging. Throughout the dirty bitmap copy round, the migration controller suspends the source VM and copies the recorded dirty memory to the destination. The "dirty pages copy" round resumes the VM at the target server and exploits active push and on-demand paging methods to fetch memory pages from the source server. The proposed scheme is useful when considering automatic load balancing.

4.1.3. Comparison of bandwidth optimization schemes

Table 2 illustrates a qualitative comparison of VM migration schemes based on selected parameters to highlight commonalities and variances in existing bandwidth optimization schemes. Live VM migration schemes follow either pre-copy (Deshpande et al., 2012; Koto et al., 2012; Svärd et al., 2011), post-copy (Hines et al., 2009; Hirofuchi et al., 2010), or hybrid (Sahni and Varma, 2012, Yin et al., 2014) migration patterns to migrate VMs across servers. VM migration approaches use optimized network bandwidth so VM can cope with large data (up to several hundred gigabytes). Migration optimization exploits deduplication (Deshpande et al., 2012), compression (Svärd et al., 2011), fingerprinting (Zhang et al., 2013), and dynamic self-ballooning (Hines et al., 2009) to improve application and network performance. Moreover, an optimization method presents additional overhead on shared resources like CPU, memory, or cache while optimizing VM migration performance parameters such as downtime, total migration time, and application QoS. Many VM migration approaches have optimized application downtime (Gerofi et al., 2011; Hines et al., 2009; Svärd et al., 2011) and total migration duration (Deshpande et al., 2012, Gerofi et al., 2011; Hines et al., 2009; Koto et al., 2012; Svärd et al., 2011; Zhang et al., 2013) by employing optimization and avoiding aggressive migration termination (a case of pre-copy). However, only few schemes (Gerofi et al., 2011, Svärd et al., 2011) consider the application's QoS degradation while migrating VMs across servers within a DC.

Table 2
Bandwidth optimization scheme's comparison (Qualitative).

Scheme (Ref.)	Pattern			Optimization		Objective			Migration overhead	Migration purpose
	PRE	*POS*	*HYB*	Type	Method	*DT*	*MT*	*QoS*		
IR (Deshpande et al., 2012)	✓			N	Deduplication		✓		CPU, memory	Power-down rack
RLE (Svärd et al., 2011)	✓			N & A	Compression	✓	✓	✓	CPU ,cache	N/A
SON (Koto et al., 2012)	✓			N & A	Soft-Pages suppression		✓		CPU	N/A
CLO (Zhang et al., 2013)	✓			N	Hash-based fingerprinting		✓		CPU	N/A
MEM (Hirofuchi et al., 2010)		✓		N & A	LZO compression	✓	✓		CPU	Instantaneous relocation
LZO (Sahni and Varma, 2012)			✓	N & A	Background, On demand			✓	CPU, Cache	Mixed-mode workload
DBAL(Hines et al., 2009)		✓		N	Self-ballooning, compression	✓	✓		CPU	N/A
VM-flocks (Al-Kiswany et al., 2011)	✓			N & A	Deduplication		✓		Memory Cache	Scalable migration
three stage (Yin et al., 2014)			✓	N	N/A	✓	✓		N/A	Load balancing

N: Network A: Application QoS: Quality of Service N/A: Not available PRE: Pre-copy POS: Post-Copy HYB: Hybrid DT: Downtime MT: Migration Time.

Table 3
Bandwidth optimization scheme's comparison (Quantitative).

Scheme	Hypervisor	No. of VMsmigrated	Workload	Downtime	Migration time	Network traffic reduction-rate
IR (Deshpande et al., 2012)	KVM	6	Net-Perf UDP-STR	N/A	26% reduction	17%
RLE (Svärd et al., 2011)	KVM	1	LM-Bench	0.29 s	65 s	80%
SON (Koto et al., 2012)	Xen	1	Post Mask	N/A	68% reduction	N/A
CLO (Zhang et al., 2013)	Xen	1	Hadoop-Load	N/A	16–53% reduction	29%
MEM (Hirofuchi et al., 2010)	KVM	1	Spectweb2005 Bank	3 s	Pre-copy:65 s Post-copy:10 s	N/A
LZO (Sahni and Varma, 2012)	KVM	1	NA	N/A	N/A	N/A
DBAL (Hines et al., 2009)	Xen	1	SpecWeb2005 Net-perf	Pre-copy: 1200 ms Post-copy:600 ms	Pre-copy:8 s Post-copy:3.6 s	7–13%
VM-flocks (Al-Kiswany et al., 2011)	KVM	1	Spree-commerce	N/A	35 min	3%
three stage (Yin et al., 2014)	KVM	1	NA	N/A	N/A	N/A

Table 3 illustrates the objective of VM migration schemes. For instance in (Deshpande et al., 2012) entire rack migration was considered while provisioning maintenance at the rack level. Similarly, in (Gerofi et al., 2011), replicating a VM across cloud servers was considered to avoid service unavailability. In (Sahni and Varma, 2012), mixed workload hosted inside VM was taken into account while migrating VMs from one DC to another.

Table 3 debates a detailed quantitative comparison of bandwidth optimization schemes. Different bandwidth optimization live VM migration schemes result in varying application downtime and total migration time based on the nature of workload hosted within the migrant VM, type of network link, number of concurrent migrant VMs, and type of hypervisor selected to manage server resources. Bandwidth optimization schemes efficiently reduce the total migration time and downtime compared to original Xen/KVM-based VM migration schemes. For instance, IR (Deshpande et al., 2012) reduces up to 26% of the total migration time compared to the original KVM-based VM migration method, through deduplication optimization. Also, IR reduces 17% of the network traffic by eliminating redundant VM memory pages while running netw-perf workload during VM migration process.

4.2. DVFS-enabled power optimization

DVFS technology (Bambagini et al., 2013; Beloglazov, 2013, Guerrero et al., 2014; Liu et al., 2013; Wang et al., 2013, Wu et al., 2013) makes use of the relation of voltage, frequency, and processor speed ($P=V, f$, where P, V, and f symbolize power, voltage, and frequency, respectively) to adjust CPU clock rate (Shuja et al., 2012). Varying voltage and frequency adjusts CPU clock-rate to enhance energy efficiency within DC. However, DVFS application is limited to the CPU (Lee et al., 2013). For reducing power consumption, software and hardware optimization approaches are proposed (Guerrero et al., 2014; Shuja et al., 2012; Lee et al., 2013) to cut the total cost of data centers.

