Mapping Virtual Machines onto Physical Machines in Cloud Computing: A Survey

ILIA PIETRI and RIZOS SAKELLARIOU, University of Manchester

Cloud computing enables users to provision resources on demand and execute applications in a way that meets their requirements by choosing virtual resources that fit their application resource needs. Then, it becomes the task of cloud resource providers to accommodate these virtual resources onto physical resources. This problem is a fundamental challenge in cloud computing as resource providers need to map virtual resources onto physical resources in a way that takes into account the providers' optimization objectives. This article surveys the relevant body of literature that deals with this mapping problem and how it can be addressed in different scenarios and through different objectives and optimization techniques. The evaluation aspects of different solutions are also considered. The article aims at both identifying and classifying research done in the area adopting a categorization that can enhance understanding of the problem.

CCS Concepts: • **Software and its engineering** → **Virtual machines**; **Cloud computing**; • **Computer systems organization** → *Cloud computing*

Additional Key Words and Phrases: VM scheduling, VM placement, VM configuration, cloud computing

ACM Reference Format:
Ilia Pietri and Rizos Sakellariou. 2016. Mapping virtual machines onto physical machines in cloud computing: A survey. ACM Comput. Surv. 49, 3, Article 49 (October 2016), 30 pages.
DOI: http://dx.doi.org/10.1145/2983575

1. INTRODUCTION

Cloud computing has been established as a paradigm that provides computing resources on a pay-per-use basis by dynamically configuring such resources to accommodate varying workload needs. This is made possible by exploiting the benefits of *virtualization*, which enables the creation of *virtual machines* (VMs) that share physical resources. A VM is an operating system or software that emulates through virtualization the behavior of a computing system with a specified set of resource characteristics, for example, CPU and memory capacity. Virtualization is a topic with a long history [Smith and Nair 2005] that has continuously evolved to provide different capabilities, such as the execution of an application onto heterogeneous systems, the execution of multiple applications in parallel, and the movement of running applications to other hosts. Overall, virtualization technologies allow the dynamic management of resources to increase their cost-efficient utilization by creating a virtual layer, which enables the multiplexing of hardware resources so applications of different users can share the resources of a *physical machine* (PM) transparently and in isolation from each other. The main goal of this layered approach, where applications are mapped

This work was partially supported by EPSRC, under Grant No. EP/I028099/1.
Authors' addresses: I. Pietri and R. Sakellariou, School of Computer Science, The University of Manchester, Manchester M13 9PL, UK; emails: iliapietri@gmail.com, rizos@manchester.ac.uk.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2016 ACM 0360-0300/2016/10-ART49 $15.00
DOI: http://dx.doi.org/10.1145/2983575

MICROSOFT CORP.
EXHIBIT 1047

Petitioner has added a romanette to page (i). Otherwise, it leaves the original page numbering.

onto a virtual layer of *virtual machines*, which are then mapped onto the *physical machines* (cloud resources), is cost-efficient resource utilization.

The existence of such a virtual layer facilitates the deployment of user requests by allocating a number of VMs to cater for user needs. The allocated VMs may belong to different classes with varying characteristics in terms of the physical resources assigned to them; these characteristics may include different CPU speeds, different memory size, different storage capacity, and many more. A specific amount of resources can be assigned to a VM when it is required or freed when it is no longer needed. At the same time, multiple VMs, hosted by different physical machines, may serve the needs of a single user application. Typically, in this paradigm, a user will have the first word in choosing suitable VMs; the provider will have to accommodate the chosen VMs onto PMs in a way that meets the provider's criteria for cost-efficient utilization of the resources, while making sure that user requirements, often captured by a *Service Level Agreement* (SLA) [Wieder et al. 2011], are met.

As consumers wish to use a provider's computing resources in a way that meets their varying (application) requirements while providers (owners) wish to optimize the use of their resources (and, consequently, their income), the ramification is that computer resources should be available *on demand*. For example, providers would not wish to keep computing resources idle; they may switch them off if not needed and switch them back on when they are required as application resource needs fluctuate. Cloud services are often abstracted using different cloud service models, such as *Infrastructure as a Service* (IaaS) or *Platform as a Service* (PaaS) models. In IaaS models, cloud providers deliver resources as VMs on demand. Users deploy their applications on VMs while maintaining the operating systems and application software. In the case of PaaS, users (application developers) can develop and run the software on the computing platform offered by the cloud providers without the need of maintaining their own hardware resources, such as the database and web servers. The provider delivers server, storage, and network resources while the user can deploy the software and configure the settings using the provider's libraries.

The ability of scaling up and down computing resources in an automatic manner has been known as *elasticity* [Herbst et al. 2013]. Although the concept of elasticity may be overloaded in reality, it is still useful to highlight the main challenge of cloud computing, which, in abstract terms, can be described as a problem of matching at each point in time current demand with available resources [Herbst et al. 2013]. This resource management problem is a matching problem that can be solved through appropriate actions that involve the intermediate layer of virtual machines. Such actions need to answer questions and solve two fundamental problems related to both the selection of the characteristics of the virtual machines and how the machines are mapped onto physical machines. Clearly, these two problems are interrelated. For example, specifying VMs with a capacity larger than the needs of an application will result in bad resource utilization regardless of how VMs are mapped onto PMs. Conversely, mapping VMs well tailored to the needs of an application on PMs of much larger capacity will also result in bad resource utilization. As a result, it has been common in the literature to treat these two problems, often termed as *configuration* (of VMs) and *placement* (of VMs to PMs), as integral aspects of the cloud computing challenge. However, as the complexity of this challenge grows, partly due to multiple optimization objectives and partly due to the trend to offer practically unlimited opportunities for VM configuration, it becomes desirable to understand and tackle the two problems separately. It is worth noting, for example, that a number of cloud providers allow fine-tuned VM configurations (e.g., ElasticHosts[1] allows users to choose from about 184 million different combinations for the size of each of CPU, memory, and storage). Still, from the

[1]https://www.elastichosts.com.

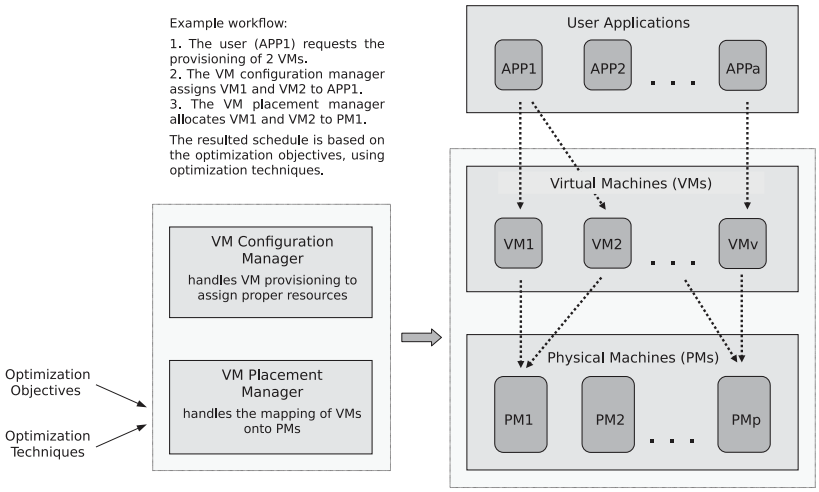

Fig. 1. Cloud computing architecture.

provider's point of view, it is important to optimize utilization so it translates into increased income while minimizing operating costs, with the highest cost often related to energy consumption [Beloglazov et al. 2012; Schulz 2009].

In view of the above, *this article surveys research on VM placement for cloud computing*, particularly focusing on how different optimization objectives can be met. In contrast to existing surveys [Galante and Bona 2012; Jennings and Stadler 2014], which typically consider the problem of managing VMs on the cloud using a holistic approach that includes both configuration and placement as defined above, we believe there is scope and it is time to produce a comprehensive survey that reviews the ever-increasing body of research by elaborating further on VM placement per se, which has become a key challenge for service providers. The idea is to enhance understanding of the VM placement problem by viewing it primarily as a scheduling and mapping problem using different objectives and not simply as a mechanism to tackle the wide problem of elasticity as has been common in previous surveys. This problem needs to be addressed either to accommodate new VMs (*initial placement*) or, dynamically, to optimize the allocation of existing VMs to PMs (*reallocation*, triggered by some observation). The underlying assumption of the survey is that this mapping problem (of VMs to PMs) is addressed by the provider following configuration of VMs through user (or other) intervention. Given some specific choices for VMs, the problem is how these VMs can be mapped onto PMs to optimize for different objectives and how one could evaluate the efficiency of different solutions in different situations.

The rest of this article is organized as follows. Section 2 presents an overview, discussing the aim of VM scheduling and mapping, the architecture, the purpose, and the context of the survey. Section 3 presents the VM placement actions. Section 4 analyzes the optimization objectives of the scheduling policies and describes common optimization techniques. Section 5 presents the application characteristics that utilize the cloud computing platform and the evaluation metrics used in the literature to assess the performance of the approaches used and categorizes the cloud computing tools. Section 6 concludes the article and summarizes challenges that have to be addressed. Finally, four tables in Sections 3–5 summarize the characteristics of representative work from the literature in relation to the focus of the relevant sections.

2. OVERVIEW

Throughout this article, we assume a pretty standard cloud computing system architecture, as diagrammatically depicted in Figure 1. Although different architecture

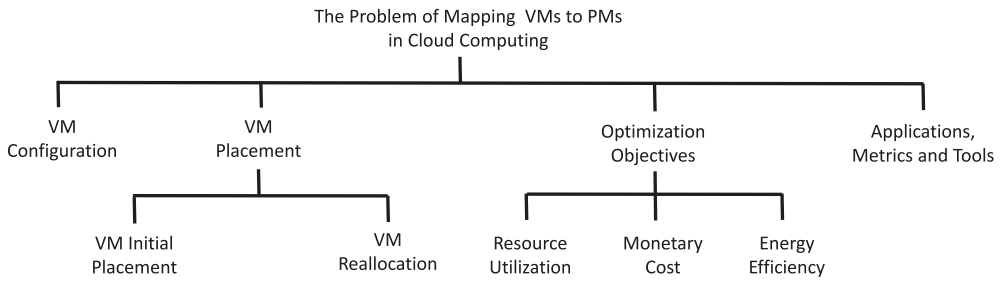

Fig. 2. Mapping VMs onto PMs in cloud computing.

diagrams can be found in the literature [Jennings and Stadler 2014; Mian et al. 2013; Van et al. 2010; Cardellini et al. 2011; Addis et al. 2010], the fundamental system characteristics from a resource management point of view do not differ. As shown on the right-hand side of the figure, there are three layers, which give rise to two fundamental mapping phases: first, user applications are mapped onto VMs; then VMs are mapped onto PMs. The two mapping phases are in principle handled by two key entities: the VM Configuration Manager and the VM Placement Manager, respectively. VM configuration will deal with issues related to the provisioning of VMs, both in terms of their number and their size (individual characteristics); user choices typically play a key role in the process. As already mentioned, VM configuration is not the focus of this article. In contrast, we assume that VM configuration has already taken place and we exclusively focus on the next phase, VM placement, which aims at optimizing the mapping of VMs onto PMs subject to different optimization objectives and applying different optimization techniques.

From a cloud provider's point of view, as the workload of the resources keeps changing, VM placement should be regarded as a continuous process. Even though an initial VM placement may take place every time before a new application's VMs start using PMs, this initial allocation may need to be reassessed as the load of physical resources changes. For example, termination of some VMs may enable the provider to switch some resources off by moving (reallocating) any VMs running on underutilized hosts. Such a reallocation may be used, for instance, to aggregate the system's load into a small number of PMs. As a result, it is useful to separate the VM placement problem to two subproblems: *initial placement* and *reallocation*, which may be invoked periodically according to certain triggering criteria. Regardless of this separation, the problem of VM placement should be viewed as a mapping problem subject to some optimization objectives. Such optimization objectives may revolve around improved resource utilization, increased income or reduced expenditure, most notably energy. Finally, different optimization techniques may target different applications and may be evaluated using different metrics and tools.

The categorization above, as diagrammatically shown in Figure 2, describes the problem of mapping VMs onto PMs in its entirety and corresponds to the structure adopted by this survey.

3. VM CONFIGURATION AND PLACEMENT

As discussed in previous sections, the process of mapping VMs onto PMs involves an initial configuration phase that determines the characteristics of VMs followed by a placement phase that attempts to place such VMs onto PMs in a cost-efficient manner according to some optimization objectives. This section focuses on the specifics of this mapping, treating it as part of a scheduling problem, which is orthogonal to particular optimization objectives.

