Workloads in the Clouds

Maria Carla Calzarossa, Marco L. Della Vedova, Luisa Massari, Dana Petcu, Mo'min I.M. Tabash, Daniele Tessera

Abstract Despite the fast evolution of cloud computing, up to now the characterization of cloud workloads has received little attention. Nevertheless, a deep understanding of their properties and behavior is essential for an effective deployment of cloud technologies and for achieving the desired service levels. While the general principles applied to parallel and distributed systems are still valid, several peculiarities require the attention of both researchers and practitioners. The aim of this chapter is to highlight the most relevant characteristics of cloud workloads as well as identify and discuss the main issues related to their deployment and the gaps that need to be filled.

Key words: Cloud computing; workload characterization; monitoring; resource management; scheduling; reliability; failure

1 Introduction

Cloud technologies are being successfully deployed nowadays in many business and scientific domains, such as e-commerce, e-government, engineering design and analysis, finance, healthcare, web hosting and online social networks. In particular, these technologies provide cost-effective scalable solu-

M.C. Calzarossa · L. Massari · M.I.M. Tabash
Dipartimento di Ingegneria Industriale e dell'Informazione, Università di Pavia, Pavia, Italy, email{mcc, luisa.massari}@unipv.it, momin.tabash01@ateneopv.it

M.L. Della Vedova · D. Tessera
Dipartimento di Matematica e Fisica, Università Cattolica del Sacro Cuore, Brescia, Italy, e-mail: {marco.dellavedova, daniele.tessera}@unicatt.it

D. Petcu
Department of Computer Science, West University of Timisoara, Timisoara, Romania, e-mail: petcu@info.uvt.ro

tions, thanks to the flexibility and elasticity in resource provisioning and the use of advanced virtualization and scheduling mechanisms [5, 13].

Cloud workloads consist of a collection of many diverse applications and services, each characterized by its own performance and resource requirements and by constraints specified in the form of Service Level Agreements (SLAs). A large number of factors affects cloud performance, including, among the others, the variability in the resource and network conditions and the highly dynamic nature of the workloads, whose intensity can suddenly grow or shrink as a consequence of the user interactions. More specifically, the use of virtualized time-shared resources could lead to performance degradation. This degradation is mainly due to the interference and resource contention arising from the co-location of heterogeneous workloads on the same physical infrastructure and to the overheads caused by the resource management policies being adopted. Similarly, the mix of workloads concurrently executed on a given virtual machine (VM) can be responsible for some unpredictable effects on the performance because of incompatible temporal patterns of the resource usage [74]. These performance issues could become even more critical in multi-cloud environments where the workload is distributed across different cloud infrastructures.

In these complex scenarios, mapping cloud resources to workload characteristics is very challenging [42]. Nevertheless, it is of primary importance for an effective deployment of cloud technologies and to achieve the desired service levels. Hence, to address resource management, provisioning and online capacity planning, and, more generally, to manage and predict performance and Quality of Service (QoS), it is essential to gain a deep understanding of the properties and the evolution of cloud workloads. Therefore, systematic and structured approaches towards workload characterization have to be considered as an integral component of all these strategies.

Despite their importance, the characterization and forecasting of cloud workloads have been addressed in the literature to a rather limited extent and mostly at the level of the VMs without taking into consideration the features of the individual workload components running on the VMs themselves. The aim of this chapter is to provide an overview of the main issues related to the entire lifecycle of workload deployment in cloud environments. More specifically, starting from the identification of the most relevant behavioral characteristics of cloud workloads, we define some broad workload categories described in terms of qualitative and quantitative attributes. The chapter then focuses on the various workload categories and discusses the challenges related to their monitoring, profiling and characterization. This thorough investigation of the state of the art is complemented by a literature review of the exploitation of scheduling strategies and failure analysis and prediction mechanisms the framework of cloud workloads.

The chapter is organized as follows. Section 2 presents the categories identified for cloud workloads, while Section 3 discusses the main issues related to their monitoring and profiling. The workload structures and resource require-

ments are addressed in Sections 4 and 5, whereas the challenges related to workload scheduling and failure analysis and prediction are briefly illustrated in Sections 6 and 7, respectively. Finally, Section 8 presents some concluding remarks.

2 Workload categories

The term workload refers to all inputs (e.g., applications, services, transactions, data transfers) submitted to and processed by an e-infrastructure. In the framework of cloud computing, these inputs usually correspond to online interactions of the users with web-based services hosted in the cloud or to jobs processed in batch mode. On the contrary, cloud workloads almost never refer to hard real-time applications.

In this section, we analyze the behavioral characteristics of cloud workloads (i.e., their qualitative and quantitative attributes) to identify some broad categories specified in terms of various dimensions, namely:

- Processing model.
- Architectural structure.
- Resource requirements.
- Non-functional requirements.

The choice of these dimensions is mainly driven by their role in the formulation of the cloud management strategies and in the assessment of the service levels foreseen by the workloads.

The *processing model* adopted by the workload, that is, online (i.e., interactive) and offline (i.e., batch or background), is an important high level dimension that identifies two workload categories. These categories are characterized by very diverse behaviors and performance requirements as well as by a different impact on management policies (e.g., resource scheduling, VM placement, VM migration). An *interactive workload* typically consists of short lived processing tasks submitted by a variable number of concurrent users, whereas a *batch workload* consists of resource intensive long lived tasks. Hence, as we will discuss later on, these workload categories exercise cloud resources to a rather different extent.