A power capping-based VM migration scheme was discussed in Jeong et al. (2013) that prioritizes the VM migration daemon to free the source server as soon as possible for quick maintenance provisioning. Total migration time is affected by configured VM memory size, limited bandwidth available, and type of workload hosted within a VM. VM migration helps reduce power consumption budget by migrating VMs; however, power consumption within a server suddenly surges during VM migration. To overcome this, the proposed scheme has reduced processor clock rate to lemmatize power consumption within a certain limit. However, not all CPU models support CPU clock rate adjustment to minimize power usage. To overcome the limited support offered by CPU architecture for DVFS application, the proposed approach considers “VM CAP” value to reduce power consumption. The VM CAP-based method sets “0” as a default value, signifying that VM can use unlimited free resources. By varying the value between 0 and 100, VM is restricted to limited processor time. The critical aspect of this approach is that it is only applicable for processors with DVFS support.

PMapper (Verma et al., 2008) is a power-aware application placement framework that considers power usage and migration cost while deciding on application placement within a DC. The PMapper architecture is based on three modules, namely performance manager, power manager, and monitoring engine. The performance manager holds the global view of the application in terms of QoS and performance SLA. The power manager triggers a power management module to adjust the CPU clock rate at the hardware level (e.g., DVFS application) when needed. Furthermore, the monitoring engine module gathers server/VM resource usage and power state statistics before forwarding them to the power and performance managers to adjust VM size to meet SLA. For optimal VM placement while considering power efficiency and application SLA, PMapper uses bin packing heuristics to map VMs on a suitable server. Moreover, during VM migration, the power manager adaptively applies DVFS to balance power efficiency and SLA guarantee.

Lago allocator (a scheduling algorithm) was proposed (Von Laszewski et al., 2009) to utilize DVFS and fan-control methods to limit the power consumption budget within a DC. The proposed scheduler dynamically checks application processing demands and optimizes energy consumption on-the-fly using DVFS. The Lago allocator accesses system resource usage statistics, total resource allocation, and power consumption level prior to placing the workload on a particular server. Furthermore, it sorts the servers (based on resource usage and power consumption) to choose the most suitable server (based on resource availability and power consumption estimates) to host the workload. It allocates workload based on minimizing energy consumption policy. It also identifies underutilized servers according to resource usage statistics and migrates the load to other servers to shut down servers for power efficiency. However, this approach does not address how the threshold for deciding a host's under-loading condition is calculated.

Wang et al. proposed a feed-back (Wang et al., 2008) based loop controller to manage the power consumption of a group of servers using a unified control architecture. This architecture is based on adaptive DVFS-enabled power efficiency controller, hierarchical controller for power capping, and an architecture to integrate power efficiency with power capping. The proposed scheme assigns a power budget to the workload based on the previous interval's power consumption budget. The control system architecture design consists of an efficiency controller, server capper, and group capper. The efficiency controller is responsible for tracking the demands of individual servers, whereas the server capper throttles power consumption according to feedback. Alternatively, the group capper throttles power consumption at the server group level (using DVFS). In addition to power distribution unfairness, the proposed scheme assumes the server group configuration and power supply structure are flat; however, they are actually hierarchical.

Table 4 presents a comparison of DVFS-enabled power optimization schemes to highlight commonalities and variances in existing schemes based on description, evaluation, architecture, type, and chosen workload parameters. All schemes discussed above reduce power consumption during VM migration to (a) prioritize VM migration, (b) optimally schedule the VMs, and

Table 4
DVFS enabled power optimization schemes comparison.

Scheme (Ref.)	Description	Evaluation	Architecture	Type	Workload
Lago controller (Von Laszewski et al., 2009)	Scheduling VM in a compute cluster in order to reduce power consumption	Simulation	Multi-core	DVFS	Scientific
PMapper (Verma et al., 2008)	Power aware application placement while considering application performance	Simulation	Single-core	DVFS	Multi-core
CAP (Jeong et al., 2013)	Reducing power consumption during VM migration (live mode)	Emulation	Multicore	DVFS, CAP	SPEC CPU2006 integer
Architecture (Wang et al., 2008)	Feed-back controller to manage the power budget within a DC	Simulation	Multicore	Power capping, DVFS	E-commerce, Enterprise

(c) to consume power as per the assigned power budget. The proposed schemes were experimentally carried out either by simulating the traces (Verma et al., 2008; Von Laszewski et al., 2009, Wang et al., 2008) or by emulating (Jeong et al., 2013) DVFS-enabled migration schemes using a prototype while considering a single-core (Verma et al., 2008; Von Laszewski et al., 2009) or multicore (Von Laszewski et al., 2009; Wang et al., 2008) processors. The proposed optimization schemes exploit DVFS (Verma et al., 2008) or power capping (cap value) to reduce power consumption while hosting different workloads such as scientific (Von Laszewski et al., 2009), e-commerce (Wang et al., 2008), enterprise, or integer operation-based applications (Jeong et al., 2013).

4.3. Storage optimization

In a DC, servers access the shared storage (NAS) through high-speed LAN links. However, storage sharing between sender and target servers at distant locations over the Internet is inappropriate because of higher latencies offered by WAN links (Ramakrishnan et al., 2007). Therefore, migrating a VM beyond LAN boundaries require storage migration in addition to memory and network redirection to meet several objectives such as high availability, security, and performance enhancement (Hirofuchi et al., 2009a). In this section, state-of-the-art storage migration (across WAN) schemes and architectures are discussed.

Prototype implementation of I/O blocked live storage migration (Hirofuchi et al., 2009b) rapidly relocates disk blocks within WAN links with minimum impact on I/O performance. The proposed model consists of two components, target server and proxy server connected to source and destination servers through a network block device connection (NBD) (Morrison and Schmittlein, 1988). Disk blocks are relocated transparently using “on demand” and “background copy” (transfer frequents accessed blocks) based block fetching techniques, by prioritizing the former over the latter. During the background copy phase, blocks are chosen and migrated based on the spatial locality reference method. The on-demand method fetches memory blocks from the source when they are not available at the destination server. Whenever the destination storage is completely synchronized with the source, the connection is demolished to release source server resources. However, the proposed approach does not write back data at the sender host for the sake of availability.