3.1. VM Configuration

During the *VM configuration* phase, users may select the combination of resources that fit to their application needs. Providers may offer VMs of predefined VM types, with each VM type characterized by specific hardware characteristics. For example, Amazon EC2[2] offers different VM types that can cope for different user needs. However, cloud providers, such as Amazon EC2, usually offer best-effort provisioning policies that do not guarantee for performance on different workload scenarios [Naskos et al. 2015]. Also, some cloud providers may offer different resource capacity combinations with the provisioning of each selected resource capacity being charged separately; example providers are CloudSigma[3] or ElasticHosts.[1] With the different VM configurations, different tradeoffs between application performance and cost (price to pay to a provider) may arise. Cost-efficient configurations may vary depending on a provider's pricing model [Sharma et al. 2011; Zaman and Grosu 2011]. Cloud brokering mechanisms, which involve some mediation between providers and users, are also used to manage pricing heterogeneity [Anastasi et al. 2014; Tordsson et al. 2012; Lucas-Simarro et al. 2013]. Providers may also deploy combinatorial auctions where users bid for a bundle of resources of different sizes under a specific budget [Zaman and Grosu 2011]. Finally, bidding strategies and allocation mechanisms for combinatorial auctions are proposed in Zaman and Grosu [2011, 2013] and Fu et al. [2014].

In principle, VM configuration aims at assigning the proper amount of resources to avoid *under-* and *over-provisioning*. Over-provisioning of resources—provisioning more resources than required—results in resource wastage as resources remain idle. On the other hand, under-provisioning—provisioning fewer resources than required—may result in performance degradation as application requirements are not satisfied. As applications often exhibit a dynamic behaviour, their resource needs may change over time. This implies that efficient resource provisioning may require a model to estimate application resource needs over time.

For example, the tenant-based resource allocation model in Espadas et al. [2013] determines the number of VMs required to serve a certain workload so the chance of under-provisioning that may result in performance degradation is minimized, over-provisioning occurrences are kept at a minimum so resource wastage is avoided, and the policy is still cost-effective. The work in Stillwell et al. [2010] aims at maximizing application performance by optimizing the yield, a metric introduced to represent the fraction of computing capacity to be allocated to a task to the capacity consumed in the case the task would run alone on the host. The resources are fairly distributed to the requests while maintaining efficient resource utilization. In Van et al. [2010], decision making is based on utility functions that attempt to strike a balance between application performance and energy cost, differentiating the importance (priority) level of applications and using a coefficient to achieve a desirable tradeoff between these two optimization parameters. In Chang et al. [2010], a cost-aware resource provisioning algorithm that allows the reuse of resources freed by other tasks is presented. In order to determine appropriate VM configuration actions, different techniques can be used to predict the application resource needs, such as probabilistic, stochastic, and statistical; machine-learning based; or simple analytical models [Watson et al. 2010; Niehorster et al. 2011; Xu et al. 2012; Pietri et al. 2014].

Finally, the configuration of the VMs may be dynamically adjusted (VM reconfiguration or resizing) to customize the VMs when application resource requirements change, as in Dutreilh et al. [2010], Mao et al. [2010], Huu and Montagnat [2010], Calheiros

[2]http://aws.amazon.com/ec2.
[3]https://www.cloudsigma.com/.

et al. [2011], and Van et al. [2009]. Existing cloud providers, such as CloudSigma[3] and ElasticHosts,[1] allow us to dynamically adjust resource provisioning at runtime. Windows Azure[4] and Rackspace[5] also provide autoscaling to respond to unpredicted changes in demand. Extra resources can be assigned to an application in order to avoid an SLA violation. Conversely, unused resources can be freed when the SLA requirements are fulfilled to avoid over-provisioning. In that way, load balancing can be achieved while maintaining some desired level of application performance. The provisioning scheme in Liu and He [2014] addresses the problem of fair sharing among different tenants, supporting mechanisms to trade resources among tenants and adjust the resource share among VMs of the same tenant in order to exploit unused resources. The application utility function in Jung et al. [2010] shows the level at which the performance requirements are satisfied, being either a reward when response time constraints are met or a penalty in the case of a violation. In Ferretti et al. [2010], response time is monitored to adjust dynamically the system configuration and avoid SLA violations. Additional SLA requirements may also include high application throughput and availability, resource utilization, and cost minimization for the user [Frincu and Craciun 2011; Addis et al. 2010; Kokkinos et al. 2013].

After determining a specific VM configuration that meets application resource needs, the allocation of the VMs to PMs can take place in a way that meets the provider's optimization objectives without compromising the application (user) performance objectives. This allocation of VMs to PMs includes the VM initial placement actions to allocate newly arrived VMs but also VM reallocation actions to optimize the placement of existing VMs (partially or as a whole) and improve the current state of the resources.

3.2. VM Initial Placement

VM initial placement refers to the mapping of VMs to PMs (or hosts) to serve the new requests. The aim of this action is to allocate the application VMs to the PMs efficiently according to the optimization objectives. The challenge that arises in VM placement is which VMs to allocate to which PMs in order to achieve the targeted goals.

The optimization problem in VM placement can be formulated as an integer programming problem that is NP-hard. Determining whether a feasible solution exists is NP-complete. Solutions can be obtained using various algorithms. In Shi and Hong [2011], the placement problem with the goal to maximize provider's profit subject to SLA and power budget constraints is formulated, and the First Fit heuristic is used to solve it. A similar approach is used in Zhang and Ardagna [2004] to solve the flow resource allocation problem. The problem is reduced to a multi-choice binary knapsack problem, and a tabu-search algorithm is applied in order to maximize provider's profit but also meet utility-based multi-class SLAs in terms of response time. The scheme is applied for independent tiers, although it can be extended for multi-tier applications. An integer linear problem is stated in Sharma et al. [2011] to determine the transitions needed to minimize infrastructure cost; a greedy heuristic is used to solve it.

Various packing algorithms, which provide good results and require low execution time, may also be used. For example, a multi-capacity bin packing algorithm is evaluated in Stillwell et al. [2009] to solve the resource allocation problem. The algorithm runs quickly and provides better or equal results with the other examined algorithms, by optimizing the yield and the failure rate. Also, an ordered Best Fit algorithm was evaluated in Berral et al. [2011] exhibiting low execution time and close-to-optimal solutions. A First Fit heuristic is used in Shi and Hong [2011] to efficiently solve the multi-level generalized assignment problem developed; this places an unallocated VM

[4]https://www.windowsazure.com.
[5]https://www.rackspace.com/.

to the first PM in the list of available machines that satisfy the SLA and power constraints. First Fit policies are also included in the Eucalyptus[6] software framework. A modified Best Fit Decreasing heuristic is used in Beloglazov and Buyya [2010b] to sort the available hosts by decreasing resource utilization and place each VM to the PM where the allocation results in the smallest increase in power consumption.

Various application and system constraints may also be considered when making placement decisions. For example, in Ardagna et al. [2007], the SLA revenues obtained by the minimization of response time and the cost incurred by the use of resources are incorporated into an optimization problem to find a tradeoff between these two, apparently contradicting, parameters. The work in Netto and Buyya [2009] focuses on the scheduling of bag-of-tasks applications. Applications with short deadlines are given a higher priority when execution can finish in time. Jobs are scheduled according to the specified deadlines to avoid SLA violations or minimize the delay if the deadline cannot be met, distributing the requests among the various providers. The consolidation approach used in Borgetto et al. [2009] aims at minimizing the number of hosts while guaranteeing Quality of Service (QoS) to the users in order to increase energy efficiency. Power-aware placement of HPC applications is the focus in Verma et al. [2008] with power consumption modelling for benchmark applications and the impact of virtualization on the placement of applications being examined.

3.3. VM Reallocation

VM reallocation involves the rescheduling or reshuffling of VMs to the PMs to adjust the mapping to the changes in resource needs and provide scalability and reliability. Such changes may be due to applications terminating, a surge in user requests, users suspending the execution of their applications at runtime [Lee et al. 2010], or varying application resource needs over time [Stillwell et al. 2010]. In such cases, reallocation actions are required to better utilize the resources. During such a reallocation, decisions can be made proactively or reactively to improve the mapping of the VMs to PMs [Beloglazov et al. 2012]. The question that arises in VM reallocation is which VMs to select and to which PMs to allocate them, an action also known as *VM migration*. Decisions can be based on different optimization goals.

Although reallocation actions may improve the state of the system, VM migrations may lead to a system overhead and performance degradation of the migrated and collocated VMs. As the migration overhead may not be negligible, reallocation actions can be selected so multiple or frequent migrations are avoided. In reality, the migration overhead needs to be taken into account in the decision making; to do this, the time taken for migration can be modelled as a migration cost [Liu et al. 2013]. However, as modelling individual migration costs may be difficult, the total number of migrations may also be considered [Breitgand and Epstein 2011]. Reallocation decisions are performed in Petrucci et al. [2010] in order to provide power and performance optimization, modelling the transition costs of migrations and switching operations. In Ferreto et al. [2011], the number of migrations used to improve host consolidation is minimized by avoiding reallocating VMs with steady resource needs, without impacting significantly the number of required hosts. The algorithm in Ghribi et al. [2013] controls the number of migrations required while minimizing energy consumption via consolidation. In Sharma et al. [2011], migration costs are incorporated in a framework aiming at minimizing total costs. In order to manage VM migration efficiently, the work in Lu et al. [2013] introduces a multi-domain pricing and penalty model that is used from the allocation policy to select VMs for migration and reallocate them to hosts. Finally, the VMware[7]

[6]http://www.eucalyptus.com.
[7]https://www.vmware.com.

scheduler makes migration decisions according to the goodness of the processor CPUs, a metric computed based on various criteria, such as the CPU utilization and Load-Line Calibration (LLC)-level CPU load of the CPUs as well as the topological distance and communication among the CPUs. As exhaustive search of the solution space may be computationally expensive, the search space is heuristically adjusted in order to determine good candidates for migration at acceptable cost and avoid frequent migrations.

VM reallocation actions also target at achieving load balancing among the hosts, reducing power consumption, or improving application performance. In Maguluri et al. [2012], a stochastic model that focuses on distributing the load among the PMs to serve the newly allocated jobs at each timeslot is used. VM reallocation actions are invoked periodically in order to maximize system throughput, without impacting system performance with delays. In the Sandpiper framework [Wood et al. 2009], a greedy algorithm determines the VMs that have to be reallocated to migrate increased load from busy hosts to the least busy ones, while reducing migration overhead. Dynamic cost-based greedy heuristics are the focus in Le and Bianchini [2011] to allocate incoming requests and distribute the load so the placement to the PMs minimizes the additional cost of power consumption incurred. In Feller et al. [2011], an ant colony-based algorithm for dynamic VM mapping is proposed. The host consolidation approach achieves energy savings by increasing utilization and lowering the number of machines used. The scheduler proposed in Bobroff et al. [2007] reallocates VMs by using an autoregressive model to predict resource demand. In Petrucci et al. [2011], response time is the performance parameter to be controlled while minimizing the energy consumed by the system. Finally, an adaptive scheduling algorithm for workflows is proposed in Rahman et al. [2013] to remap the workflow tasks to the resources based on the critical path of the workflow graph.

3.4. Trigger of VM Placement/Reallocation

VM reallocation actions may improve the state of the system but may also be a computationally expensive procedure. For example, the algorithms used to determine the actions to be deployed, such as bin-packing algorithms, may be costly with the execution of the algorithm taking longer than the desired time. Also, frequent migrations need to be avoided, especially when the migration cost exceeds the gain from the new placement or when the performance of other running applications is significantly affected [Chen et al. 2011]. This raises the question of when it is a good time to reassess the mapping of VMs onto PMs. If this is happening often, then the cost may outweigh the benefits. On the other hand, a delayed response may lead to the rejection of new requests during a peak period or an increase in cost from resource over-provisioning. As a result, well-thought-out triggering mechanisms are required to reassess and optimize the current state of the resources in a timely, yet cost-efficient, manner. Such triggering mechanisms, or simply triggers, can be event driven, periodic, or hybrid.

Event-Driven Invocation. Event-driven triggers may relate to the arrival of new VM requests, requests to resize existing VMs, or the termination of some VMs. When new VMs arrive, some action needs to be taken to map VMs onto PMs. This action may involve only the new VMs [Lee et al. 2010; Stillwell et al. 2010] or may also include a reassessment and possible reallocation of already running VMs [Berral et al. 2010; Yang et al. 2014]. Requests for VM resizing may invoke both reconfiguration and reallocation actions to respond to the changes in application resource needs. For example, a new VM type may be required to fit the workload (due to VM resizing). Following this, reallocation actions may be invoked to better utilize the resources [Li et al. 2009; Wu

et al. 2011]. Finally, when the execution of a workload finishes, the assigned resources are released. Then, there is scope to reallocate the remaining VMs trying to identify opportunities to switch off underloaded hosts [Li et al. 2009; Lin et al. 2011].