Another dimension chosen to classify cloud workloads focuses on their *architectural structure* expressed in the form of the processing and data flows characterizing each individual application. More precisely, these flows are described by the number and types of services or tasks being instantiated by a cloud application and their mutual dependencies, and, as such, have a strong impact on the scheduling policies. In particular, multiple task applications can be organized according to different models, namely:

- Pipeline model.
- Parallel model.

- Hybrid model.

In the *pipeline model*, tasks need to be processed sequentially one after the other with tight precedence constraints. On the contrary, in the *parallel model*, the tasks are characterized by precedence constraints that allow for concurrent execution of multiple tasks. In addition, these models are often combined in a sort of *hybrid* architectural *model* where the relationships among tasks are usually more complex. Figure 1 shows an example of a directed acyclic graph that represents the data flow of a simple cloud application organized according to a hybrid model. The nodes and edges denote the datasets and relationships between them, respectively.

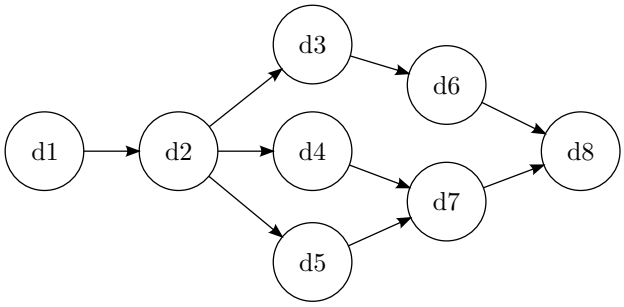

Fig. 1 Directed acyclic graph representing the data flow of a cloud application organized according to a hybrid model.

In the framework of scientific workloads, their description often relies on the *many-task computing* (MTC) paradigm [67], an architectural structure consisting of loosely coupled tasks and involving large volumes of data. Conversely, interactive cloud applications are typically organized according to *multi-tier* architectures. As we will discuss in Section 4, the interdependency among tiers and the patterns followed by the applications strongly affect the deployment of scaling strategies in cloud environments. Moreover, it is not always possible to derive a detailed view of the workload structure because of the lack of specific design information.

The definitions of workload architectural structures do not include any details about the behavioral characteristics of the workload at runtime (e.g., resource requirements, scheduling events). Nevertheless, *qualitative attributes* (e.g., priority, termination status) and *quantitative attributes* (e.g., workload intensity, demands and usage patterns of cloud resources) are very relevant to devise accurate resource allocation strategies. In particular, quantitative attributes provide a detailed characterization of the *computing*, *communication* and *storage requirements* of the workload and have to be assessed very carefully to avoid overprovisioning or underprovisioning of the resources (e.g., CPU, memory, I/O, network).

Depending on the amount of resources used, workloads are classified as:

- Compute or I/O intensive.
- Elastic or bandwidth sensitive.

Generally speaking, we can say that network bandwidth is more critical for online interactive workloads, whereas storage and computing resources often characterize batch workloads. Moreover, the *resource requirements* of some workloads are *stable* (i.e., uniformly distributed over their execution), whereas other workloads (e.g., workloads associated with online services) exhibit some specific *temporal patterns*, such as *periodic*, *bursting*, *growing* and *on/off*. These patterns typically depend on the intrinsic characteristics of the applications, as well as on the workload intensity. In details, patterns can refer to a single resource or multiple resources. A communication intensive phase can be followed by a compute intensive phase. Similarly, during the execution of an application, the bandwidth usage can change and follow some specific patterns.

As already pointed out, cloud workloads consist of streams of jobs and requests submitted at unpredictable times. Hence, their *arrival process* is seldom deterministic. It is often characterized by various effects (e.g., *diurnal* patterns, *seasonal* effects, *flash crowd* phenomena). In general, the *burstiness* in the workload intensity and heavy load conditions cause sudden and unexpected peaks in the resource demands that have a critical impact on resource provisioning strategies. Figure 2 shows two examples of qualitative patterns, namely, a diurnal pattern typically associated with the intensity of interactive workloads and a periodic pattern corresponding to CPU usage.

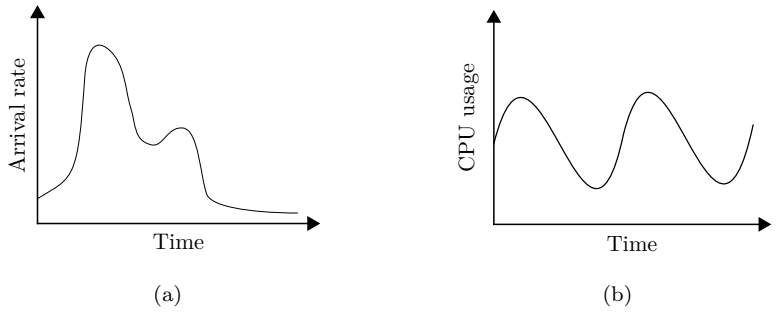

Fig. 2 Examples of a diurnal pattern characterizing the workload arrivals (a) and a periodic pattern describing the CPU usage (b).

An additional dimension describing the workload refers to *non-functional requirements* related to SLA constraints (e.g., performance, dependability, security). Among these attributes *reliability* is particularly important in cloud environments especially when deploying business-critical or safety-critical applications. Reliability denotes the probability that workloads can successfully

complete in a given time frame. The presence of *failures* decreases the reliability. Failures are due to various types of events (e.g., software bugs, exceptions, overflows, timeouts). For example, for data intensive workloads, a sudden increase in the rate at which data are submitted for processing can lead to failures, thus making the service unavailable. Moreover, failures are often correlated, that is, they often occur between dependent or co-located services or applications.