In case of connection failure during storage migration, the hosted application's performance significantly degrades and the system may crash. A detailed performance evaluation of Hirofuchi et al. (2009b) is presented in Hirofuchi et al. (2009a) to highlight its significance for I/O performance using several workloads. The limited WAN bandwidth degrades the live storage migration process. Therefore, to efficiently utilize bandwidth capacity, the background copy method is improved with compression using the LZO algorithm to reduce total transferred data for storage synchronization and migration time. LZO is a lightweight protocol used to reduce input data content. Introducing compression enhances network performance in terms of bandwidth utilization. The experiments revealed that I/O performance improved significantly compared to conventional remote storage migration methods in terms of total migration time and cache hit ratio.

Bitmap based storage migration scheme (Moharana, 2011) has employed simple hash algorithm such as SHA-1 to create and transfer a list of storage blocks (called sent bitmap) to the destination server. At the destination server, a sys-bitmap of already available storage blocks is generated by comparing destination storage digests with sent-bitmap. In response, source hypervisor only transfers non-available storage blocks based on sys-bitmap in parallel to dirty log generation (sys-bitmap updating). During halt-copy phase, sys-bitmap is transferred along dirty blocks and CPU state. After resuming VM at receiver, dirty blocks are migrated on demand in parallel to background copy. This approach has significantly reduced amount of blocks to be transferred to improve network performance. However, the performance of designed approach ruined if receiver server running applications update the storage blocks frequently.

A similar but enhanced approach of Moharana (2011) is discussed in Luo et al. (2008), which deploys two algorithms (TPM and IM) for complete system migration, including VM memory, storage, and CPU state between two servers. Three-phase migration (TPM) iteratively migrates storage blocks from source to destination in parallel to bitmap logging. During the stop-and-copy (second) phase, only block-bitmaps (bit “1” represents dirty page and “0” represents clean) along dirty pages and device state are transferred. In the third phase, based on block-bitmap, the storage at sender and receiver is synchronized using the push and pull method. However, in order to migrate back VMs after server maintenance, an intelligent incremental migration (IM) approach is proposed that only transfers blocks that are updated after migration from the source to reduce migration time and total migration data.

For server replication, storage is migrated either in synchronous mode (Haselhorst et al., 2011) or asynchronous mode (Medina and García, 2014). Synchronous replication is costly as it affects running applications, network, and system resources. In Ramakrishnan et al. (2007), a cooperative, context-aware migration approach was proposed, which enables the migration management system to orchestrate DC migration across server platforms, WAN, and disk storage system. To handle planned outages in this scheme, the bulk of data is migrated in asynchronous mode prior to synchronous mode replication of storage migration. In addition, for unplanned migration, check-pointing based asynchronous migration is proposed prior to stop and copy. The proposed scheme efficiency depends on the frequency of snapshots periodically captured and transferred between primary and secondary servers.

Table 5
Storage migration schemes comparison.

Scheme (Ref.)	Objectives	Storage model	Hypervisor tool	Optimization	Reliability	Evaluation
NBD (Hirofuchi et al., 2009b)	Minimizing impact of disk migration on I/O performance	Pre-copy	Xen, KVM	Spatial locality	NO	Emulation
LZO (Hirofuchi et al., 2009a)	Transparently relocating I/O operation during storage migration	Pre-copy	Xen	Data compression (LZO)	NO	Emulation
Bitmap (Moharana, 2011) TPM	improving service performance & migration time	Post-copy	N/A	Block bitmaps	YES	Simulation (CloudSim)
(Luo et al., 2008)	Optimizing downtime during whole system migration	Hybrid	Xen	Block bitmaps	YES	Emulation
Context-aware (Ramakrishnan et al., 2007)	Cooperative, context aware storage migration	N/A	N/A	Check pointing	YES	N/A
workload-aware (Zheng et al., 2011)	Optimizing storage I/O operations (workload aware)	Pre-copy	KVM	Spatial, temporal, & access locality	NO	Simulation

The workload-aware storage migration algorithm (Zheng et al., 2011) benefits from spatial locality, temporal locality, and access popularity heuristics to optimize data transfer and improve storage I/O performance during storage migration against a variety of workload range. Rather than copying all storage from source to target server, the proposed scheme deliberately computes a schedule for storage transfer at a particular granularity level (called chink). The application's I/O history forecasts the candidate blocks for expected future read/write operations based on predictions using heuristics. The proposed scheduling algorithm defers the transfer of frequently written chunks (multiple blocks' units to reduce I/O traffic) to relieve network resources. However, frequently read chunks are demanded on priority at the receiver side to reduce I/O traffic across the network. Chunk size significantly affects overall performance; however, maintaining workload history is costly because it requires additional memory resources.

Inter-site storage migration schemes severely suffer from limited shared bandwidth capacity. Table 5 compares existing storage migration schemes in terms of objectives, storage model, hypervisor management tool, optimization method, reliability, and evaluation-setup features. The objective parameter refers to the main focus of live storage migration optimization scheme. Storage migration schemes exploit pre-copy, post-copy, or hybrid live VM migration methods to migrate storage across servers using different management tools such as KVM, XEN (Wang et al., 2010b), etc. Storage migration schemes exploit diverse optimization techniques such as spatial locality (Hirofuchi et al., 2009b), compression (Hirofuchi et al., 2009a), and block-bitmaps (Luo et al., 2008) to transparently migrate storage blocks while reducing migration impact on service performance. The reliability feature highlights whether the proposed storage migration schemes are trustworthy or not (Luo et al., 2008; Ramakrishnan et al., 2007). Unreliable storage migration methods do not consider network/system outage during storage migration. Highly reliable (Moharana, 2011; Ramakrishnan et al., 2007) storage migration schemes migrate storage disks in parallel to VM image transfer in order to reduce the likelihood of SLA violation.

5. Discussion on research issues and trends

This section presents several research issues, challenges, and trends regarding server consolidation, VM memory, and storage migration. The major issues with server consolidation for power efficiency include aggressive consolidation decisions, un-controlled migrations, non-optimal VM placement, SLA violation, accuracy issues in workload demand prediction, and overlooking security concerns. Similarly, for VM memory and storage migration, the major issues include resource-hungry optimization designs, lack of intelligent pre-copy round termination, large VM data size, and poor network resource usage during system migration. In the following section, the aforementioned issues and VM migration trends are briefly discussed in reference to the literature.