Periodic Invocation. Scheduling actions can also be invoked periodically using predefined schedule intervals. In that case, the controller (typically controlling a scheduler) is invoked periodically to determine the actions to be taken in each scheduling round [Ardagna et al. 2007; Bobroff et al. 2007; Petrucci et al. 2010]. In von Laszewski et al. [2009], the controller sleeps for the predefined time interval and schedules the VMs waiting in the queue in each round based on the power profile of each host. In Van den Bossche et al. [2010], a 1h step is used to trigger the cost-aware controller. Periodic control is also used in Ardagna et al. [2007] to provide load balancing and scheduling in the proposed SLA-based allocation policy. Finally, in Bobroff et al. [2007], VM reallocation actions are periodically triggered to minimize the number of the hosts required to run the workload without exceeding a specified rate of SLA violations.

Hybrid Invocation. A combination of event-driven and/or periodic invocation of scheduling actions is also used in many studies, like in Ferretti et al. [2010], Goiri et al. [2010b], and Van et al. [2010]. In Goiri et al. [2010b], separate events, such as a new VM arrival or VM termination, trigger scheduling actions, while SLA violations are periodically detected. The scheduling actions aim at reducing overall power consumption, while meeting the SLA requirements of the applications. In Van et al. [2010], the triggering of the control loop can be configured so the configuration and placement actions are invoked sequentially or at different time scales. In Li et al. [2009], performance metrics are monitored periodically to determine whether extra resources are required for the workload execution, while VM resizing may trigger VM migration actions. The proposed allocation scheme adjusts the over-provision ratio, the percentage of extra resources to be allocated to a workload than actually needed, to control the triggering of resizing events and as a result frequent migrations. The approach developed in Calcavecchia et al. [2012] combines both continuous deployment to allocate newly arrived VMs to the PMs and periodic reallocation to optimize the placement of the VMs using historical data of the VM resource usage to predict the demand behavior of each VM in the future. The VMware scheduler described earlier in Section 3.3 is invoked when a time quantum allocated to a virtual CPU (vCPU) expires or a vCPU changes state (e.g., idle or ready state).

Threshold-Based Triggering. In threshold-based triggering, performance metrics are monitored to trigger the reallocation of VMs when a threshold is exceeded for a particular time interval. For example, utilization metrics are periodically collected to trigger the reallocation of VMs from overloaded hosts to less loaded hosts [Choi et al. 2008]. The proposed learning model adjusts automatically the utilization thresholds in order to distribute the load and better utilize the resources, while triggering a moderate number of migrations. In Mastroianni et al. [2011], CPU utilization of each host is controlled periodically and VM reallocation actions are invoked when it is not within the specified limits. In Wood et al. [2009], a hotspot event is invoked only when the threshold is exceeded for a number of observations within a time interval to control the number of migrations and avoid frequent transitions.

Table I summarizes the work in relation to this section with some representative examples.

4. OPTIMIZATION OBJECTIVES

This section categorizes and describes the optimization objectives of scheduling actions, which typically attempt to optimize one or more system parameters. In practice,

Table I. Representative Work on VM Configuration and Placement

Example	Description and Main Objectives	Scheduling Type Focus	Techniques Used	Evaluation
VM Configuration: FlexPRICE [Henzinger et al. 2010]	A flexible model to provide a set of options that matches execution time and price	VM Configuration at specific time intervals	Construct price curves. Various scheduling heuristics were employed	Simulation with synthetic data
VM Initial Placement: Borgetto et al. [2009]	Minimizing the number of physical machines while respecting application QoS	(Event-driven) VM Placement	A bin packing algorithm that uses the energetic yield introduced to find a placement that achieves a tradeoff between energy efficiency and QoS performance, allowing the handling of workload concentration	Simulation with synthetic data generated using normal distribution
VM Reallocation: Petrucci et al. [2010]	Dynamic management of power consumption and application performance	VM placement and reallocation with periodical control	Mixed Integer Programming model that incorporates transition costs and allows workload balancing	Simulation with web applications (World Cup Web traces)
Event-driven Triggering: Wu et al. [2011]	Algorithms that aim at minimizing infrastructure cost and SLA violations for example, response time	VM reconfiguration, placement and reallocation with event driven invocation	Minimum available space policy	Simulations on CloudSim using synthetic data
Periodic Triggering: Beloglazov et al. [2012]	Energy efficient algorithms with QoS fulfilment	VM (re)allocation periodically	Power Consumption model and priority queue	Simulations on CloudSim with synthetic data using distributions
Hybrid Control: Goiri et al. [2010b]	Energy-aware consolidation aiming at reducing the number of physical machines while respecting application QoS in terms of deadlines	VM Placement and reallocation and system reconfiguration with hybrid invocation	A greedy heuristic with cost model that incorporates SLA, VM migration and creation penalties and rewarding of loaded and reliable nodes, allowing power management with switching on/off operations	Simulations based on OMNET++ for HPC applications with hybrid resource needs using a workload from Grid5000
Windows Azure resource manager	Automatic scaling of resources	Threshold-based VM reconfiguration	Using predefined application rules	Real PaaS platform
VMWare scheduler	Scheduling to achieve load balancing and maximize resource utilization in order to optimize system throughput	VM reallocation with hybrid invocation	Use of heuristics to achieve good decision making and avoid frequent migrations	Middleware platform

system and user goals may be often conflicting; thus, considering a tradeoff between different parameters may be the goal of many scheduling actions. Three key categories to describe optimization objectives in VM scheduling become of interest:

—Resource utilization
—Monetary units
—Energy consumption

4.1. Resource Utilization

Increasing resource utilization is one of the main goals of cloud providers. Resource wastage may lead to increased energy consumption and costs due to underloaded (or underutilized) hosts but also profit loss due to the reduction of the number of applications that can be accepted. Although optimizing resource utilization is a performance-related problem, it may also be examined in relation to fairness as well. For example, in Stillwell et al. [2009], resource utilization is increased by maximizing the yield, a metric defined as the fraction of computing capacity to be allocated to a task to the capacity consuming when running the task on the host alone, so performance and fairness (defined in relation to stretch or slowdown [Bender et al. 1998]) are taken into account in the decision making. To address resource utilization, actions may be needed both in relation to hosts and the distribution of workload onto them; these are grouped next.

Host consolidation, that is, the allocation of multiple VMs onto PMs to share the hardware resources, can be used to increase resource utilization [Meng et al. 2010; Lee and Zomaya 2012; Wang et al. 2011; Farahnakian et al. 2014]. The model in Meng et al. [2010] improves host consolidation by multiplexing VMs with complementary resource need trends. The VMs are initially divided into groups according to their utilization and performance patterns and resources are allocated to each group instead of each application VM separately. The algorithm allocates the amount of the joint required resources to the group, which is the minimum required capacity to be allocated due to the complementarity of the VM resource needs. In that way, resource utilization is improved while VMs do not interfere or impact the performance of each other. The work in Eyraud-Dubois and Larchevêque [2013] focuses on the dynamic allocation of the VMs to PMs so overall resource utilization is increased while meeting the VM resource requirements over time. The proposed approach uses a bin packing algorithm that achieves efficient allocation even with unpredictable change in the CPU utilization of the VM, migrating VMs to correct and minimize SLA violations if needed. The work in Carrera et al. [2012] addresses the problem of VM placement for heterogeneous workloads with dynamic resource sharing, managing long running jobs, and transactional applications to achieve fairness among the different application targets and meet the SLA objectives. Finally, the work in Gupta et al. [2012, 2013] improves resource utilization, taking into account the cross-VM interference of the applications in the proposed bin packing heuristic to combine HPC applications and consolidate the VMs to the PMs so application performance is not compromised.

A performance model to represent different VM combinations and performance interference among VMs and evaluate system performance in a virtualized environment is developed in Kimura et al. [2014]. The problem of scheduling VMs onto shared hosts so interference among collocated VMs is reduced is also the subject in Salimi and Sharifi [2013]. A mathematical model to estimate the interference among collocated VMs is developed and used to pause and resume VMs so workload performance is improved. The consolidation approach in Wang et al. [2011] formulates the optimization problem as a stochastic bin packing problem that takes into account the network bandwidth constraints by modelling the bandwidth demands of the VMs as a probabilistic distribution. The algorithm reduces the number of required hosts. Although host consolidation

may increase resource utilization, tight consolidation of the hosts may affect the performance of the running workloads due to some resource sharing. For instance, Corradi et al. [2014] focuses on power, host, and networking resource sharing and evaluates performance degradation and consolidation constraints. Finally, in Lovász et al. [2013], a model to predict the performance degradation of a service running on a consolidated host is proposed. The model is used by two energy-aware heuristics to take into account the performance constraints of the consolidated services and approximate the formulated energy-optimal and performance-aware problem. Finally, the work in Chen et al. [2012] uses queuing theory modelling to predict application performance metrics on multi-core systems, taking into account the interference and load-dependent characteristics of the collocated VMs. The model is used to improve the consolidation of the VMs, maximizing resource utilization while meeting the application performance requirements.

Workload concentration is also used to improve resource utilization by aggregating the load to an optimal number of hosts, which is achieved by switching on and off hosts. The framework in Li et al. [2009] minimizes the number of running hosts using both static and adaptive scheduling actions in order to reduce energy consumption. Additionally, resource over-provisioning is used to avoid frequent VM resizing. Selecting the host with the least available space to map the next incoming VM waiting to be allocated is another scheduling policy used to achieve workload concentration, such as the Least Free Capacity scheme in Do and Rotter [2012]. The profit- and priority-based placement policy in Lee et al. [2010] reduces the number of instances created in order to increase the utilization of the hosts without exceeding the SLA constraints. To do so, scheduling is based on the proposed pricing model that incorporates processor-sharing and the time-varying utility function where the price charged depends on the expected response time of the service. The scheduler assigns the requests to the instances based on both the profit and response time criteria to control the incoming rate of requests of each VM instance.

An underutilized host that is not expected to serve new incoming requests can be set in a retiring state—waiting for the still running VMs to finish without accepting new incoming requests—so the host can be set to a lower-power-consuming state (e.g., to be switched off or set to idle state). When the execution of the remaining VMs is expected to continue for a long period, the remaining VMs can be migrated to other active hosts so the underutilized host can be set to a lower-power-consuming state earlier. In Lin et al. [2011], a dynamic round-robin scheme that follows these two power saving rules is proposed. The dynamic round-robin scheme combined with the First Fit heuristic further improves energy efficiency.

Workload balancing techniques are used to distribute the load among the hosts and avoid host overloading that may impact application performance, such as response time. Workload balancing policies include *round-robin* scheduling to distribute evenly the requests among the available hosts, *Join the Shortest Queue* scheduling to balance the number of waiting VMs at the queues by assigning the request to the shortest waiting queue, the *least connections* algorithm that allocates a new request to the host with the least number of active connections or requests, and the *Maximum Available Space* policy to increase the utilization of the less loaded hosts. Amazon EC2 and Rackspace[5] support different workload balancing algorithms, such as round-robin, weighted round-robin, least connections, and weighted least connections. The weighted round-robin (and least connections) policy is a modified version of the round-robin (and least connections, respectively) strategy that enables us to specify the frequency with which a request is assigned to a host. In that way, the load is distributed among heterogeneous hosts so more requests are allocated to more efficient hosts. A weighted round-robin method is also used in Petrucci et al. [2011] to balance the load among the

hosts based on the current capacity of each host. In Petrucci et al. [2010], the round-robin algorithm can dynamically respond to QoS requirements and energy issues, supporting reconfiguration, migrations, and Dynamic Voltage and Frequency Scaling (DVFS) techniques to optimize the current state of the system. An approach similar to the Join the Shortest Queue heuristic is used by the OpenNebula[8] scheduler in order to distribute the VMs to the hosts when the striping policy mode is selected. The heuristic assigns each pending VM for (re)allocation to the host with the least number of running VMs in order to maximize the resources available to the VMs. An alternative policy to the Join the Shortest Queue heuristic is applied in Maguluri et al. [2012]. In this policy, a host is randomly selected in each round and compared with the host selected in the previous round in terms of the queue length. The host with the shortest queue is then used for the placement of the VMs in the current round. In that way, throughput is increased without impacting system performance with delays. The Maximum Available Space policy is used during the migration phase in Calheiros et al. [2009] to migrate the VMs from overloaded hosts to less loaded ones. Each migrated VM is reallocated to the host with the lowest utilization (load), while respecting the bandwidth and latency constraints. Finally, an ant colony optimization approach is used in Ferdaus et al. [2014] in order to achieve workload balancing.