The remainder of the chapter focuses on the approaches typically adopted for monitoring and characterizing the workload categories presented in this section. The issues related to workload scheduling and failure analysis will also be discussed.

3 Workload monitoring and profiling

Monitoring and profiling are the basis for measuring the qualitative and quantitative attributes of the workloads. Generally speaking, monitoring keeps track of the activities performed by the workloads being processed and of the status of the allocated and the available resources. Profiling focuses on describing how workload exploits the cloud resources. Monitoring and profiling in the clouds are particularly difficult because of the heterogeneity and dynamicity of these environments [82]. Nevertheless, these activities play a critical role when addressing scenarios, such as:

- Capacity planning and resource management.
- Performance tuning.
- Billing.
- Security and troubleshooting.
- SLA verification.

Various approaches have been devised to tackle specific monitoring issues (e.g., measurement sources and accuracy, sampling granularity, intrusiveness and scalability). In what follows, we focus on the workload attributes that can be monitored at runtime to describe the resource usages. The level of details of the measurements collected in the clouds depends on the monitoring perspective adopted, namely, *cloud providers* and *cloud users*. Three basic types of cloud monitoring targets can be considered:

- Client.
- Virtual machine.
- Physical machine.

More specifically, cloud providers can measure resource usages of physical machines and of individual VMs from the vantage point of the hypervisor. On the other hand, cloud users are restricted to measure their own workloads using client logging and profiling facilities. Indeed, the VM isolation typical

of virtualization technologies hides the characteristics and performance of the underlying physical machines and the VM management policies. In details, to collect measurements on resource usage and cross-correlate them with application specific data and scheduling details, cloud users have often to resort to profiling facilities made available by providers (see, e.g., [28, 70]). To derive a more detailed description of the workloads being processed, VM measurements can be complemented with additional information about the workload structure, as well as with guest operating system statistics. Moreover, application logs are exploited to correlate the resource usages with workload intensity and characteristics.

Monitoring tools usually collect measurements by deploying distributed software agents that periodically gather information about the usage of resources, such as CPU, memory and I/O devices. In general, monitoring approaches rely on system tools and interfaces (e.g., `vmstat`, `iostat`, `netstat`) or on proprietary solutions [1, 45]. Moreover, depending on the monitoring capabilities of the virtualization technologies, ad-hoc scripts can be used for sampling low level quantitative attributes, such as CPU waiting times, number of virtual memory swaps, TLB flushes and interrupts [7]. The monitoring agents can also collect VM scheduling and provisioning events, (e.g., number and types of allocated VMs) [11]. The granularity and level of details of the measurements have to be chosen with the aim of limiting the monitoring intrusiveness. Measurements are usually stored into *tracelogs*, that is, collections of time stamped recordings of various types of information (e.g., resource demands, scheduling events, application specific data). Note that, despite the importance of workload measurements for both researchers and practitioners, cloud providers are seldom willing to publish detailed measurements about their own workloads often to prevent leakage of confidential competitive information.

Profiling is another approach applied to measure the resource usage of individual workload activities for driving performance tuning actions. In particular, profiling can be exploited by cloud users for optimal dynamic resource provisioning and by cloud providers for tuning VMs placement and scheduling policies [27]. In cloud environments, profiling has to cope with new challenges due to interference among co-located VMs. Indeed, the sharing of hardware resources could result in unpredictable behaviors of hardware components, such as cache memories, CPU pipelines and physical I/O devices [85]. Typical solutions for collecting profiling measurements are based on *dynamic instrumentation* and *sampling hardware performance counters*. An alternative approach is based on measuring at the *hypervisor level* the overall behavior of the VMs hosting the target applications. In details, the dynamic instrumentation takes advantage of software probes that selectively record runtime events about the application behavior (e.g., time stamps related to the execution of a given portion of an application). On the other hand, hardware based profiling exploits CPU performance monitoring unit for sampling counters related to low level events, such as cache misses, clock cycles per

instruction, pipeline stalls and branch mispredictions. In general, profiling can cause significant intrusiveness. Indeed, fine grained instrumentation and high sampling frequency result in large volume of measurements and perturbations of the workload behavior. On the contrary, coarse grain sampling and instrumentation could lead to ignore some rare though important events that might have a significant impact on the overall resource usages. To reduce the intrusiveness and the resource requirements of profiling activities various solutions, such as adaptive bursty tracing technique, based on a sampling rate inversely proportional to code execution frequency, have been devised [48].

Although monitoring and profiling are essential aspects of cloud computing, up to now no portable, general purpose and interoperable monitoring and profiling tools exist. This lack results in a plethora of open source and commercial tools addressing specific targets and platforms [15, 31]. Examples of *open source monitoring tools* are: `Nagios`[1], that is part of the OpenStack suite[2], `Ganglia`[3], `Collectl`[4] and `MonALISA`[5]. Cloud providers offer several *commercial tools* (e.g., `Amazon Cloudwatch`, `Microsoft Azure Watch`, `IBM Tivoli Monitoring`, `Rackspace`, `Rightscale`, `Cloudify`, `Aneka`). While these monitoring facilities are designed to be deployed to cloud environments, external monitoring services like `CloudHarmony`[6], `CloudSleuth`[7], `CloudClimate`[8] and `Up.time`[9], focus on monitoring applications and infrastructures from multiple locations on the Internet.