Aggressive server consolidation frameworks (Hirofuchi et al., 2011; Wood et al., 2009) aim to pack the maximum number of VMs onto few servers for the sake of energy efficiency within Cloud Data Centers. Co-resident VMs (within a DC) share system resources, including CPU, system cache, memory, and network resources (Sindelar et al., 2011, Tziritas et al., 2013) to execute tasks. However, while co-hosting workloads, the majority of server consolidation frameworks considered only one (seldom two) system resource(s) while consolidating workloads but no other factors (e.g., CPU, memory, bandwidth, system cache, power), resulting in system instability (Kakadia et al., 2013). Non-optimal VM placement within a DC causes inefficient resource management, which results in application service degradation. In addition to achieving energy efficiency, another major concern of cloud operators is avoiding SLA violations during server consolidation to ensure customer satisfaction. The SLA between operator and customer is agreed upon based on several parameters, such as the number of resources available all the time, end-to-end delay, jitters, throughput, and the application's query drop rate. However, a limited number of server consolidation frameworks (Beloglazov and Buyya, 2010a, 2010b, 2010c, 2013; Ferreto et al., 2011, Kakadia et al., 2013) considered SLA violation during the consolidation process. Unnecessary and uncontrolled migrations (Hirofuchi et al., 2011; Mi et al., 2010, Sindelar et al., 2011; Wood et al., 2009) are the main reasons for SLA violation. Most server consolidation frameworks follow static thresholds to trigger migrations for server consolidation, which results in unnecessary migration (Beloglazov and Buyya, 2010a). One solution would be to use QoS-enabled migration triggering points, which would improve the application's QoS during server consolidation by reducing redundant migrations. For controlled migration decisions, machine learning-based adaptive threshold schemes can be designed to calculate thresholds based on multiple factors, such as power level, CPU usage, memory status, temperature, and communication patterns to decrease the likelihood of SLA violations. However, implementing a server consolidation process poses several challenges. The first challenge concerns accuracy while estimating the resource demand of cloud-hosted applications. Application profiling can help while estimating resource demands of currently hosted applications (Liu et al., 2013). The second challenge is the optimal distribution of VMs over physical machines. The optimal distribution of VMs is trivial due to the dynamic resource demand of hosted applications. The third challenge entails balancing server workload across servers at runtime (Pop et al., 2012; Vogels, 2008). Another important aspect that is ignored in server consolidation is security, which must be considered while co-hosting VMs within servers. Before VM placement, server consolidation schemes should check VM identity, so as not to ruin system performance. To handle this issue, the target server should use some authentication models to check VM identity before placing it in the server.

VM migration is a resource-intensive operation, demanding abundant system resources such as CPU cycles, bandwidth capacity, and system memory during VM memory transfer between communicating servers (Deshpande et al., 2012; Koto et al., 2012; Svärd et al., 2011) due to large VM memory size, unpredictable workload nature, and application QoS as agreed by customer and provider. VM migration rigorously affects the hosted and co-hosted applications' performance due to high resource contention among running applications and VM migration daemon (Hirofuchi et al., 2010). Optimization technologies such as deduplication (Deshpande et al., 2012), compression (Hirofuchi et al., 2010), and delta-based content suppression (Gerofi et al., 2011) work efficiently for specific workload types. These optimization approaches perform poorly when a non-ideal workload is hosted. For instance, deduplication approaches are efficient when source and destination servers host similar workloads. If the workloads (along with the host OS) are different, the running applications' performance is severely degraded. It is necessary to design an optimization approach that can proactively adopt optimization schemes based on learned workload behavior to decrease the trade-off between resource usage and optimization efficiency in case of dissimilar workload hosting. Furthermore, the optimization approaches mentioned above consume many system resources on computationally expensive operations, such as compressing memory pages, finding a similarity index among memory pages using deduplication, and generating page hashes (Deshpande et al., 2012; Gerofi et al., 2011; Hirofuchi et al., 2010). Lightweight optimization schemes should be designed to reduce the burden on available resources. Similarly, pre-copy iterative dirty-copying rounds are

costly if not properly managed, as similar memory is copied over and over. The iterative rounds are terminated once a favorable stop-and-copy point is encountered (Koto et al., 2012). The proposed schemes trigger a stop-and-copy phase when iterative rounds are no longer progressing, or when the dirty rate is constantly higher, becomes lower than in previous rounds, or it reaches a predefined threshold. All these solutions lead to unnecessary migration triggering due to instantaneous migration termination decisions (Gerofi et al., 2011). One way to optimize the termination point is to design a dynamic termination approach that considers the workload nature while deciding on iterative round termination. For example, for applications that prioritize turnaround time over downtime, it is better to terminate the pre-copy rounds after a few iterations. However, for applications whose top priority is downtime, it is better to seek suitable timing to terminate the iterations at the cost of bandwidth capacity. Complete system migration performance is affected as both storage and VM memory share the same network link during migration across WAN links. A better solution is to schedule storage and VM memory migration on heterogeneous communication links to optimize system throughput.

Nowadays, smart-phone technology is experiencing an unremitting renewal phase (Khan et al., 2013). Smart-phones offer adequate resources in terms of large numbers of CPU cores, increased physical memory, more cache, and storage to host new, emerging mobile applications (e.g., high-performance computing applications). However, the smart-phone application development rate is superior to smart-phone resource capabilities. Today's rich mobile applications are multi-tiered, context-aware, and power-efficient (Abolfazli et al., 2014). The rich mobile application features demand adequate system resources due to computationally expensive operations involved in application execution (Shiraz et al., 2013a). To overcome the issue of smartphones' limited resource capacity, emerging mobile applications exploit VM migration technology to migrate hosted VMs to the cloud for execution (Sanaei et al., 2014). Resource-intensive workloads are intelligently offloaded to benefit from cloud resources (Shiraz et al., 2013a, Shiraz and Gani, 2012). Clone-based migration (Chen and Noble, 2001) assists the migration daemon to track and identify the VM memory portion that is repeatedly updated in a specified period of time before seamless transfer to the cloud. The advantage of cloning is that cloud-cloned machines can recover VMs in case mobile VMs crash. However, the major concern with using clone-based technology is the strict synchronization required between source and target clients. Key issues in migrating VMs from smartphones to the cloud include limited bandwidth availability, energy consumption while transferring and receiving data to and from the cloud, inability of mobile phones to use compression-based approaches due to resource limitations (battery usage), and the characterization of application parts to be migrated. Overlay extraction and transferring via 3G/Wi-Fi links from smart-phones to the cloud can augment system performance (Sanaei et al., 2014). Also, rather than transferring VMs from smart-phones to the cloud, a mini-cloudlet (e.g., in a coffee shop) within closer proximity can be deployed to overcome problems with cloud-based expensive migration. A cloud-net consisting of a set of servers can help users in terms of on-demand resource assignment policies. However, the principal problems with using cloudlets are mobility and security.