4.2. Monetary Units

Cost-based and utility-driven approaches express the optimization problem in some monetary units. Minimizing the operating costs or maximizing the profit and utility gain are goals from the provider's perspective, while minimization of the application cost or meeting budget constraints are goals from the user's perspective. Economic or cost functions are used to model the cost of a configuration and the total profit of the provider. The infrastructure cost, transition costs to model transition overhead, penalty costs from SLA violations, and/or the revenue from the users are some of the factors incorporated in the cost models. Revenues may also account for resource outsourcing and insourcing (renting) to other providers.

In Liu et al. [2009], the cost function comprises migration costs, energy costs, and the cost of overloaded hosts to model the potential impact of overloading on system performance. The proposed approach aims at achieving a tradeoff between energy efficiency and the provided performance. The cost model in Maurer et al. [2011] incorporates penalty costs due to SLA violations, costs due to unused resources, and the cost of actions (the percentage of the actions to be executed compared to all possible actions that could be executed) in order to optimally adjust the utilization thresholds that invoke the actions to be executed. The infrastructure cost mainly comprises the energy consumed by the used hosts and the energy spent for cooling [Le and Bianchini 2011]. The energy consumed by a host is computed taking into account the *idle power* consumed while the host is idle and the *dynamic power* required to execute the jobs, depending on the utilization level of each resource. Resource outsourcing to other providers and switching on/off operations are also incorporated in the cost model in Goiri et al. [2010a].

Operating Costs. Minimizing the operating costs has been the focus of many studies [Mian et al. 2013; Quiroz et al. 2009; Le and Bianchini 2011]. These studies may include electricity costs due to energy consumption, penalty costs due to SLA violations, and overhead costs due to transitions or migrations. Resource and penalty costs are incorporated in the model in Mian et al. [2013]. In Quiroz et al. [2009], a tradeoff between the over-provisioning cost (the additional cost from unused resources) and the

[8]http://www.opennebula.org.

wait cost that models the time between the arrival and execution of an application request (the delay of the instantiation of new VMs) is achieved. Electricity costs are taken into account in Le and Bianchini [2011] to schedule the jobs based on the energy price, cooling, and peak power demand costs. Migrations are also considered to increase cost savings.

Maximization of Provider's Profit and Utility. Increasing the gain of the provider in profit- or utility-based scheduling schemes, as in Goiri et al. [2012], Zhang and Ardagna [2004], Cardosa et al. [2009], and Goudarzi and Pedram [2011], is another goal used. In Goiri et al. [2012], the VMs are distributed to the hosts so the provider's profit is maximized taking into account energy consumption, system overheads, and penalties from SLA violations. Also, renting costs are considered to support outsourcing of VMs to other providers to further increase the profit. Dynamic programming is used in Goudarzi and Pedram [2011] to maximize total profit of the provider that is modelled taking into account the operational costs and the SLA contracts for multi-tier applications. In Cardosa et al. [2009], the utility of an application is used to prioritize more profitable applications and compute the total utility gain of a host under a specific configuration. The utility function of the application is also used to determine the resource capacity to be allocated. The optimization problem is then transformed into finding the configuration that maximizes the total utility gain of the system while also considering the power costs. However, the proposed approach does not consider migrations that could further improve the consolidation.

4.3. Energy Consumption

Minimizing energy consumption has become a main challenge in cloud computing due to the economic and environmental factors associated with increasing energy costs [Berl et al. 2009]. Resource utilization, for example, CPU, disk, storage and network, and associated equipment, such as cooling systems, are the main contributors to energy consumption and result in increased operating costs for the providers and CO_2 emissions that impact the environment.

Minimization of Active Machines. Host consolidation policies that increase resource utilization may also be used to reduce energy consumption. As the CPU comprises one of the main components that consume energy and its consumption is reasonably well understood, several studies focus on improving CPU usage and minimize the number of active machines in order to increase energy efficiency. In Kim et al. [2014], a model to estimate the energy consumed by a VM in a consolidated host is proposed. The prediction is based on performance counters, monitoring events generated by the VM. The estimation model is used by a scheduler to provide resources to a VM according to its energy budget and control its energy consumption within each time interval, suspending the execution of the VM when it consumes the energy budget it is allowed for the current interval.

Switching hosts on when resource needs increase and switching unused hosts off when resource needs decrease are among the actions that should be performed in order to improve energy efficiency. As a result, determining the optimal number of hosts required to serve the workload and provide scalability is one of the challenges that arise, which is the focus of many studies [Mastroianni et al. 2011, 2013; Zhang et al. 2014]. The work in Zhang et al. [2014] proposes a framework to determine dynamically the number of machines and adjust the provisioning of resources so a tradeoff between energy consumption and scheduling delay is achieved. The proposed framework deals with the heterogeneity of the workload to cluster tasks with similar requirements and resource needs and adjust the placement to heterogeneous physical machines taking

into account the reconfiguration cost. The policy in Dyachuk and Mazzucco [2010] dynamically adapts the number of active hosts to minimize the number of hosts required to serve the current workload while fulfilling the performance requirements of the requests when the traffic parameters are not known. In Mastroianni et al. [2011], more loaded hosts are preferred over underutilized and inactive hosts to avoid power wastage. However, VMs from overloaded hosts are migrated to avoid SLA violations, while underutilized hosts are emptied to switch them off. Thresholds are set to avoid overloading and frequent migrations. In Dong et al. [2013], the proposed algorithm combines the minimum cut algorithm to cluster VMs so network traffic is reduced and allocates the VMs to the PMs using the best-fit heuristic to increase resource utilization and improve energy efficiency by reducing both the number of active machines and network elements. Finally, Eucalyptus[6] provides scheduling with host switching on/off.

Dynamic Voltage and Frequency Scaling. Dynamic power management techniques, such as DVFS techniques, is another approach used. The power consumed by the processor is correlated with the operating frequency of the CPU. Lowering the CPU frequency may lead to power savings and potential energy savings but may impact application performance. This means that the same application may need to run longer to complete execution, thereby increasing overall energy consumption [Rauber and Rünger 2015]. In other words, in order to increase energy efficiency while meeting the SLA constraints, algorithms to determine a good frequency to use for each application are required.

In Garg et al. [2011], DVFS is deployed to determine the operating frequency of the CPU so application deadlines are met. The meta-scheduler allocates each new application to a time slot in a selected data center and determines the CPU frequency to be assigned. Scheduling takes into account the diversity of geographically distributed data centres by considering economic and environmental factors on the decision making. In Kim et al. [2011], different adaptive DVFS-based provisioning schemes to increase energy savings and profit are evaluated. The proposed algorithm selects the least expensive VM and the placement that meets the required application throughput (MIPS rate) in order to minimize the user cost. Then, DVFS is applied to increase the profit and energy efficiency. In von Laszewski et al. [2009], the PMs are configured to operate to their lowest possible voltage (frequency) in each scheduling round and the voltage of a PM is scaled up when the performance requirements of a VM to be allocated to it cannot be met. PMs with low voltages are preferred for the allocation of the VMs to avoid increasing the voltages. VMs are sorted according to the required frequency (in descending order), prioritizing VMs with higher resource needs. In Pierson and Casanova [2011], DVFS is incorporated in the placement algorithm to achieve a tradeoff between power consumption and the provided QoS level by integrating the available power states of a host in the formulated mixed integer linear problem. In Ding et al. [2015], an energy-aware algorithm that supports DVFS is proposed. The algorithm initially schedules deadline-constrained VMs to the cores of the PMs preferring PMs that can provide more computation resource within a certain power budget. To do so, the PMs are prioritized according to the performance-power ratio, a metric that is introduced to compute the ratio of the computation capacity to the peak power of each PM. The frequency to operate each core is then determined based on the total amount of resources required by the collocated VMs. Power-conserving mechanisms are also used in Quan et al. [2012], Wu et al. [2014], Young et al. [2013], and Gao et al. [2014].

Thermal Management. A significant amount of energy is consumed by cooling systems. *Thermal hotspots* may occur when load concentration is high and the energy

consumption by cooling systems increases. As a result, a challenge that arises in VM scheduling is distributing the load among the hosts so thermal hotspots are avoided. In Sandpiper [Wood et al. 2009], resource statistics data are collected to profile the resource usage of the VMs and PMs and are used to detect hotspots. Hotspots are reactively mitigated using VM resizing or migration from overloaded to less loaded hosts preferring VMs with larger volume (load) to size (memory footprint) ratio. In Chen et al. [2011], a proactive scheduling approach is used where temperature information is considered to determine the VM placement of the incoming workload to achieve thermal balance and avoid hotspots. In Xiao et al. [2013], the temperature hotspot, a metric used to measure the degree of overload of a resource, is used to mitigate hotspots. The proposed algorithm increases resource utilization of the hosts by minimizing skewness, a metric introduced to quantify uneven resource utilization, and migrates VMs based on the temperatures in order to eliminate hotspots (or at least reduce the temperature at most) when the utilization of a resource is above an allowed threshold.

Representative work in relation to different optimization objectives is presented in Table II.

4.4. Optimization Techniques

A set of common optimization techniques are typically used to achieve the optimization goals described in the previous section. These include the following:

—*Utility and Reward Functions*. Utility and reward functions model the value the execution of an application (or a configuration) has for either the user or the system (to prioritize applications accordingly). The overall gain from the transition of the system to a new configuration can also be modelled [Lee et al. 2010; Van et al. 2010; Jung et al. 2010].

—*Integer Programming*. The optimization problem can be formulated as an Integer Programming problem that describes the conflicting constraints and goals, as in Shi and Hong [2011], Zhang and Ardagna [2004], Sharma et al. [2011], and Yang et al. [2013].

—*Probabilistic, Stochastic and Statistical Models*. Probabilistic, stochastic, and statistical models are used to predict the application resource needs and configure the system depending on the estimated demand [Watson et al. 2010; Maguluri et al. 2012; Wang et al. 2011].

—*Genetic Algorithms and Artificial Intelligence Techniques*. Genetic algorithms and artificial intelligence techniques, such as swarm intelligence and neural networks, constitute another group of (often computationally expensive) methods used to predict application performance and transit the system to an optimal state [He et al. 2011; Jeyarani et al. 2012; Kousiouris et al. 2011; He et al. 2014; Farahnakian et al. 2014; Zheng et al. 2016].

—*Machine Learning Techniques*. Supervised learning algorithms, such as SVMs, and reinforcement learning techniques are examples of machine-learning techniques used to model application behaviour and plan the VM configuration actions [Niehorster et al. 2011; Xu et al. 2012].

—*Bin Packing Algorithms and Dynamic Programming*. Bin-packing algorithms and Dynamic programming techniques are mainly used for placement optimization problems [Stillwell et al. 2009; Shi and Hong 2011; Berral et al. 2011; Beloglazov and Buyya 2010b; Goudarzi and Pedram 2011].

—*Heuristics*. Heuristics, often employing simple analytical models, are used to obtain reasonably good results at a low computational time [Wood et al. 2009; Le and Bianchini 2011; Borgetto et al. 2012; Hwang and Pedram 2013].

Table III includes representative work that uses such techniques.

Table II. Representative Work on Different Optimization Objectives

Example	Description and Main Objectives	Scheduling Type Focus	Techniques Used	Evaluation
Host Consolidation: Meng et al. [2010]	A joint VM based provisioning framework to combine VMs with complementary needs aiming to increase resource utilization and minimize the number of nodes required	VM Configuration and Reconfiguration at each time frame	Statistical modelling, First Fit Decreasing algorithm for the bin packing problem	Workloads from commercial data centres
QoS Requirements: Stillwell et al. [2009]	Resource provisioning to increase resource utilization and optimize resource availability to applications, defining a metric that reflects performance and fairness	VM (re)configuration and (re)allocation	Mixed integer linear program and bin packing algorithm	Experiments using simulations with synthetically generated applications of hybrid needs
Monetary Units: Sharma et al. [2011], Kingfisher	A cost-based provisioning framework that allows minimization of infrastructure or transition cost	VM (re)configuration and reallocation at steps	Integer linear program with greedy heuristic	Experiments using the OpenNebula toolkit with realistic web applications using the TPC-W benchmark (e-commerce application)
Energy Efficiency: Wood et al. [2009], Sandpiper	A framework that provides thermal management choosing and migrating VMs according to their volume-to-size ratio, a defined metric	VM (re)allocation and reconfiguration with periodical control	Based on auto-regressive predictors and a greedy heuristic for migrations	Experiments using RuBiS for web applications
Amazon EC2 load balancer	Achieve load balancing to distribute application traffic	VM allocation	Use of (weighted) round robin techniques	Real IaaS platform
OpenNebula scheduler	Achieve different optimization goals, such as minimize the number of hosts used or distribute the load among the hosts	VM (re)allocation with event-driven triggering	Use of heuristics for workload concentration and load balancing	Middleware platform

5. APPLICATIONS, METRICS, AND PLATFORMS

In this section, the characteristics of the applications, metrics, and tools used to implement and assess the techniques of previous sections are described.