The development of a common framework for workload monitoring and profiling in the clouds is an open issue, that might also prevent users to deploy their businesses in these environments [39]. To improve the scalability and effectiveness of monitoring service consolidation and isolation, recent studies introduced the concept of *Monitoring as a Service* (MaaS) [59, 62]. The possibility for cloud users to monitor the global state of their applications is a challenging research question that deserves some further explorations.

4 Workload structures

In this section, we present a literature review of the most common structures of cloud workload introduced in Section 2, and the models used for their rep-

[1] http://nagios.sourceforge.net

[2] http://www.openstack.org

[3] http://ganglia.sourceforge.net

[4] http://collectl.sourgefourge.net

[5] http://monalisa.caltech.edu

[6] http://cloudharmony.com

[7] http://cloudsleuth.net

[8] http://www.cloudclimate.com

[9] http://www.suptimesoftware.com

resentation. Workload architectural structure is the description of the tasks an application consists of and their relationships. This structure is usually known at design time, whereas it can be difficult to derive it at runtime. Nevertheless, it is an important characteristic to be taken into account for the dynamic provisioning and optimal allocation of cloud resources and for identifying cost-effective solutions able to exploit the available parallelism (see, e.g., [14, 57, 84]). From the cloud provider perspective the aim is to maximize both resource utilization and energy savings, whereas from the cloud user perspective the aim is to minimize the operational costs while achieving optimal performance. The workload structures covered in this section refer to the following frameworks:

- MapReduce programming model.
- Workflow technologies.
- Many task computing paradigm.
- Multi-tier architecture.

These structures are not restricted to a single processing model. For example, a multi-tier architecture can be exploited for both batch and interactive applications. In details, *batch applications* often consist of tasks with parent-child relationships. These applications are modeled as workflows describing the tasks in terms of data dependencies and data and control flows. As stated in Section 2, typical workflow schemes are pipeline, parallel and hybrid, that is, sequential, concurrent and combinations of sequential and concurrent tasks, respectively.

Several approaches have been proposed to take advantage of the concurrency of application workflows. For example, *MapReduce* [24] is a programming model introduced to ease the exploitation of the parallelism in big data analytic workflows. Applications based on this paradigm are executed according to a hybrid structure consisting of multiple concurrent tasks (i.e., map and reduce workers), as illustrated in Figure 3. The intermediate data shuffle addresses the data dependencies of the workflow. To describe and predict interarrival times and resource demands of MapReduce workloads, statistical techniques, such as kernel canonical correlation analysis and probabilistic distribution fitting, have been proposed [6, 33]. Due to the heterogeneity of the application domains in which MapReduce is exploited, the workloads are often characterized in terms of different attributes (e.g., workload intensity, task durations and constraints) [20, 71]. Very popular cloud technologies based on MapReduce (i.e., Apache Hadoop[10], Spark[11]) perform automatic optimizations and data distributions. However, the deployment of Hadoop applications requires the tuning of many configuration parameters that might heavily affect the overall performance [87]. On the other hand, Spark applications take advantage of in-memory computations to reduce the overhead of Hadoop distributed file system [44].

[10] http://hadoop.apache.org

[11] http://spark.apache.org

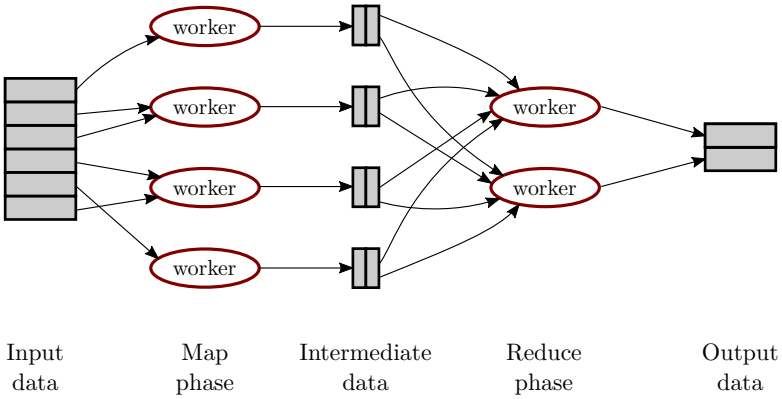

Fig. 3 Overview of MapReduce programming model.

In the framework of *scientific computing*, workflow technologies are an approach for easy and efficient development of applications with hybrid structures. In the literature, the workflow of these applications has been analyzed in terms of resource demands (e.g., number of tasks and their CPU, memory and I/O demands) [47, 92]. Similarly to workflow technologies, the *many task computing* paradigm is widely used to develop distributed loosely coupled scientific applications. MTC applications typically require over short time periods a large amount of computational resources to process the so-called bag-of-tasks. Hence, MTC is well suited to take advantage of dynamic provisioning of cloud resources. The studies related to the deployment of these applications on the clouds mainly focus on performance analysis of various types of infrastructures, such as commercial cloud computing services and federated clouds [61, 72]. In particular, performance and resource demands of scientific MTC applications have been investigated by analyzing workload tracelogs collected in environments other than clouds (e.g., parallel production infrastructures, grids). The behavior of scientific workflows is characterized in [40] in terms of number of jobs and of bag-of-tasks to identify the bottleneck in the resources. Workload tracelogs have also been analyzed for developing strategies aimed at reducing the impact of transient failures on the overall behavior of MTC applications [17]. These strategies, based on checkpoint and speculative execution policies, reduce the large overheads due to the entire bag-of-tasks resubmission, although they might affect resource usages with unnecessary duplicated task executions. It is worth noting that clouds can be a cost-effective and scalable alternative to the traditional high performance computing environments for a large variety of scientific applications, even though performance can be an issue. Indeed, bandwidth and jitters on network delays are among the most critical factors that limit the performance of scientific applications [58].