Migrating a VM among DCs controlled and managed by the same owner is very common. However, an instance of different owners is challenging due to heterogeneous DC operating policies (Abolfazli et al., 2014; Nathuji et al., 2007; Shiraz et al., 2014; Travostino et al., 2010). For instance, suppose an application is hosted in a data center located in the USA and a major amount of application users are located in Malaysia. In this case, it would be beneficial to host the application in a data center near Malaysia (e.g. in Singapore) rather than USA. To consider these factors, VM migration controller can issues a request to migrate the application to a DC in Singapore which is operated and managed by a different company. However, the proposed idea raises several challenges, such as: What will be the charging policy for the VM in the new data center? How to ensure application privacy at the new location? How to ensure application content security? Moreover, how to enforce user confidentiality? In order to design a VM migration approach that considers inter-site migration among DCs with different operators, exploring the aforementioned issues is a must in order to design a highly scalable, flexible, and robust VM migration framework.

6. Conclusion and future works

In this paper, the notions of cloud computing, VM migration, storage migration, server consolidation, and dynamic voltage frequency scaling based power optimization are discussed. Current server consolidation and bandwidth optimization schemes are analyzed based on thematic taxonomies to highlight similarities and differences among existing schemes. Furthermore, issues with VM migration schemes are highlighted while challenges and trends in the VM migration domain are emphasized to open new research directions in order to design optimal VM migration models. Current server consolidation framework minimizes Data Center infrastructure using diverse modes to optimize power consumption. Several server consolidation frameworks co-locate VMs based on assorted factors, including mutual bandwidth, memory sharing capability, and system power budget to reduce DC computational cost. The server consolidation frameworks exploit centralized or decentralized architecture to dynamically consolidate servers. However, aggressive server consolidation degrades the application's performance. Likewise, DVFS-enabled power optimization schemes lessen power consumption at the cost of application performance while migrating VMs over WAN/LAN links. VM migration over WAN link is intricate as it requires storage migration in addition to memory transfer. Large-sized data severely affect application performance during the VM migration process. To supress large-sized data, bandwidth optimization methods exploit deduplication, compression, throttling, and dynamic rate limiting methods to effectively utilize network capacity. However, these methods drain current system resources, which rigorously affect co-hosted workload performance.

The large size of VM memory, unpredictable workload nature, limited bandwidth capacity, restricted resource sharing, inability to accurately predict application demands, and aggressive migration decisions, call for dynamic, lightweight, adaptive, and optimal VM migration designs in order to improve application performance. The lightweight VM migration design can reduce the overall development efforts, augments the application performance, and can speed-up the processing in CDC. Further, the incorporation of dynamic workload behavior characteristics in the distributed VM migration schemes design appears to be an optimal solution for the issues of bandwidth extensive VM migration. Furthermore, the inclusion of heterogeneous, dedicated, and fast communication links for storage and VM memory transferring can augment the application performance by reducing total migration time and application service downtime.

Acknowledgment

This work is fully sponsored by Bright Spark Unit and partially sponsored by the Malaysian Ministry of Education under the High Impact Research Grant of University Malaya UM.C/625/1/HIR/MOE/FCSIT/03.

References

Abolfazli S, Sanaei Z, Gani A, Xia F, Yang LT. Rich mobile applications: genesis, taxonomy, and open issues. J Netw Comput Appl 2014;40:345–62.

Aikema D, Mirtchovski A, Kiddle C, Simmonds R Green cloud VM migration: power use analysis. in: Proceedings of the IEEE International Green Computing Conference (IGCC), 2012. p. 1–6.

Al-Kiswany S, Subhraveti D, Sarkar P, Ripeanu M VMFlock: virtual machine co-migration for the cloud. In: Proceedings of the 20th International Symposium on High Performance Distributed Computing: ACM; 2011. p. 159–170.

Ali S, Jing S-Y, Kun S. Profit-aware DVFS enabled resource management of iaas cloud. Int J Comput Sci Issues (IJCSI) 2013:10.

Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, et al. A view of cloud computing. Commun ACM 2010;53:50–8.

Asberg M, Forsberg N, Nolte T, Kato S Towards real-time scheduling of virtual machines without kernel modifications. In: Proceedings of the 16th IEEE Conference on Emerging Technologies & Factory Automation (ETFA),2011. p. 1–4.

Bambagini M, Buttazzo G, Bertogna M Energy-aware scheduling for tasks with mixed energy requirements. In: Proceedings of the Real-Time Scheduling Open Problems Seminar (RTSOPS) 2013.

Barham P, Dragovic B, Fraser K, Hand S, Harris T, Ho A, et al. Xen and the art of virtualization. ACM SIGOPS Oper Syst Rev 2003;37:164–77.

Beloglazov A Energy-efficient management of virtual machines in data centers for cloud computing: PhD thesis; 2013.

Beloglazov A, Buyya R Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers. In: Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science: ACM; 2010a. p. 4.

Beloglazov A, Buyya R Energy efficient allocation of virtual machines in cloud data centers. In: Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), 2010b. p. 577–578.

Beloglazov A, Buyya R Energy efficient resource management in virtualized cloud data centers. In: Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing: IEEE Computer Society; 2010c. p. 826–831.

Beloglazov A, Buyya R. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Trans Parallel Distrib Syst 2013;24:1366–79.

Bugnion E, Devine S, Rosenblum M, Sugerman J, Wang EY. Bringing virtualization to the x86 architecture with the original VMware workstation. ACM Trans Comput Syst (TOCS) 2012;30:12.