5.1. Application Characteristics

The applications can be categorized depending on their resource needs, nature, and domain.

Resource Needs. Key resource needs include CPU, memory, disk, and network capacity. Based on the resource needs, the applications can be categorized to *computationally*

Table III. Representative Work on Optimization Techniques

Example	Description and Main Objectives	Scheduling Type Focus	Techniques Used	Evaluation
Utility and Reward Functions: Jung et al. [2010], Mistral	A scheduling framework that considers power and transition costs while respecting application performance in terms of response time	VM reconfiguration and (re)allocation in stability intervals based on historical data	Use of utility function and a bin packing algorithm	RUBiS application with realistic traces for web applications from the World Cup site and a web host system (HP user's)
Stochastic Model: Bobroff et al. [2007]	host consolidation algorithm that aims at improving resource utilization while reducing SLA violations	Periodic reallocation	Using historical data for forecasting	Use of production workload traces
Integer Programming Problem: Tordsson et al. [2012]	A brokering mechanism for the placement of VMs across multiple cloud providers to incorporate application cost and QoS	VM configuration (periodically)	Integer linear program formulation and workload balancing	Realistic traces from NAS Grid Benchmarks
Genetic Algorithm: He et al. [2011]	VM consolidation and resource provisioning that aims at minimizing transition overhead	Periodical VM reconfiguration and reallocation	A genetic algorithm	Synthetic data for CPU, memory and I/O intensive applications
Bin Packing Algorithm: Beloglazov and Buyya [2010a]	Energy aware consolidation while respecting QoS focusing on finding the VMs to be migrated	VM Reallocation in each round (time frame)	A modified bin packing algorithm, implementation of DVFS and switching on/off operations	Simulations on CloudSim
Heuristics: Borgetto et al. [2012]	An energy-conscious allocation scheme for the tradeoff between power consumption and job performance	VM placement	Linear program formulation	Synthetic data with implementation in Grid5000
Machine Learning Techniques: Niehorster et al. [2011]	Development of agent-based system that learn behaviour models and provide cost estimates	Event driven VM reallocation and reconfiguration	Machine learning technique	Use of the RUBiS benchmark
ElasticHosts load balancer	Offer to users cost-efficient provisioning and good performance	VM allocation (and reconfigu-ration)	Use of round-robin policies for load balancing	Real IaaS platform
Eucalyptus scheduler	Achieve load balancing and good performance	VM allocation	Use of round robin, First Fit and host switching on/off policies	Middleware platform

intensive, consuming most of the execution time on the CPU, such as Matlab benchmarks [Kousiouris et al. 2011] and HPC applications benchmarks [Goiri et al. 2010c]; *data-intensive* applications that process large amounts of data and may require a large amount of memory and/or storage, such as the TPC-W and TPC-H benchmarks [Bu et al. 2011; Mian et al. 2013]; or *hybrid workloads* with varying resource needs, such as network and CPU together [Song et al. 2009; Viswanathan et al. 2011; Kim et al. 2009].

Evaluation Data. The evaluation data can be divided in terms of the way they were generated into synthetic data, for example, using probability distributions or emulating real data [Iqbal et al. 2010; Cardosa et al. 2009; wor 2015] or real applications [Almeida et al. 2010; Lee and Zomaya 2010; Mao et al. 2010; Huu and Montagnat 2010; Byun et al. 2011; von Laszewski et al. 2009].

Application Domain. The last categorization in relation to application characteristics is based on the target domain of the application: web-based applications, including traces from web servers [Almeida et al. 2010; Lucas-Simarro et al. 2012]; Wikipedia traces [Mazzucco et al. 2010; Mazzucco and Dumas 2011; Dyachuk and Mazzucco 2010]; World Cup traces [Petrucci et al. 2010, 2011; Mi et al. 2010; Cardellini et al. 2011]; e-commerce and transaction applications [Sharma et al. 2011; Zhang et al. 2010; Emeakaroha et al. 2011; Watson et al. 2010]; or HPC and parallel applications, including synthetic data [Stillwell et al. 2009; Lucas-Simarro et al. 2012], traces from clusters, and supercomputers or scientific workflows [Netto and Buyya 2009; Viswanathan et al. 2011; Cardosa et al. 2009].

5.2. Evaluation Metrics

This section describes the metrics used to evaluate the performance of different VM mapping approaches. The metrics are divided into four different groups of metrics: application performance metrics, host consolidation metrics, energy efficiency metrics, and monetary metrics.

Application Performance Metrics. Application performance metrics measure user satisfaction trying to capture some QoS level of performance, which may include response time or execution time [Bu et al. 2009; Lee and Zomaya 2010; Dutreilh et al. 2010], the ratio of the capacity provided to the application to the maximum capacity at optimal allocation [Espadas et al. 2013; Mastroianni et al. 2011], the number of SLA violations and delay [Lin et al. 2011; Netto and Buyya 2009; Meng et al. 2010; Wu et al. 2011], and the number of accepted or rejected requests [Tang et al. 2007; Nathani et al. 2012; Assunção et al. 2009].

Host Consolidation Metrics. To assess the effectiveness of techniques making use of host consolidation, different metrics may be typically used. Total resource utilization and system throughput [Kochut 2008; Song et al. 2009; Xu et al. 2012; Nathani et al. 2012] measure the efficiency of the system to exploit the system resources and serve as many users as possible. Algorithm execution time [Calheiros et al. 2012; Korupolu et al. 2009] is used to measure how quickly the system can respond to the changing resource needs. Finally, the number of VM migrations [Beloglazov and Buyya 2010a; Yazir et al. 2010] is also used to measure the efficiency of the algorithm as frequent transitions may impact the performance of the system.

Energy Efficiency Metrics. Although the natural way to evaluate the degree to which a proposed model achieves energy improvements is to consider overall power and energy consumption [Jeyarani et al. 2012; Berral et al. 2010; Lee and Zomaya 2012], some studies also considered the number of system resources required [Chang et al. 2010; Berral et al. 2010], these resources being either VMs or physical hosts.

Monetary Metrics. To evaluate mapping techniques from a monetary point of view, one could focus on the provider's cost (e.g., Le and Bianchini [2011] and Wu et al. [2011]), the provider's revenue [Shi and Hong 2011; Mazzucco and Dumas 2011], or user cost [Mao et al. 2010; Byun et al. 2011].

5.3. Platforms

In this section, the platforms used in the literature to deploy and assess VM mapping techniques are described. These are divided into real platforms, middleware, and simulation tools. Real platforms include some IaaS and PaaS models. Middleware and software tools include cloud computing platforms to manage data centre infrastructures and deliver software. Finally, simulation tools include simulation environments developed to test different techniques.

Real Platforms. The Amazon Elastic Compute Cloud (EC2)[2] is one of the most common real platforms used to test various mapping techniques. Following an IaaS model, it provides on-demand instances (VMs) to users to execute their applications. More recently, even more IaaS providers, such as ElasticHosts,[1] have been used. Both providers, ElasticHosts and Amazon EC2, offer round-robin load-balancing policies to distribute the requests among the hosts. Finally, Windows Azure,[4] following a PaaS model, is also commonly used. Windows Azure offers users the option to automatically scale the resources allocated to their applications by using predefined application rules. For example, users may specify thresholds in order to keep the average CPU utilization and memory usage within an acceptable range. The scheduler monitors and evaluates the selected metrics in order to determine the actions required so the specified rules are met, for example, thresholds are not exceeded in the case of a sudden workload burst.

Middleware and Software Tools. There is a variety of middleware and software tools that have been used to test different VM mapping mechanisms: OpenNebula,[8] Aneka,[9] Eucalyptus,[6] Globus Nimbus,[10] and OpenStack[11] are some of them. OpenNebula is a cloud management toolkit to build and manage IaaS clouds such as monitoring and deploying the VMs. It has been used to incorporate and test the scheduling mechanisms developed in Rodero-Merino et al. [2010], Dutreilh et al. [2010], Sharma et al. [2011], Petrucci et al. [2011], and von Laszewski et al. [2009]. Decision making is based on a rank scheduling policy that selects suitable hosts to allocate a VM according to different optimization objectives, while hosts that do not meet the resource requirements of the VM are excluded. The ranking can be adapted by selecting suitable predefined policy modes; these may include packing (workload concentration) modes to minimize the number of used hosts, load balancing and striping modes to allocate VMs to hosts with less load and distribute the VMs to the hosts, customized ranking modes to define the ranking function and sort the candidate hosts accordingly, or fixed ranking to sort the hosts according to predefined priorities. Also, OpenNebula supports VM migration, offering the option to reallocate running VMs to more suitable hosts. Reallocation actions are invoked when certain conditions are met. Aneka is another software tool for the management of resources from heterogeneous sources such as private and public clouds, used in Vecchiola et al. [2012]. Two algorithms can be deployed to provide dynamic resource provisioning; the FixedQueueProvisioningAlgorithm ensures that the size of the queue of the tasks waiting for execution does not exceed a specified threshold in order to guarantee high system throughput and good performance. The Deadline

[9]http://www.manjrasoft.com/products.html.
[10]http://www.nimbusproject.org.
[11]https://www.openstack.org/.

Priority Provisioning Algorithm is a best effort policy that adjusts the number of resources used for the execution of an application in order to meet the specified deadline. Eucalyptus [Nurmi et al. 2009] is an open-source software platform to deploy a cloud infrastructure and is commonly used to perform experiments [Espadas et al. 2013; Iqbal et al. 2010]. Eucalyptus supports three different scheduling policies; round-robin to distribute the load among the hosts, a greedy First Fit algorithm that finds the first host that satisfies the VM resource requirements, and the power save algorithm that switches idle hosts off until new VMs are allocated to it [Lin et al. 2011]. Nimbus is another open-source IaaS platform mainly aiming at the scientific community [Vázquez et al. 2011]. Nimbus allows administrators to deploy their own policies in order to automatically determine the number of resources needed for the execution of an application and manage the configuration of VMs, launching or terminating VMs accordingly. Finally, the scheduler used at OpenStack[11] filters the candidate hosts to allocate a VM by assigning weights according to selected criteria, such as the availability zone, RAM capacity, and capability of the host. The VM is assigned to the host with the minimum weight.

Simulation Tools. Experiments to test scheduling mechanisms are often performed using simulation, as simulation allows controlled experiments that help study behaviour in isolation. By far the most common simulator in use is CloudSim,[12] a cloud-computing simulation toolkit used in numerous studies [Jeyarani et al. 2012; Beloglazov and Buyya 2010b; Wu et al. 2011; Calheiros et al. 2009]. OMNET++[13], a modular framework used to build network simulators, has also been used often [Berral et al. 2010; Goiri et al. 2010a, 2010b, 2012].

Representative work in relation to applications, metrics, and platforms is summarized in Table IV.

6. DISCUSSION

Although techniques to map VMs onto PMs to provide scalability and/or elasticity are well discussed in the literature, the problem still remains an issue where further research is needed. As providers offer different pricing options and a wide range of possible VM configurations, users can choose different combinations in line with their performance and cost targets. However, with the increasing number of configurations made available, choosing appropriate hardware to serve the application VMs and map them onto PMs may risk an adverse impact on performance. For example, the performance of an application is affected depending on the CPU model and location of the used hardware when provisioning VMs of the same instance type [O'Loughlin and Gillam 2014] and the resource requirements of the collocated VMs, while the impact of the selected CPU frequency on application performance may vary for applications with different characteristics such as a workload's CPU-boundedness [Hsu and Kremer 2003]. The different combinations of application and system performance levels may reveal potential directions for future research. The challenge is to find good matches between application and system requirements so over-provisioning or under-provisioning of resources is avoided. At the moment, the trend of cloud providers to offer a large number of possible VM configurations suggests that sophisticated techniques for dynamic adjustment to actual workload requirements may have to be studied. In turn, this requires good profiling tools and prediction models to monitor and assess the impact of workload execution on heterogeneous systems.