Regarding *interactive workloads*, that usually need to cope with the dynamic behavior of users, cloud computing is mainly adopted for deploying large scale applications in domains, such as e-commerce, financial services, healthcare, gaming and media servers. A common solution to address these highly variable load conditions is based on *multi-tier architectures*, where each tier, deployed on one or multiple VMs, addresses a specific functionality (e.g., web, database, application logic, load balancing). As an example, Figure 4 depicts an architecture of a five-tier web application. The advantage of this solution is the possibility of dynamically scaling each tier independently, both horizontally and vertically. Horizontal scaling deals with varying the number of VM instances (see Fig. 4 (b)). On the contrary, vertical scaling is about varying the amount of resources allocated to individual VMs. Figure 4 (c) shows that the VM deploying the web server scales up and doubles its number of cores.

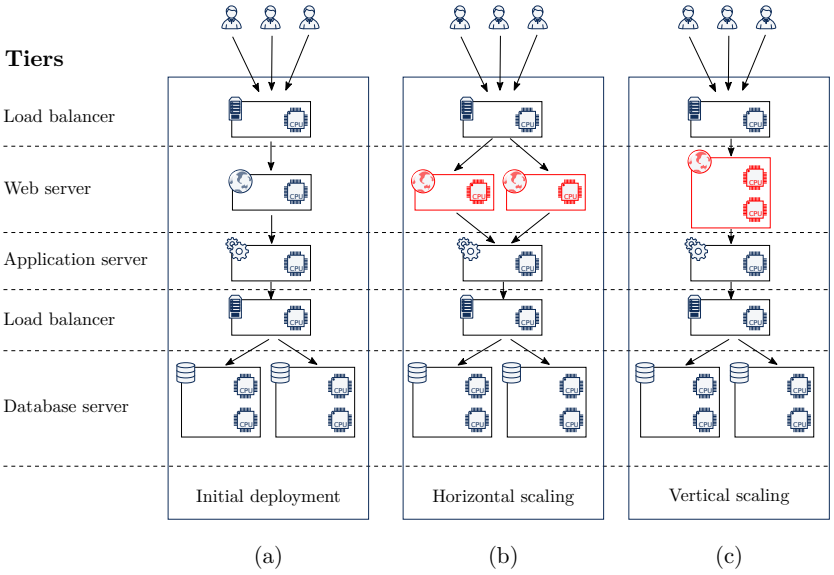

Fig. 4 Example of a five-tier architecture typical of large-scale web applications with its initial deployment (a), and the horizontal (b) and vertical (c) scaling of the web server tier, respectively.

Resource provisioning for multi-tier architectures is challenging because of the functional interdependence among the tiers and the network overhead. Therefore, the sizing of each tier plays a critical role for this kind of applications [41]. Moreover, it is difficult to model multi-tier applications due to the dynamic and unpredictable behavior of their users. In this framework, resource provisioning and scaling have been investigated using stochastic models based on queuing networks and control theory [38]. For vertical scaling,

linear regression methods, Markov chains and queuing network models are often used to represent the relationships between workload being processed and resource demands. Many studies focus on probability distributions and statistical bound techniques to derive performance metrics of cloud environments, such as response time, throughput and resource utilizations (see, e.g., [10, 75]). These metrics are used to characterize the workload, predict its behavior and scale resources accordingly. An alternative approach is based on multi-input multi-output control systems, where inputs are the resources allocated to each tier and outputs are the measured performance metrics [93]. For horizontal scaling, reactive heuristics leveraging a threshold-based set of rules are commonly used. Thresholds on resources utilization trigger the start or the shutdown of VMs in order to ensure given QoS levels [53]. Moreover, proactive approaches for resource provisioning take into account the resource demands as a function of the workload intensity. For example, queuing networks can be used for modeling the relationships between number and characteristics of allocated VMs and metrics, such as blocking and immediate service probabilities [51]. Additionally, optimal resource provisioning has been addressed by means of queuing network and simulation models [36, 37].

5 Workload attributes

In this section, we present a literature review of the approaches typically applied to characterize cloud workloads in terms of both *qualitative attributes* related to jobs and tasks events and *quantitative attributes* describing workload intensity and the demands of cloud resources (i.e., computing, communication and storage). These attributes are usually obtained from historical data (e.g., tracelogs) and runtime measurements. It is important to point out that tracelogs published by Google [69] are among the few publicly available cloud measurements. These logs store both qualitative and quantitative anonymized attributes about the jobs executed on a large cluster (i.e., demands and actual usages of CPU, memory and disk space, scheduling events of jobs and tasks).

The workload models obtained as a result of characterization studies are very useful when addressing the optimization of resource usage, the definition of scheduling policies and energy aware solutions, the prediction of failures and many other cloud management issues. In the literature, cloud workloads are characterized by focusing on jobs and tasks and analyzing their attributes, referring to:

- Resource usages.
- Workload intensity.