Buyya R, Beloglazov A, Abawajy J Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges. arXiv preprint arXiv:10060308. 2010.

Cao Z, Dong S. An energy-aware heuristic framework for virtual machine consolidation in Cloud computing. J Supercomput 2014:1–23.

Chen PM, Noble BD When virtual is better than real [operating system relocation to virtual machines]. In: Proceedings of the Eighth IEEE Workshop on Hot Topics in Operating Systems,2001. p. 133–138.

Clark Christopher, Fraser Keir, Steven Hand, Jacob Gorm Hanseny,. Live Migration of Virtual Machines: In: Proceedings of the 2nd conference on Symposium on Networked Systems Design\& Implementation-Volume 2; 2005.

Deshpande U, Kulkarni U, Gopalan K Inter-rack live migration of multiple virtual machines. In: Proceedings of the 6th international workshop on Virtualization Technologies in Distributed Computing Date: ACM; 2012. p. 19–26.

Dong J, Jin X, Wang H, Li Y, Zhang P, Cheng S Energy-saving virtual machine placement in Cloud data centers. In: Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid),2013. p. 618–24.

Ferreto TC, Netto MA, Calheiros RN, De Rose CA. Server consolidation with migration control for virtualized data centers. Futur Gener Comput Syst 2011;27:1027–34.

Gerofi B, Vass Z, Ishikawa Y Utilizing memory content similarity for improving the performance of replicated virtual machines. In: Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing (UCC), 2011. p. 73–80.

Glazer DW, Tropper C. On process migration and load balancing in time warp. Parallel Distrib Syst, IEEE Trans 1993;4:318–27.

Guerrero GD, Cebrián JM, Pérez-Sánchez H, García JM, Ujaldón M, Cecilia JM. Toward energy efficiency in heterogeneous processors: findings on virtual screening methods. Concurr Comput: Pract Exp 2014;26(10):1832–46.

Habib I. Virtualization with kvm. Linux J 2008;2008:8.

Haselhorst K, Schmidt M, Schwarzkopf R, Fallenbeck N, Freisleben B Efficient storage synchronization for live migration in cloud infrastructures. In: Proceedings of the 19th IEEE Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2011 ; 2011. p. 511–18.

Hemalatha M. Cluster based bee algorithm for virtual machine placement in Cloud data centre. J Theor Appl Inform Technol 2013:57.

Hines MR, Deshpande U, Gopalan K. Post-copy live migration of virtual machines. ACM SIGOPS Operat Syst Rev 2009;43:14–26.

Hirofuchi T, Nakada H, Itoh S, Sekiguchi S Enabling instantaneous relocation of virtual machines with a lightweight vmm extension. In: Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), 2010. p. 73–83.

Hirofuchi T, Nakada H, Itoh S, Sekiguchi S Reactive consolidation of virtual machines enabled by postcopy live migration. In: Proceedings of the 5th international workshop on Virtualization technologies in distributed computing: ACM; 2011. p. 11–18.

Hirofuchi T, Nakada H, Ogawa H, Itoh S, Sekiguchi S A live storage migration mechanism over wan and its performance evaluation. In: Proceedings of the 3rd International Workshop on Virtualization Technologies in Distributed Computing: ACM; 2009a. p. 67–74.

Hirofuchi T, Ogawa H, Nakada H, Itoh S, Sekiguchi S A live storage migration mechanism over wan for relocatable virtual machine services on clouds. In: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid: IEEE Computer Society; 2009b. p. 460–65.

Hu L, Zhao J, Xu G, Ding Y, Chu J. HMDC: live virtual machine migration based on hybrid memory copy and delta compression. Appl Math 2013;7:639–46.

Huang J, Du D, Duan Q, Zhang Y, Zhao Y, Luo H, et al. Modeling and analysis on congestion control for data transmission in sensor clouds. Int J Distrib Sens Netw 2014;2014 Article ID 453983, 9 pages, http://dx.doi.org/10.1155/2014/453983.

Huang Q, Su S, Xu S, Li J, Xu P, Shuang K Migration-based elastic consolidation scheduling in Cloud data center. In: Proceedings of the 33rd IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW), 2013. p. 93–97.

Intel Corporation H-PC, Toshiba Corporation Advanced Configuration and Power Interface Specification. Hewlett-Packard/Intel/Microsoft/Phoenix/Toshiba: Intel Corporation; 2005.

Jeong J, Kim S-H, Kim H, Lee J, Seo E. Analysis of virtual machine live-migration as a method for power-capping. J Supercomp 2013;66:1629–55.

Shuja Juniad, Maddani Sajjad A, Bilal Kashif, Hayat Khizar, Khan Samee U, Sarwar Shahzad. Energy-efficient data centres. Computing 2012:937–94.

Kakadia D, Kopri N, Varma V Network-aware virtual machine consolidation for large data centers. In: Proceedings of the Third International Workshop on Network-Aware Data Management: ACM; 2013. p. 6.

Kapil D, Pilli ES, Joshi RC Live virtual machine migration techniques: survey and research challenges. In: Proceedings of the 3rd IEEE InternationalAdvance Computing Conference (IACC),: IEEE; 2013. p. 963–69.

Khan A, Othman M, Madani S, Khan S. A survey of mobile cloud computing application models, 2013.

Khan AN, Kiah MM, Ali M, Madani SA, Shamshirband S. BSS: block-based sharing scheme for secure data storage services in mobile cloud environment. J Supercomput 2014:1–31.

Kim K, Lee S, Yoo H, Kim D. Agriculture sensor-cloud infrastructure and routing protocol in the physical sensor network layer. Int J Distrib Sens Netw 2014;2014.

Koomey J. Growth in data center electricity use 2005 to 2010. A report by Analytical Press, completed at the request of The New York Times. 2011.

Koomey JG. Worldwide electricity used in data centers. Environ Res Lett 2008;3:034008.

Koto A, Yamada H, Ohmura K, Kono K. Towards unobtrusive VM live migration for cloud computing platforms. In: Proceedings of the Asia-Pacific Workshop on Systems: ACM; 2012. p. 7.

Kozuch M, Satyanarayanan M. Internet suspend/resume. In: Proceedings of the Fourth Workshop on Mobile Computing Systems and Applications, IEEE; 2002. p. 40–46.