[12]http://www.cloudbus.org/cloudsim.
[13]https://www.omnetpp.org.

Table IV. Representative Work on Applications, Metrics, and Platforms

Example	Description and Main Objectives	Scheduling Type Focus	Techniques Used	Evaluation
Application Characteristics—CPU Intensive Benchmarks: Goiri et al. [2010c]	An allocation policy to maximize provider's profit while considering energy efficiency, overheads and SLA violations and supporting resource outsourcing	Event-driven VM (re)allocation and reconfiguration	Cost-based heuristic with switching on/off operations	Use of real CPU intensive tasks from Grid5000 and implementation in OMNet++
Evaluation Data—Synthetic Traces: Almeida et al. [2010]	Scheduling policy to increase provider's profit while considering QoS requirements	VM configuration and initial placement with periodical invocation	FCFS scheduling	Use of both synthetic and realistic data, web applications
Application Domain—Web Applications: Mazzucco and Dumas [2011]	A model that aims at maximizing provider's profit	periodical VM reallocation	Use of a stochastic model	Use of Wikipedia traces for real web applications
Real Platforms: Mao et al. [2010]	A scaling framework to optimize the number of instances used allowing reduction of user cost while meeting the application deadlines	VM reconfiguration in intervals	Integer programming problem	Implementation on Windows Azure using simulations and a real scientific application, MODIS
Middleware and Software Tools: Espadas et al. [2013]	A cost-based provisioning and load balancing framework that considers multi-tenancy, finding the optimal number of VM instances required	Event-triggered VM (re)configuration and placement	Workload balancing	Implementation on Eucalyptus and simulation of a J2EE application
Simulation Tools: Calheiros et al. [2009]	Automatic virtual machines and links mapping	VM (re)allocation at each iteration	Workload balancing using available CPU utilization as the load metric	Use of synthetic data and implementation in CloudSim
OpenStack cloud controller	Achieve different optimization objectives based on selected filters	Event-driven VM allocation	Use of greedy algorithm	Middleware platform
Aneka scheduler	Dynamic resource provisioning to achieve high system throughput and meet application performance constraints	Periodic VM allocation and reconfiguration	Best effort policy	Middleware platform

In addition, traditional optimization objectives typically focus on a single goal to improve system performance. However, there are conflicting requirements, application constraints are variable, and, no matter how simply they are defined, the mapping problem of VMs onto PMs should be considered as a multi-objective optimization problem. Different solutions may consider what aspects of such an optimization problem may become more relevant than others. For example, if there is a power cap on the use of the infrastructure energy consumption may take priority when mapping VMs onto PMs. This, in turn, may imply that a different mapping mechanism is more suitable

in this scenario as opposed to the mapping mechanisms that are suitable in scenarios that are not constrained by a power cap. This raises the question of studying advanced multi-objective optimization techniques as part of the VM-to-PM mapping problem.

Finally, when and how to invoke the scheduling operations still remains an open challenge, even though, as already mentioned in Section 3.4, there are studies focusing on finding good time intervals for the scheduling decisions. Frequent controls may be costly but in-time response may be required to adjust to sudden changes and maintain a certain level of performance. To address this issue, good prediction mechanisms as well as robust mapping schemes that cope well with changes may need to be investigated.

7. CONCLUSION

A survey and categorization of techniques to map VMs onto PMs in cloud computing has been presented. The goal of this survey has been to focus on the scheduling actions and what triggers them; the optimization goals and techniques deployed; and the metrics, applications, and platforms used in evaluation of different techniques to map VMs onto PMs. Related work has been described and categorized based on this classification to enable a deep understanding of the problem with respect to its mapping properties. Finally, research directions for future work have been discussed.

REFERENCES

2015. Workflow Generator. Retrieved from https://confluence.pegasus.isi.edu/display/pegasus/Workflow Generator. (2015).

Bernardetta Addis, Danilo Ardagna, Barbara Panicucci, and Li Zhang. 2010. Autonomic management of cloud service centers with availability guarantees. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 220–227.

Jussara Almeida, Virgílio Almeida, Danilo Ardagna, Ítalo Cunha, Chiara Francalanci, and Marco Trubian. 2010. Joint admission control and resource allocation in virtualized servers. *J. Parallel Distrib. Comput.* 70, 4 (2010), 344–362.

Gaetano F. Anastasi, Emanuele Carlini, Massimo Coppola, and Patrizio Dazzi. 2014. QBROKAGE: A genetic approach for QoS cloud brokering. In *Proceedings of the 7th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 304–311.

Danilo Ardagna, Marco Trubian, and Li Zhang. 2007. SLA based resource allocation policies in autonomic environments. *J. Parallel Distrib. Comput.* 67, 3 (2007), 259–270.

Marcos Dias De Assunção, Alexandre Costanzo, and Rajkumar Buyya. 2009. Evaluating the cost-benefit of using cloud computing to extend the capacity of clusters categories and subject descriptors. In *Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing*. ACM, New York, NY, 141–150.

Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Gener. Comput. Syst.* 28, 5 (2012), 755–768.

Anton Beloglazov and Rajkumar Buyya. 2010a. 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*. IEEE, 577–578.

Anton Beloglazov and Rajkumar Buyya. 2010b. 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, 826–831.

Michael Bender, Soumen Chakrabarti, and S. Muthukrishnan. 1998. Flow and stretch metrics for scheduling continuous job streams. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, NY, 270–279.

Andreas Berl, Erol Gelenbe, Marco Di Girolamo, Giovanni Giuliani, Hermann De Meer, Minh Quan Dang, and Kostas Pentikousis. 2009. Energy-efficient cloud computing. *Comput. J.* 53, 7 (2009), 1045–1051.

Josep Ll Berral, Ricard Gavald, and Jordi Torres. 2011. Adaptive scheduling on power-aware managed data-centers using machine learning. In *Proceedings of the 12th IEEE/ACM International Conference on Grid Computing*. IEEE, 66–73.

Josep Ll Berral, ÍÑigo Goiri, Ramón Nou, Ferran Julià, Jordi Guitart, Ricard Gavaldà, and Jordi Torres. 2010. Towards energy-aware scheduling in data centers using machine learning. In *Proceedings of the 1st*

International Conference on Energy-Efficient Computing and Networking (e-Energy). ACM, New York, NY, 215–224.

Norman Bobroff, Andrzej Kochut, and Kirk Beaty. 2007. Dynamic placement of virtual machines for managing SLA violations. In *Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, 119–128.

Damien Borgetto, Henri Casanova, Georges Da Costa, and Jean-Marc Pierson. 2012. Energy-aware service allocation. *Future Gener. Comput. Syst.* 28, 5 (2012), 769–779.

Damien Borgetto, Georges Da Costa, Jean-Marc Pierson, and Amal Sayah. 2009. Energy-aware resource allocation. In *Proceedings of the 10th IEEE/ACM International Conference on Grid Computing*. IEEE, 183–188.

David Breitgand and Amir Epstein. 2011. SLA-aware placement of multi-virtual machine elastic services in compute clouds. In *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, 161–168.

Xiangping Bu, Jia Rao, and Cheng-Zhong Xu. 2009. A reinforcement learning approach to online web systems auto-configuration. In *Proceedings of the 29th IEEE International Conference on Distributed Computing Systems*. IEEE, 2–11.

Xiangping Bu, Jia Rao, and Cheng-zhong Xu. 2011. A model-free learning approach for coordinated configuration of virtual machines and appliances. In *Proceedings of the 19th Annual IEEE International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*. IEEE, 12–21.

Eun-Kyu Byun, Yang-Suk Kee, Jin-Soo Kim, and Seungryoul Maeng. 2011. Cost optimized provisioning of elastic resources for application workflows. *Future Gener. Comput. Syst.* 27, 8 (2011), 1011–1026.

Nicolò Maria Calcavecchia, Ofer Biran, Erez Hadad, and Yosef Moatti. 2012. VM placement strategies for cloud scenarios. In *Proceedings of the 5th International Conference on Cloud Computing (CLOUD)*. IEEE, 852–859.

Rodrigo N. Calheiros, Rajkumar Buyya, and César A. F. De Rose. 2009. A heuristic for mapping virtual machines and links in emulation testbeds. In *Proceedings of the International Conference on Parallel Processing*. IEEE, 518–525.

Rodrigo N. Calheiros, Rajiv Ranjan, and Rajkumar Buyya. 2011. Virtual machine provisioning based on analytical performance and QoS in cloud computing environments. In *Proceedings of the International Conference on Parallel Processing (ICPP)*. IEEE, 295–304.

Rodrigo N. Calheiros, Adel Nadjaran Toosi, Christian Vecchiola, and Rajkumar Buyya. 2012. A coordinator for scaling elastic applications across multiple clouds. *Future Gener. Comput. Syst.* 28, 8 (2012), 1350–1362.

Valeria Cardellini, Emiliano Casalicchio, Francesco Lo Presti, and Luca Silvestri. 2011. SLA-aware resource management for application service providers in the cloud. In *Proceedings of the 1st International Symposium on Network Cloud Computing and Applications*. IEEE, 20–27.

Michael Cardosa, Madhukar R. Korupolu, and Aameek Singh. 2009. Shares and utilities based power consolidation in virtualized server environments. In *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, 327–334.

David Carrera, Malgorzata Steinder, Ian Whalley, Jordi Torres, and Eduard Ayguade. 2012. Autonomic placement of mixed batch and transactional workloads. *IEEE Trans. Parallel Distrib. Syst.* 23, 2 (2012), 219–231.

Fangzhe Chang, Jennifer Ren, and Ramesh Viswanathan. 2010. Optimal resource allocation in clouds. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 418–425.

Hui Chen, Meina Song, Junde Song, Ada Gavrilovska, and Karsten Schwan. 2011. HEaRS: A hierarchical energy-aware resource scheduler for virtualized data centers. In *Proceedings of the IEEE International Conference on Cluster Computing*. IEEE, 508–512.

Lydia Y. Chen, Danilo Ansaloni, Evgenia Smirni, Akira Yokokawa, and Walter Binder. 2012. Achieving application-centric performance targets via consolidation on multicores: Myth or reality? In *Proceedings of the 21st International Symposium on High-Performance Parallel and Distributed Computing*. ACM, New York, NY, 37–48.

Hyung Won Choi, Hukeun Kwak, Andrew Sohn, and Kyusik Chung. 2008. Autonomous learning for efficient resource utilization of dynamic VM migration. In *Proceedings of the 22nd Annual International Conference on Supercomputing*. ACM, New York, NY, 185–194.

Antonio Corradi, Mario Fanelli, and Luca Foschini. 2014. VM consolidation: A real case based on OpenStack cloud. *Future Gener. Comput. Syst.* 32 (2014), 118–127.

Youwei Ding, Xiaolin Qin, Liang Liu, and Taochun Wang. 2015. Energy efficient scheduling of virtual machines in cloud with deadline constraint. *Future Gener. Comput. Syst.* 50 (2015), 62–74.

Tien Van Do and Csaba Rotter. 2012. Comparison of scheduling schemes for on-demand IaaS requests. *J. Syst. Softw.* 85, 6 (Feb. 2012), 1400–1408.

Jiankang Dong, Xing Jin, Hongbo Wang, Yangyang Li, Peng Zhang, and Shiduan Cheng. 2013. 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)*. IEEE, 618–624.

Xavier Dutreilh, Aurélien Moreau, Jacques Malenfant, Nicolas Rivierre, and Isis Truck. 2010. From data center resource allocation to control theory and back. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 410–417.

Dmytro Dyachuk and Michele Mazzucco. 2010. On allocation policies for power and performance. In *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing*. IEEE, 313–320.

Vincent C. Emeakaroha, Marco A. S. Netto, Rodrigo N. Calheiros, Ivona Brandic, Rajkumar Buyya, and César A. F. De Rose. 2011. Towards autonomic detection of SLA violations in cloud infrastructures. *Future Gener. Comput. Syst.* 28, 7 (Nov. 2011), 1017–1029.

Javier Espadas, Arturo Molina, Guillermo Jiménez, Martín Molina, Raúl Ramírez, and David Concha. 2013. A tenant-based resource allocation model for scaling software-as-a-service applications over cloud computing infrastructures. *Future Gener. Comput. Syst.* 29, 1 (2013), 273–286.

Lionel Eyraud-Dubois and Hubert Larchevêque. 2013. Optimizing resource allocation while handling SLA violations in cloud computing platforms. In *Proceedings of the 27th International Symposium on Parallel & Distributed Processing (IPDPS)*. IEEE, 79–87.