Commonly adopted approaches are based on various types of techniques, often used in combination, such as:

- Statistical and numerical techniques.
- Stochastic processes.

Some papers [55, 68] analyze the *resource usages* and their dynamics at *job* and *task* levels, by applying a statistical approach based on a high level *exploratory analysis* (i.e., descriptive statistics, empirical distributions of resource usages, visual inspection of their temporal behavior). These studies rely on the Google tracelogs. In particular, patterns of task submissions, interarrival times, relationships between resource usage and task status (i.e., killed, normally terminated, failed) are considered. For example, the identification of jobs resubmitted because of failures or evictions provides some interesting insights for predicting the resources actually required by the workload.

In order to derive realistic models that capture the heterogeneity of jobs and tasks, more advanced *statistical* and *numerical techniques* (e.g., clustering, fitting) are adopted. *Clustering* techniques are usually applied to identify groups of workload components characterized by similar behaviors. Early papers [21, 60] classify jobs and tasks based on their CPU and memory usage. In particular, a medium grain classification of tasks highlights the presence of few tasks that consume a large amount of resources. More recently, the statistical properties of the workload are analyzed to classify cloud applications in terms of both quantitative (i.e., resource requirements) and qualitative (i.e., task events) attributes [25]. In general, job and task classification has been applied for devising scheduling and allocation policies. For example, the approach proposed in [66] estimates the resource demands of tasks and predicts the cluster to which a new arriving job belongs to according to its initial resource demands. As a consequence, resource utilization and energy saving can be improved. In [9] clustering is applied to identify tasks characterized by similar memory and CPU usages, as well as tasks whose memory usage is independent from their CPU usage. Moreover, this study analyses the dynamics of the CPU usage to discover weekly and daily patterns and in particular synchronized peaks whose presence is important for devising more efficient allocation strategies.

As pointed out in Section 3, it is difficult to obtain detailed measures on resource usages of the *specific workload components*. Most studies rely on measurements collected at the *VM level*. Although these measurements refer to the overall resource usage of individual VMs, they provide an accurate description of the application behavior in virtualized environments. Understanding and modeling this behavior are important in many domains, such as workload scheduling, VM failure monitoring and intrusion detection. To highlight the variability in resource usage and the presence of temporal patterns, some studies combine statistical metrics (e.g., correlation between attributes) with auto-correlation functions and time series analysis [7, 76]. The evolution of CPU, memory and disk utilizations is analyzed in [11] by representing their dynamics and fluctuations as a time series at different time scales. *Numerical fitting* techniques are applied to build models that capture

the temporal variability in resource usages. Moreover, by looking at the correlations among resource usages, dependencies to be exploited in the design of effective consolidation strategies are identified. A time series approach is also adopted in [50] to represent CPU usage patterns. Additionally, a co-clustering technique identifies groups of VMs with workload patterns, whereas a Hidden Markov Model predicts the changes of these patterns.

Workload intensity is another important aspect extensively analyzed in the literature because of its strong impact on cloud performance. In [78] workload intensity is quantified in terms of task submission rate and clustering is applied to highlight variability in the submission rate across groups of tasks. Other papers model the workload intensity by means of *stochastic processes*. It has been shown that simple Poisson processes generating independent identically distributed interarrival times are not suited to represent real cloud workloads [46]. *Burstiness*, a well-known characteristic of network traffic, has also been observed in cloud environments. Bursty and fractal behaviors of the arrival processes affect in particular load balancing strategies [81]. In addition, detecting, measuring and predicting these phenomena are important for devising efficient resource provisioning and energy saving strategies. To describe the time varying behavior and self-similar effects, metrics, such as index of dispersion and coefficient of variation, are complemented with models based on 2-state Markovian Arrival Processes, parameterized with different levels of burstiness [88]. The two states represent the bursty and non-bursty request arrival processes, respectively (see Fig. 5). Markovian Arrival Processes are integrated in [64] with analytical queueing models to predict system performance. A different approach based

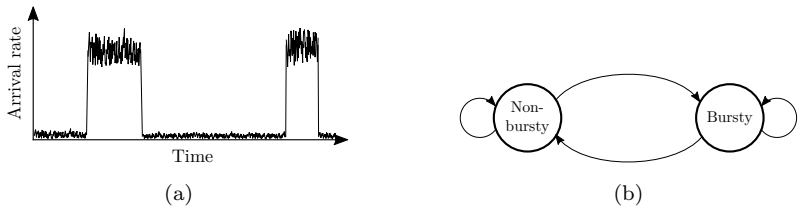

Fig. 5 Example of non-bursty and bursty arrivals (a) and the corresponding 2-state Markovian Arrival Process (b).

on fractal techniques is proposed in [16, 35] for representing workload dynamics in terms of job arrivals. The arrival process is modeled using fractional order differential equations with time dependent parameters, whereas fitting is applied to identify statistical distributions for CPU and memory usages.

The literature review presented in this section highlights the importance of taking into account workload characteristics to effectively deploy cloud technologies. Even though different approaches to workload characterization in cloud environments have been proposed, few studies focus on the attributes

of the individual workload components. In addition, there is the need to devise more systematic approaches towards workload characterization. The lack of publicly available workload measurements makes quite difficult to investigate real life cloud computing scenarios.