Kozuch M, Satyanarayanan M, Bressoud T, Ke Y. Efficient state transfer for Internet suspend/resume. Carnegie Mellon University, USA: Intel Research Pittsburgh; Tech Rep IRP-TR-02-03.

KREMER J. Cloud Computing and Virtualization. 1 ed. KWCS: Kremer & Ward Consulting Services. p. 34.

Kremer J. Cloud Computing Virtualization. White paper on virtualization.

Lee L-T, Liu K-Y, Huang H-Y, Tseng C-Y A. Dynamic resource management with energy saving mechanism for supporting cloud computing. Int J Grid Distrib Comput 2013:6.

Liu H, Jin H, Liao X, Yu C, Xu C-Z. Live virtual machine migration via asynchronous replication and state synchronization. IEEE Trans Parallel Distrib Syst 2011;22:1986–99.

Liu S, Ren S, Quan G, Zhao M, Ren S Profit aware load balancing for distributed Cloud data centers. In: Proceedings of the 27th International Symposium on Parallel & Distributed Processing (IPDPS),IEEE; 2013. p. 611–22.

Luo Y, Zhang B, Wang X, Wang Z, Sun Y, Chen H Live and incremental whole-system migration of virtual machines using block-bitmap. In: Proceedings of the International Conference on Cluster Computing, IEEE; 2008. p. 99–106.

Mahajan R, Singh D. Cloud computing issues. Int J Comput Technol 2013;4:626–30.

Marston S, Li Z, Bandyopadhyay S, Zhang J, Ghalsasi A. Cloud computing—The business perspective. Decis. Support Syst 2011;51:176–89.

Medina V, García JM. A survey of migration mechanisms of virtual machines. ACM Comput Surv(CSUR) 2014;46:30.

Mell P, Grance T. The NIST definition of cloud computing (draft). NIST Spec Publ 2011;800:7.

Mi H, Wang H, Yin G, Zhou Y, Shi D, Yuan L Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers. In: Proceedings of the International Conference on Services Computing (SCC),IEEE; 2010. p. 514–21.

Milojičić DS, Douglis F, Paindaveine Y, Wheeler R, Zhou S. Process migration. ACM Comput Surv (CSUR) 2000;32:241–99.

Mishra M, Das A, Kulkarni P, Sahoo A. Dynamic resource management using virtual machine migrations. Commun Mag, IEEE 2012;50:34–40.

Mishra R, Jaiswal A. Ant colony optimization: a solution of load balancing in cloud. Int J Web Semant Technol 2012:3.

Moharana SC. An efficient approach for storage migration of virtual machines using bitmap. Computer networks and intelligent computing. Berlin, Heidelberg: Springer; 438–47.