Fahimeh Farahnakian, Adnan Ashraf, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, Ivan Porres, and Hannu Tenhunen. 2014. Energy-aware dynamic VM consolidation in cloud data centers using ant colony system. In *Proceedings of the 7th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 104–111.

Eugen Feller, Louis Rilling, and Christine Morin. 2011. Energy-aware ant colony based workload placement in clouds. In *Proceedings of the 12th IEEE/ACM International Conference on Grid Computing*. IEEE, 26–33.

Md Hasanul Ferdaus, Manzur Murshed, Rodrigo N. Calheiros, and Rajkumar Buyya. 2014. Virtual machine consolidation in cloud data centers using ACO metaheuristic. In *Proceedings of Euro-Par Parallel Processing*. Springer, 306–317.

Tiago C. Ferreto, Marco A. S. Netto, Rodrigo N. Calheiros, and César A. F. De Rose. 2011. Server consolidation with migration control for virtualized data centers. *Future Gener. Comput. Syst.* 27, 8 (2011), 1027–1034.

Stefano Ferretti, Vittorio Ghini, Fabio Panzieri, Michele Pellegrini, and Elisa Turrini. 2010. QoS aware clouds. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 321–328.

Marc E. Frincu and Ciprian Craciun. 2011. Multi-objective meta-heuristics for scheduling applications with high availability requirements and cost constraints in multi-cloud environments. In *Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing*. IEEE, 267–274.

Haoming Fu, Zongpeng Li, Chuan Wu, and Xiaowen Chu. 2014. Core-selecting auctions for dynamically allocating heterogeneous VMs in cloud computing. In *Proceedings of the 7th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 152–159.

Guilherme Galante and Luis Carlos E. de Bona. 2012. A survey on cloud computing elasticity. In *Proceedings of the 5th IEEE International Conference on Utility and Cloud Computing (UCC)*. IEEE, 263–270.

Yongqiang Gao, Haibing Guan, Zhengwei Qi, Tao Song, Fei Huan, and Liang Liu. 2014. Service level agreement based energy-efficient resource management in cloud data centers. *Comput. Elec. Eng.* 40, 5 (2014), 1621–1633.

Saurabh Kumar Garg, Chee Shin Yeo, Arun Anandasivam, and Rajkumar Buyya. 2011. Environment-conscious scheduling of HPC applications on distributed cloud-oriented data centers. *J. Parallel Distrib. Comput.* 71, 6 (2011), 732–749.

Chaima Ghribi, Makhlouf Hadji, and Djamal Zeghlache. 2013. Energy efficient VM scheduling for cloud data centers: Exact allocation and migration algorithms. In *Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*. IEEE, 671–678.

Inigo Goiri, Josep Ll. Berral, J. Oriol Oriol Fitó, Ferran Julià, Ramon Nou, Jordi Guitart, Ricard Gavaldà, and Jordi Torres. 2012. Energy-efficient and multifaceted resource management for profit-driven virtualized data centers. *Future Gener. Comput. Syst.* 28, 5 (2012), 718–731.

Inigo Goiri, Jordi Guitart, and Jordi Torres. 2010a. Characterizing cloud federation for enhancing providers' profit. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 123–130.

Inigo Goiri, Ferran Julia, Ramon Nou, Josep Ll. Berral, Jordi Guitart, and Jordi Torres. 2010b. Energy-aware scheduling in virtualized datacenters. In *Proceedings of the IEEE International Conference on Cluster Computing*. IEEE, 58–67.

I. Goiri, J. Oriol Fito, F. Julia, R. Nou, J. Berral, J. Guitart, and J. Torres. 2010c. Multifaceted resource management for dealing with heterogeneous workloads in virtualized data centers. In *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing (GRID)*. IEEE, 25–32.

Hadi Goudarzi and Massoud Pedram. 2011. Multi-dimensional SLA-based resource allocation for multi-tier cloud computing systems. In *Proceedings of the 4th IEEE International Conference on Cloud Computing*. IEEE, 324–331.

Abhishek Gupta, Laxmikant V. Kalé, Dejan Milojicic, Paolo Faraboschi, and Susanne M. Balle. 2013. HPC-aware VM placement in infrastructure clouds. In *Proceedings of the IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 11–20.

Abhishek Gupta, Dejan Milojicic, and Laxmikant V. Kalé. 2012. Optimizing VM placement for HPC in the cloud. In *Proceedings of the Workshop on Cloud Services*. ACM, New York, NY, 1–6.

Ligang He, Deqing Zou, Zhang Zhang, Chao Chen, Hai Jin, and Stephen A. Jarvis. 2014. Developing resource consolidation frameworks for moldable virtual machines in clouds. *Future Gener. Comput. Syst.* 32 (2014), 69–81.

Ligang He, Deqing Zou, Zhang Zhang, Kai Yang, Hai Jin, and Stephen A. Jarvis. 2011. Optimizing resource consumptions in clouds. In *Proceedings of the 12th IEEE/ACM International Conference on Grid Computing*. IEEE, 42–49.

Thomas A. Henzinger, Anmol V. Singh, Vasu Singh, Thomas Wies, and Damien Zufferey. 2010. FlexPRICE: Flexible provisioning of resources in a cloud environment. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 83–90.

Nikolas Roman Herbst, Samuel Kounev, and Ralf Reussner. 2013. Elasticity in cloud computing: What it is, and what it is not. In *Proceedings of the 10th International Conference on Autonomic Computing (ICAC'13)*. USENIX, 23–27.

Chung-Hsing Hsu and Ulrich Kremer. 2003. The design, implementation, and evaluation of a compiler algorithm for CPU energy reduction. *ACM SIGPLAN Not.* 38, 5 (2003), 38–48.

Tram Truong Huu and Johan Montagnat. 2010. Virtual resources allocation for workflow-based applications distribution on a cloud infrastructure. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. IEEE, 612–617.

Inkwon Hwang and Massoud Pedram. 2013. Hierarchical virtual machine consolidation in a cloud computing system. In *Proceedings of the 6th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 196–203.

Waheed Iqbal, Matthew N. Dailey, and David Carrera. 2010. SLA-driven dynamic resource management for multi-tier web applications in a cloud. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. IEEE, 832–837.

Brendan Jennings and Rolf Stadler. 2014. Resource management in clouds: Survey and research challenges. *J. Netw. Syst. Manag.* 23, 3 (2014), 567–619.

Rajarathinam Jeyarani, N. Nagaveni, and Rajarathinam Vasanth Ram. 2012. Design and implementation of adaptive power-aware virtual machine provisioner (APA-VMP) using swarm intelligence. *Future Gener. Comput. Syst.* 28, 5 (May 2012), 811–821.

Gueyoung Jung, Matti A. Hiltunen, Kaustubh R. Joshi, Richard D. Schlichting, and Calton Pu. 2010. Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In *Proceedings of the 30th IEEE International Conference on Distributed Computing Systems*. IEEE, 62–73.

Hwanju Kim, Hyeontaek Lim, Jinkyu Jeong, Heeseung Jo, and Joowon Joonwon Lee. 2009. Task-aware virtual machine scheduling for I/O performance. In *Proceedings of the ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*. ACM, New York, NY, 101–110.

Kyong Hoon Kim, Anton Beloglazov, and Rajkumar Buyya. 2011. Power-aware provisioning of virtual machines for real-time cloud services. *Concurr. Comput.: Pract. Exp.* 23, 13 (2011), 1491–1505.

Nakku Kim, Jungwook Cho, and Euiseong Seo. 2014. Energy-credit scheduler: An energy-aware virtual machine scheduler for cloud systems. *Future Gener. Comput. Syst.* 32 (2014), 128–137.

Daichi Kimura, Eriko Numata, and Masato Kawatsu. 2014. Performance modeling to divide performance interference of virtualization and virtual machine combination. In *Proceedings of the 7th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 256–263.

Andrzej Kochut. 2008. On impact of dynamic virtual machine reallocation on data center efficiency. In *Proceedings of the IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems*. IEEE, 1–8.

Panagiotis Kokkinos, Theodora A. Varvarigou, Aristotelis Kretsis, Polyzois Soumplis, and Emmanouel A. Varvarigos. 2013. Cost and utilization optimization of Amazon EC2 instances. In *Proceedings of the 6th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 518–525.

Madhukar Korupolu, Aameek Singh, and Bhuvan Bamba. 2009. Coupled placement in modern data centers. In *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing*. IEEE, 1–12.

George Kousiouris, Tommaso Cucinotta, and Theodora Varvarigou. 2011. The effects of scheduling, workload type and consolidation scenarios on virtual machine performance and their prediction through optimized artificial neural networks. *J. Syst. Softw.* 84, 8 (2011), 1270–1291.

Kien Le and Ricardo Bianchini. 2011. Reducing electricity cost through virtual machine placement in high performance computing clouds. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, New York, NY, 22:1–22:12.

Young Choon Lee, Chen Wang, Albert Y. Zomaya, and Bing Bing Zhou. 2010. Profit-driven service request scheduling in clouds. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. IEEE, 15–24.

Young Choon Lee and Albert Y. Zomaya. 2010. Rescheduling for reliable job completion with the support of clouds. *Future Gener. Comput. Syst.* 26, 8 (Oct. 2010), 1192–1199.

Young Choon Lee and Albert Y. Zomaya. 2012. Energy efficient utilization of resources in cloud computing systems. *J. Supercomput.* 60, 2 (2012), 268–280.

Bo Li, Jianxin Li, Jinpeng Huai, Tianyu Wo, Qin Li, and Liang Zhong. 2009. EnaCloud: An energy-saving application live placement approach for cloud computing environments. In *Proceedings of the IEEE International Conference on Cloud Computing*. IEEE, 17–24.

Ching-Chi Lin, Pangfeng Liu, and Jan-Jan Wu. 2011. Energy-efficient virtual machine provision algorithms for cloud systems. In *Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing*. IEEE, 81–88.

Haikun Liu and Bingsheng He. 2014. Reciprocal resource fairness: Towards cooperative multiple-resource fair sharing in IaaS clouds. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC14)*. IEEE, 970–981.

Haikun Liu, Hai Jin, Cheng-Zhong Xu, and Xiaofei Liao. 2013. Performance and energy modeling for live migration of virtual machines. *Clust. Comput.* 16, 2 (2013), 249–264.

Liang Liu, Hao Wang, Xue Liu, Xing Jin, Wenbo He, Qingbo Wang, and Ying Chen. 2009. GreenCloud: A new architecture for green data center. In *Proceedings of the 6th International Conference Industry Session on Autonomic Computing and Communications Industry Session*. ACM, New York, NY, 29–38.

Gergő Lovász, Florian Niedermeier, and Hermann de Meer. 2013. Performance tradeoffs of energy-aware virtual machine consolidation. *Clust. Comput.* 16, 3 (2013), 481–496.

Kuan Lu, Ramin Yahyapour, Philipp Wieder, Constantinos Kotsokalis, Edwin Yaqub, and Ali Imran Jehangiri. 2013. QoS-aware VM placement in multi-domain service level agreements scenarios. In *Proceedings of the 6th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 661–668.

Jose Luis Lucas-Simarro, Rafael Moreno-Vozmediano, Ruben S. Montero, and Ignacio M. Llorente. 2012. Scheduling strategies for optimal service deployment across multiple clouds. *Future Gener. Comput. Syst.* (Jan. 2012), 1–11.

Jose Luis Lucas-Simarro, Rafael Moreno-Vozmediano, Ruben S. Montero, and Ignacio M. Llorente. 2013. Scheduling strategies for optimal service deployment across multiple clouds. *Future Gener. Comput. Syst.* 29, 6 (2013), 1431–1441.

Siva Theja Maguluri, R. Srikant, and Lei Ying. 2012. Stochastic models of load balancing and scheduling in cloud computing clusters. In *Proceedings of the IEEE INFOCOM*. IEEE, 702–710.

Ming Mao, Jie Li, and Marty Humphrey. 2010. Cloud auto-scaling with deadline and budget constraints. In *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing*. IEEE, 41–48.

Carlo Mastroianni, Michela Meo, and Giuseppe Papuzzo. 2011. Self-economy in cloud data centers: Statistical assignment and migration of virtual machines. In *Proceedings of Euro-Par Parallel Processing*. Springer, 407–418.

Carlo Mastroianni, Michela Meo, and Giuseppe Papuzzo. 2013. Probabilistic consolidation of virtual machines in self-organizing cloud data centers. *IEEE Trans. Cloud Comput.* 1, 2 (2013), 215–228.

Michael Maurer, Ivona Brandic, and Rizos Sakellariou. 2011. Enacting SLAs in clouds using rules. In *Proceedings of Euro-Par*. Springer, 455–466.