6 Workload scheduling

Workload scheduling, i.e., the mapping between jobs/tasks and VMs, is a challenging issue in cloud environments because of the heterogeneity of workload characteristics (e.g., intensity and resource demands). The problem of finding an *optimal mapping* is *NP-complete* and therefore intractable with exact methods when the number of VMs and tasks is large, as it is typically the case of cloud environments. For this reason, (meta-)heuristics are currently used to find sub-optimal solutions. Meta-heuristics based on methods, such as neural networks, evolutionary algorithms or set-of-rules, are proved to be efficient in solving optimization problems related to scheduling. In the remainder of this section, we review the literature (see Table 1 for an overview) by briefly discussing the following aspects of workload scheduling:

- Scheduling objectives.
- Optimization approaches.
- Resource scaling.
- Load balancing.
- Scheduling of real-time applications.

Objectives of the scheduling problem are multiple (e.g., to minimize makespan, data transfer, energy consumption and economic cost, to satisfy SLAs). A simple approach takes into account one objective at a time. Alternative approaches are aimed at combining multiple objectives into a single aggregate objective function (see, e.g., [79]) or considering multi-objective algorithms (see, e.g., [29, 90]). A recent survey summarizes the evolutionary approaches for scheduling in cloud environments [89]. The different viewpoints for scheduling and the corresponding objectives are identified as follows:

- *Scheduling for user QoS*, where objectives include the makespan and user costs minimization, application performance and reliability.
- *Scheduling for provider efficiency*, where objectives are load balancing, utilization maximization and energy savings.
- *Scheduling for negotiation*, where the goal is to satisfy both user and provider objectives at the same time.

Exact methods for solving the optimization problem (e.g., constrained binary integer programming) can be used in simple scenarios only, such as trivial parallel workloads where tasks are fully decoupled without any precedence constraint [83]. For more general workload structures, the problem complexity

Reference	**Workload structure**	**Method**	**Optimization objectives**	**Deployment model**
Somasundaram and Govindarajan [79]	Tasks with deadlines	Particle swarm	Execution time and economic cost	Private
Duan et al. [29]	MTC	Game theory (multi-objective)	Makespan and economic cost	Hybrid
Zhang et al. [90]	MTC	Vectorized ordinal optimization (multi-objective)	Makespan and economic cost	Private
Zhan et al. [89]	Survey on evolutionary approaches to scheduling			
Van den Bossche et al. [83]	Tasks with deadlines	Binary integer programming	Resource utilization	Hybrid
Pandey et al. [65]	Workflow	Particle swarm	Makespan	Public
Kessaci et al. [49]	Tasks with deadlines	Pareto-based genetic algorithm (multi-objective)	Energy consumption, carbon emission and profit	Federated
de Oliveira et al. [63]	Independent tasks	Ant colony	Makespan and load balancing	Federated
Wu et al. [86]	Survey on workflow scheduling			
Jiang et al. [43]	Workflow	Path clustering heuristics and list based scheduling	Makespan and resource utilization	Private (High Performance Computing)
Zhang et al. [91]	Independent tasks with priorities	Model predictive control with heuristics	Energy consumption, scheduling delay and economic cost	Public and Private
Mao and Humphrey [56]	Workflow	Heuristics	Economic cost	Public and Private
Dutta et al. [30]	Interactive	Decision tree	Resource usage	Public and Private
Ardagna et al. [4]	Interactive	Reactive set of rules	Resource usage	Public and Private
Cheng et al. [22]	Interactive and batch	Nonlinear optimization	QoS and load balancing	Hybrid
Singh et al. [77]	Interactive multi-tier	Clustering and queueing	QoS	Public
Spicuglia et al. [80]	Interactive	Reactive heuristic with thresholds	Response time, resource utilization and load balancing	Public and Private
Li et al. [52]	Real-time tasks	Heuristic with penalties on deadline	Economic cost	Public and Private
Liu et al. [54]	Real-time tasks	Heuristic with eviction	Economic cost	Public and Private

Table 1 Summary of the state of the art on workload scheduling in cloud environments. References are ordered as they appear in the text.

increases and it is necessary to devise *heuristic* optimization methods, such as particle swarm optimization [65], genetic algorithms [49], ant colony optimization [63] and game theoretic algorithms [29]. Another recent survey [86], focusing on the main issues related to *workflow scheduling*, subdivides the scheduling methods into three main categories, namely:

- *Static scheduling*, where workload structure is known a priori and resources have instantaneous availability.
- *Dynamic scheduling*, where workload structure can be obtained at runtime.
- *Static planning with dynamic scheduling*, where the structure and communication time can be estimated. Tasks are statically planned, although dynamically scheduled to resources at runtime.

Examples of offline methods for static scheduling multi-tenant workflows in cloud environments are presented in [43]. These methods take advantage of gaps in the schedule due to communication overheads and task dependencies. In particular, the gap search is performed on the entire task group or in a distributed fashion by working on its partitions. The scheduling problem can be even more complex when *task priorities* are considered [91]. Low-priority tasks are often evicted because of the overcommitment of physical resources. Moreover, changes in the cloud environment properties can affect the priorities of jobs and tasks.

Job scheduling and *resource scaling* are often considered in conjunction [56]. Several frameworks have been recently introduced to address resource scalability. For example, SmartScale [30] is an automated scaling framework that uses a combination of vertical and horizontal approaches to optimize both resource usages and reconfiguration overheads. Scaling mechanisms are also encountered in [4] where different scalability patterns are considered and an approach to performance monitoring that allows automatic scalability management is proposed. Auto-scaling is often used in conjunction with *load balancing* strategies. Even though physical machines are often the main target of these strategies, effective load balancing and resource allocation policies take into account the concurrent execution of different application types, i.e., interactive, batch, and the mix of applications with different resource requirements and workload structures (see Sect. 4) [22, 77, 80].