Morrison DG, Schmittlein DC. Generalizing the NBD model for customer purchases: what are the implications and is it worth the effort? J Bus Econ Statis 1988;6:145–59.
Moura Silva L, Alonso J, Silva P, Torres J, Andrzejak A Using virtualization to improve software rejuvenation. In: Proceedings of the International Symposium on Network Computing and Applications (NCA), IEEE; 2007. p. 33–44.
Nagarajan AB, Mueller F, Engelmann C, Scott SL Proactive fault tolerance for HPC with Xen virtualization. In: Proceedings of the 21st Annual International Conference on Supercomputing: ACM; 2007. p. 23–32.
Nakada H, Hirofuchi T, Ogawa H, Itoh S. Toward virtual machine packing optimization based on genetic algorithm. Distributed computing, artificial intelligence, bioinformatics soft computing, and ambient assisted living. Berlin, Heidelber: Springer; 651–4.
Nathan S, Kulkarni P, Bellur U Resource availability based performance benchmarking of virtual machine migrations. In: Proceedings of the ACM/SPEC International Conference on International Conference on Performance Engineering: ACM; 2013. p. 387–98.
Nathuji R, Isci C, Gorbatov E Exploiting platform heterogeneity for power efficient data centers. In: Proceedings of the Fourth International Conference on IEEE Autonomic Computing (ICAC), 2007. p. 5.
Nguyen TA, LeeD, ParkJS, Towards virtualization technology on satellite on-board computer system with hardware redundancy, software rejuvenation and virtual machine live migration techniques: modeling, analysis and implementation proposal, In: Proceedings of the International Conference on Space, Aeronautical and Navigational Electronics, IEICE Technical Report 2013;113:157–62.
Niranjan Mysore R, Pamboris A, Farrington N, Huang N, Miri P, Radhakrishnan S, et al. Portland: a scalable fault-tolerant layer 2 data center network fabric. SIGCOMM Comput Comm Rev: ACM 2009:39–50.
Pop CB, Anghel I, Cioara T, Salomie I, Vartic I A swarm-inspired data center consolidation methodology. In: Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics: ACM; 2012. p. 41.
Ramakrishnan K, Shenoy P, Van der Merwe J Live data center migration across WANs: a robust cooperative context aware approach. In: Proceedings of the SIGCOMM workshop on Internet network management: ACM; 2007. p. 262–67.
Sahni S, Varma V A hybrid approach to live migration of virtual machines. In: Proceedings of the International Conference on Cloud Computing in Emerging Markets (CCEM): IEEE; 2012. p. 1–5.
Sanaei Z, Abolfazli S, Gani A, Buyya R. Heterogeneity in mobile cloud computing: taxonomy and open challenges. IEEE Commun Surv Tutor 2014;16:369–92.
Server S. Storage servers, Word Press. July 2013.
Shiraz M, Abolfazli S, Sanaei Z, Gani A. A study on virtual machine deployment for application outsourcing in mobile cloud computing. J Supercomput 2013;63:946–64.
Shiraz M, Gani A. Mobile cloud computing: critical analysis of application deployment in virtual machines. Int Proc Comput Sci Inform Technol 2012:27.
Shiraz M, Gani A, Ahmad RW, SAA Shah, Karim A, Rahman ZA. A lightweight distributed framework for computational offloading in mobile cloud computing. PloS ONE 2014;9:e102270.
Shiraz M, Gani A, Khokhar RH, Buyya R. A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing. IEEE Commun Surv Tutor 2013b;15:1294–313.
Shribman A, Hudzia B Pre-Copy and post-copy VM live migration for memory intensive applications. In: Proceedings of the Euro-Par 2012: Parallel Processing Workshops: Springer; 2013. p. 539–47.
Shrivastava V, Zerfos P, Lee K-W, Jamjoom H, Liu Y-H, Banerjee S. Application-aware virtual machine migration in data centers. In: Proceedings of INFOCOM: IEEE; 2011. p. 66–70.
Shuja J, Bilal K, Madani SA, Othman M, Ranjan R, Balaji P, et al. Survey of techniques and architectures for designing energy-efficient data centers. IEEE Syst J 2014:1–13.
Sindelar M, Sitaraman RK, Shenoy P Sharing-aware algorithms for virtual machine colocation. In: Proceedings of the 23rd symposium on Parallelism in algorithms and architectures,: ACM; 2011. p. 367–78.
Svärd P, Hudzia B, Tordsson J, Elmroth E. Evaluation of delta compression techniques for efficient live migration of large virtual machines. ACM Sigplan Notices 2011;46:111–20.
Tao J, Fürlinger K, Wang L, Marten H A performance study of virtual machines on multicore architectures. In: Proceedings of the 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2012. p. 89–96.
Thein T, Park JS. Availability analysis of application servers using software rejuvenation and virtualization. J Comput Sci Technol 2009;24:339–46.
Travostino F, Wang P, Raghunath S. Seamless live migration of virtual machines across optical networks. Google Patents; 2010.
Tziritas N, Xu C-Z, Loukopoulos T, Khan SU, Yu Z Application-aware workload consolidation to minimize both energy consumption and network load in cloud environments. In: Proceedings of the 42nd International Conference on Parallel Processing (ICPP): IEEE; 2013. p. 449–57.
Uddin M, Shah A, Alsaqour R, Memon J. Measuring efficiency of tier level data centers to implement green energy efficient data centers. Middle-East J Sci Res 2013;15:200–7.
Van Bockhaven, Jan Laan Cedric. Cryptanalysis of, and practical attacks against E-Safenet encryption. Netherlands: University of Amsterdam; 2014.
Verma A, Ahuja P, Neogi A pMapper: power and migration cost aware application placement in virtualized systems. In: Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware: Springer; 2008. p. 243–64.
Vogels W. Beyond server consolidation. Queue 2008;6:20–6.
Von Laszewski G, Wang L, Younge AJ, He X Power-aware scheduling of virtual machines in dvfs-enabled clusters. In: Proceedings of the International Conference on Cluster Computing and Workshops-CLUSTER'09, IEEE; 2009. p. 1–10.
Voorsluys W, Broberg J, Venugopal S, Buyya R. Cost of virtual machine live migration in clouds: a performance evaluation. Cloud Comput. Berlin, Heidelberg: Springer; 254–65.
Wang J, Fan ZH. Family health telemonitoring system based on WSN. Adv Mater Res 2014;860:2762–5.
Wang L, Chen D, Zhao J, Tao J. Resource management of distributed virtual machines. Int J Ad Hoc Ubiquitous Comput 2012;10:96–111.
Wang L, Laszewski GV, Tao J, Kunze M. Virtual data system on distributed virtual machines in computational grids. Int J Ad Hoc Ubiquitous Comput 2010;6:194–204.
Wang L, Von Laszewski G, Chen D, Tao J, Kunze M. Provide virtual machine information for grid computing. IEEE Trans Syst, Man Cybern, Part A: Syst Hum 2010;40:1362–74.
Wang L, Von Laszewski G, Kunze M, Tao J Schedule distributed virtual machines in a service oriented environment. In: Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA): IEEE; 2010c. p. 230–36.
Wang Z, Xu X, Xiong N, Yang LT, Zhao W. Energy cost evaluation of parallel algorithms for multiprocessor systems. Cluster Comput 2013;16:77–90.
Wang Z, Zhu X, McCarthy C, Ranganathan P, Talwar V Feedback control algorithms for power management of servers. In: Proceedings of the Third International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks (FeBid), Annapolis, MD 2008.
Webb M. SMART 2020: enabling the low carbon economy in the information age. The Climate Group London. 2008;1:1.
Whaiduzzaman M, Sookhak M, Gani A, Buyya R. A survey on vehicular cloud computing. Journal of Network and Computer Applications 2013.
Wood T, Tarasuk-Levin G, Shenoy P, Desnoyers P, Cecchet E, Corner MD Memory buddies: exploiting page sharing for smart colocation in virtualized data centers. In: Proceedings of the SIGPLAN/SIGOPS International Conference on Virtual Execution Environments: ACM; 2009. p. 31–40.
Wu C-M, Chang R-S, Chan H-Y. A green energy-efficient scheduling algorithm using the DVFS technique for cloud data centers. Futur Gener Comput Syst 2013.
Xing Y, Zhan Y. Virtualization and Cloud Computing. Future wireless networks and information systems. Berlin, Heidelberg: Springer; 305–12.
Xu F, Liu F, Jin H, Vasilakos A. Managing performance overhead of virtual machines in cloud computing: a survey, state of the art, and future directions. Proc IEEE 2014;102:11–31.
Yao L, Wu G, Ren J, Zhu Y, Li Y. Guaranteeing fault-tolerant requirement load balancing scheme based on VM migration. Comput J 2014;57:225–32.
Yin F, Liu W, Song J. Live virtual machine migration with optimized three-stage memory copy. Future information technology. Berlin, Heidelberg: Springer; 69–75.
Younge AJ, Henschel R, Brown JT, von Laszewski G, Qiu J, Fox GC. Analysis of virtualization technologies for high performance computing environments. In: Proceedings of the IEEE International Conference on Cloud Computing (CLOUD), 2011. p. 9–16.
Zhang X, Huo Z, Ma J, Meng D Exploiting data deduplication to accelerate live virtual machine migration. In: Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER), 2010. p. 88–96.
Zhang Z, Xiao L, Zhu M, Ruan L. Mvmotion: a metadata based virtual machine migration in cloud. Cluster Comput 2013:1–12.
Zheng J, Ng TSE, Sripanidkulchai K. Workload-aware live storage migration for clouds. In: Proceedings of the ACM SIGPLAN Notices; 2011. p. 133–44.
Zhou R, Liu F, Li C, Li T Optimizing virtual machine live storage migration in heterogeneous storage environment. In: Proceedings of the 9th SIGPLAN/SIGOPS International Conference on Virtual Execution Environments: ACM; 2013. p. 73–84.