Michele Mazzucco and Marlon Dumas. 2011. Reserved or on-demand instances? A revenue maximization model for cloud providers. In *Proceedings of the 4th IEEE International Conference on Cloud Computing*. IEEE, 428–435.

Michele Mazzucco, Dmytro Dyachuk, and Ralph Deters. 2010. Maximizing cloud providers' revenues via energy aware allocation policies. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 131–138.

Xiaoqiao Meng, Canturk Isci, Jeffrey Kephart, Li Zhang, Eric Bouillet, and Dimitrios Pendarakis. 2010. Efficient resource provisioning in compute clouds via VM multiplexing. In *Proceedings of the 7th International Conference on Autonomic Computing*. ACM, New York, NY, 11–20.

Haibo Mi, Huaimin Wang, Gang Yin, Yangfan Zhou, Dianxi Shi, and Lin Yuan. 2010. Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers. In *Proceedings of the IEEE International Conference on Services Computing*. IEEE, 514–521.

Rizwan Mian, Patrick Martin, and Jose Luis Vazquez-Poletti. 2013. Provisioning data analytic workloads in a cloud. *Future Gener. Comput. Syst.* 29, 6 (2013), 1452–1458.

Athanasios Naskos, Emmanouela Stachtiari, Panagiotis Katsaros, and Anastasios Gounaris. 2015. Probabilistic model checking at runtime for the provisioning of cloud resources. In *Runtime Verification*. Springer, 275–280.

Amit Nathani, Sanjay Chaudhary, and Gaurav Somani. 2012. Policy based resource allocation in IaaS cloud. 28, 1 (2012), 94–103.

Marco A. S. Netto and Rajkumar Buyya. 2009. Offer-based scheduling of deadline-constrained bag-of-tasks applications for utility computing systems. In *Proceedings of the IEEE International Parallel and Distributed Processing Symposium*. IEEE, 1–11.

Oliver Niehorster, Alexander Krieger, Jens Simon, and A. Brinkmann. 2011. Autonomic resource management with support vector machines. In *Proceedings of the 12th IEEE/ACM International Conference on Grid Computing*. IEEE, 157–164.

Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov. 2009. The Eucalyptus open-source cloud-computing system. In *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*. IEEE, 124–131.

John O'Loughlin and Lee Gillam. 2014. Performance evaluation for cost-efficient public infrastructure cloud use. In *Economics of Grids, Clouds, Systems, and Services*. Springer, 133–145.

Vinicius Petrucci, Enrique V. Carrera, Orlando Loques, Julius C. B. Leite, and Daniel Mossé. 2011. Optimized management of power and performance for virtualized heterogeneous server clusters. In *Proceedings of the 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 23–32.

Vinicius Petrucci, Orlando Loques, and D. Mosse. 2010. A dynamic optimization model for power and performance management of virtualized clusters. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*. ACM, 225–233.

Jean-Marc Pierson and Henri Casanova. 2011. On the utility of DVFS for power-aware job placement in clusters. In *Proceedings of Euro-Par*. Springer, 255–266.

Ilia Pietri, Gideon Juve, Ewa Deelman, and Rizos Sakellariou. 2014. A performance model to estimate execution time of scientific workflows on the cloud. In *Proceedings of the 9th Workshop on Workflows in Support of Large-Scale Science*. IEEE, 11–19.

Dang Minh Quan, Federico Mezza, Domenico Sannenli, and Raffaele Giafreda. 2012. T-Alloc: A practical energy efficient resource allocation algorithm for traditional data centers. *Future Gener. Comput. Syst.* 28, 5 (2012), 791–800.

Andres Quiroz, Hyunjoo Kim, Manish Parashar, N. Gnanasambandam, and N Sharma. 2009. Towards autonomic workload provisioning for enterprise grids and clouds. In *Proceedings of the 10th IEEE/ACM International Conference on Grid Computing*. IEEE, 50–57.

Mustafizur Rahman, Rafiul Hassan, Rajiv Ranjan, and Rajkumar Buyya. 2013. Adaptive workflow scheduling for dynamic grid and cloud computing environment. *Concurr. Comput.: Pract. Exp.* 25, 13 (2013), 1816–1842.

Thomas Rauber and Gudula Rünger. 2015. Modeling and analyzing the energy consumption of fork-join-based task parallel programs. *Concurr. Comput.: Pract. Exp.* 27, 1 (2015), 211–236.

Luis Rodero-Merino, Luis M. Vaquero, Victor Gil, Fermín Galán, Javier Fontán, Rubén S. Montero, and Ignacio M. Llorente. 2010. From infrastructure delivery to service management in clouds. *Future Gener. Comput. Syst.* 26, 8 (2010), 1226–1240.

Hadi Salimi and Mohsen Sharifi. 2013. Batch scheduling of consolidated virtual machines based on their workload interference model. *Future Gener. Comput. Syst.* 29, 8 (2013), 2057–2066.

Greg Schulz. 2009. *The Green and Virtual Data Center*. CRC Press.

Upendra Sharma, Prashant Shenoy, Sambit Sahu, and Anees Shaikh. 2011. A cost-aware elasticity provisioning system for the cloud. In *Proceedings of the 31st International Conference on Distributed Computing Systems*. IEEE, 559–570.

Weiming Shi and Bo Hong. 2011. Towards profitable virtual machine placement in the data center. In *Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing*. IEEE, 138–145.

James E. Smith and Ravi Nair. 2005. The architecture of virtual machines. *Computer* 38, 5 (2005), 32–38.

Ying Song, Hui Wang, Yaqiong Li, Binquan Feng, and Yuzhong Sun. 2009. Multi-tiered on-demand resource scheduling for VM-based data center. In *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*. IEEE, 148–155.

Mark Stillwell, David Schanzenbach, Frédéric Vivien, and Henri Casanova. 2009. Resource allocation using virtual clusters. In *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*. IEEE, 260–267.

Mark Stillwell, David Schanzenbach, Frédéric Vivien, and Henri Casanova. 2010. Resource allocation algorithms for virtualized service hosting platforms. *J. Parallel Distrib. Comput.* 70, 9 (2010), 962–974.

Chunqiang Tang, Malgorzata Steinder, Michael Spreitzer, and Giovanni Pacifici. 2007. A scalable application placement controller for enterprise data centers. In *Proceedings of the 16th International Conference on World Wide Web*. ACM, 331–340.

Johan Tordsson, Rubén S. Montero, Rafael Moreno-Vozmediano, and Ignacio M. Llorente. 2012. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future Gener. Comput. Syst.* 28, 2 (2012), 358–367.

Hien Nguyen Van, Dang Tran, and Jean-Marc Menaud. 2009. Autonomic virtual resource management for service hosting platforms. In *Proceedings of the Workshop on Software Engineering Challenges in Cloud Computing*. IEEE, 1–8.

Hien Nguyen Van, Frédéric Dang Tran, and Jean-Marc Menaud. 2010. Performance and power management for cloud infrastructures. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 329–336.

Ruben Van den Bossche, Kurt Vanmechelen, and Jan Broeckhove. 2010. Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 228–235.

Constantino Vázquez, Eduardo Huedo, Rubén S. Montero, and Ignacio M. Llorente. 2011. On the use of clouds for grid resource provisioning. *Future Gener. Comput. Syst.* 27, 5 (2011), 600–605.

Christian Vecchiola, Rodrigo N. Calheiros, Dileban Karunamoorthy, and Rajkumar Buyya. 2012. Deadline-driven provisioning of resources for scientific applications in hybrid clouds with Aneka. *Future Gener. Comput. Syst.* 28, 1 (Jan. 2012), 58–65.

Akshat Verma, Puneet Ahuja, and Anindya Neogi. 2008. Power-aware dynamic placement of HPC applications. In *Proceedings of the 22nd Annual International Conference on Supercomputing*. ACM, New York, NY, 175–184.

Hariharasudhan Viswanathan, Eun Kyung Lee, Ivan Rodero, Dario Pompili, Manish Parashar, and Marc Gamell. 2011. Energy-aware application-centric VM allocation for HPC workloads. In *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing Workshops and PhD Forum*. IEEE, 890–897.

Gregor von Laszewski, Lizhe Wang, Andrew J. Younge, and Xi He. 2009. Power-aware scheduling of virtual machines in DVFS-enabled clusters. In *Proceedings of the IEEE International Conference on Cluster Computing and Workshops*. IEEE, 1–10.

Meng Wang, Xiaoqiao Meng, and Li Zhang. 2011. Consolidating virtual machines with dynamic bandwidth demand in data centers. In *Proceedings of the IEEE INFOCOM*. IEEE, 71–75.

Brian J. Watson, Manish Marwah, Daniel Gmach, Yuan Chen, Martin Arlitt, and Zhikui Wang. 2010. Probabilistic performance modeling of virtualized resource allocation. In *Proceedings of the 7th International Conference on Autonomic Computing*. ACM, New York, NY, 99–108.

Philipp Wieder, Joe M. Butler, Wolfgang Theilmann, and Ramin Yahyapour. 2011. *Service Level Agreements for Cloud Computing*. Springer Science & Business Media.

Timothy Wood, Prashant Shenoy, Arun Venkataramani, and Mazin Yousif. 2009. Sandpiper: Black-box and gray-box resource management for virtual machines. *Comput. Netw.* 53, 17 (2009), 2923–2938.

Chia-Ming Wu, Ruay-Shiung Chang, and Hsin-Yu Chan. 2014. A green energy-efficient scheduling algorithm using DVFS technique for cloud datacenters. *Future Gener. Comput. Syst.* 37 (2014), 141–147.

Linlin Wu, Saurabh Kumar Garg, and Rajkumar Buyya. 2011. SLA-based resource allocation for software as a service provider (SaaS) in cloud computing environments. In *Proceedings of the 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 195–204.

Zhen Xiao, Weijia Song, and Qi Chen. 2013. Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE Trans. Parallel Distrib. Syst.* 24, 6 (2013), 1107–1117.

Cheng-Zhong Xu, Jia Rao, and Xiangping Bu. 2012. URL: A unified reinforcement learning approach for autonomic cloud management. *J. Parallel Distrib. Comput.* 72, 2 (Feb. 2012), 95–105.

Jingqi Yang, Chuanchang Liu, Yanlei Shang, Zexiang Mao, and Junliang Chen. 2013. Workload predicting-based automatic scaling in service clouds. In *Proceedings of the 6th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 810–815.

Laurence Yang, Xiaomin Zhu, Huangke Chen, Ji Wang, Shu Yin, and Xiaocheng Liu. 2014. Real-time tasks oriented energy-aware scheduling in virtualized clouds. *IEEE Trans. Cloud Comput.* 2, 2 (2014), 168–180.

Yagiz Onat Yazir, Chris Matthews, Roozbeh Farahbod, Stephen Neville, Adel Guitouni, Sudhakar Ganti, and Yvonne Coady. 2010. Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 91–98.

Bobby Dalton Young, Sudeep Pasricha, Anthony A. Maciejewski, Howard Jay Siegel, and James T. Smith. 2013. Heterogeneous makespan and energy-constrained DAG scheduling. In *Proceedings of the Workshop on Energy Efficient High Performance Parallel and Distributed Computing*. ACM, New York, NY, 3–12.

Sharrukh Zaman and Daniel Grosu. 2011. Efficient bidding for virtual machine instances in clouds. In *Proceedings of the 4th IEEE International Conference on Cloud Computing*. IEEE, 41–48.

Sharrukh Zaman and Daniel Grosu. 2013. A combinatorial auction-based mechanism for dynamic VM provisioning and allocation in clouds. *IEEE Trans. Cloud Comput.* 1, 2 (2013), 129–141.

Li Zhang and Danilo Ardagna. 2004. SLA based profit optimization in autonomic computing systems. In *Proceedings of the 2nd International Conference on Service Oriented Computing*. ACM, New York, NY, 173–182.

Qi Zhang, M. Zhani, Raouf Boutaba, and J. Hellerstein. 2014. Dynamic heterogeneity-aware resource provisioning in the cloud. *IEEE Trans. Cloud Comput.* 2, 1 (2014), 14–28.

Ying Zhang, Gang Huang, Xuanzhe Liu, and Hong Mei. 2010. Integrating resource consumption and allocation for infrastructure resources on-demand. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing*. IEEE, 75–82.

Qinghua Zheng, Rui Li, Xiuqi Li, Nazaraf Shah, Jianke Zhang, Feng Tian, Kuo-Ming Chao, and Jia Li. 2016. Virtual machine consolidated placement based on multi-objective biogeography-based optimization. *Future Gener. Comput. Syst.* 54 (2016), 95–122.

Received November 2015; revised April 2016; accepted August 2016