Hard real-time applications (i.e., applications characterized by hard deadlines that are a-priori guaranteed to be met) are not well suited to the current cloud infrastructures. In fact, the virtualization technologies and network protocols used in the clouds are not designed to provide the timing guarantees required by these applications. However, the so-called *soft deadlines* are often taken into account by the schedulers because of the penalties associated with the negotiated SLAs [52, 54]. Despite hard real-time applications, for online services hosted in cloud environments the main goal of the scheduling is to maximize the profit by providing timely services [52].

The analysis of the state of the art presented in this section has shown that workload scheduling in the clouds is a very important research field.

Although there are numerous studies on workload scheduling on parallel and distributed systems, few papers address real cloud environments, and even fewer cloud workload management systems. Nevertheless, all these topics need further investigation.

7 Workload failures

As described in the previous sections, workloads typically consist of diverse applications with different priorities and deadlines that reflect the user requirements. Unforeseen workload behaviors or incompatibility between workload requirements and the resources offered in the clouds result in *failures*. Increasing functionality and complexity of cloud environments are leading to inevitable failures that can be caused by different types of events, such as outage, vulnerability and automatic updates [32]. Other examples of failures are: software crashes due to hidden bugs, out of memory exceptions due to the lack of resources, denial of service due to malicious activities, deadline violations due to unexpected processing delays. There are also failures caused by unknown events.

A decrease in the reliability associated with the workload does not necessarily mean that the applications are not successfully completed because of bugs. The failure rate often depends on the workload intensity and mixes. In particular, heavy load conditions are often responsible of the increase of the overall failure rate. All failures and in particular deadlines violations are crucial in cloud environments because of their negative impact on QoS and SLA. Hence, whenever a SLA has been established between a cloud provider and a cloud user, various strategies, such as replication and checkpointing, have to be deployed in order to cope with failures.

In the literature (see Table 2 for an overview) cloud failures have been addressed under two different perspectives, namely:

- Failure analysis.
- Failure prediction.

In particular, to prevent wasting resources, avoid performance degradation and reduce costs and energy consumption, in the last years, extensive research has focused on *failure analysis*. Failures are characterized using different statistical and analytical techniques focused on resource usage (e.g., CPU, memory, disk I/O, bandwidth) and on other workload qualitative attributes (e.g., priority, termination status). The basis of these analyses is often represented by the large variety of workload information collected in cloud production environments (see Sect. 3). For example, the analysis of the Google tracelog presented in [34] focuses on the characteristics of failures of cloud workloads. This empirical study considers the failure and repair times, and, in particular, two important metrics, namely:

- Mean Time Between Failure (MTBF).
- Mean Time To Repair (MTTR).

More specifically, the statistical properties of these metrics together with the theoretical distributions that best fit the empirical data (e.g., Weibull, lognormal) are the basis for characterizing the behavior of the failures. The study shows that, in general, the workload failure rates vary significantly and depend on the priority associated with the individual tasks, thus reflecting the diversity in the workload characteristics. The Google tracelog is also analyzed in [18] to evaluate the effects exercised on failures by workload attributes, such as job and task resource usage, task resubmission for single and multiple task jobs and termination statuses. In addition, the study investigates the relationships between user behavior and failures. Clustering techniques have been applied to identify groups of users submitting jobs with similar characteristics and termination status, thus exhibiting similar reliability properties. Transient failures associated with scientific workflows are investigated in [17] by modeling failure interarrival times and system overheads.

Reference	**Target**	**Failure type**	**Parameters**	**Modeling Approach**
Garraghan et al. [34]	Google tracelog	Task and server	Failure and repair times and task termination status	Probabilistic
Chen et al. [18]	Google tracelog	Job and task	Job and task attributes	Statistical and probabilistic
Chen et al. [17]	Scientific workflows	Transient	Task runtime and failure interarrival time	Probabilistic
Di Martino et al. [26]	Cloud data	Operational	Failure rate and MTBF	Probabilistic
Chen et al. [19]	Google tracelog	Job and task	Resource usage, task priority and resubmission	Machine learning
Samak et al. [73]	Scientific workflows	Job	VM attributes	Machine learning
Bala and Chana [8]	Scientific workflows	Task	Resource utilizations	Machine learning

Table 2 Summary of the state of the art in the field of cloud failure analysis and prediction. References are ordered as they appear in the text.

Failed jobs typically consume a significant amount of resources. Hence, it is crucial to mitigate their negative impact by predicting failures in a timely manner. In [26] the operational failures of a business data processing platform are characterized to estimate common failure types, their rates and relationships with the workload intensity and data volume. In addition, a

trend analysis is performed to assess whether failure arrivals significantly change over time.

Failure prediction usually relies on machine learning techniques, such as Naive Bayes, Random Forest, and Artificial Neural Networks. In particular, Recurrent Neural Networks are applied in [19] to predict the failures of jobs and tasks by analyzing their resource usages. In the framework of large scale scientific applications represented as *workflows*, Naive Bayes classifiers are used to study the behavior of jobs and predict their failure probability [73]. Similarly, failure prediction models for tasks in workflow applications are proposed in [8]. These models rely on various machine learning approaches and are the basis of proactive fault tolerant strategies for failure prediction to be used for the identification of tasks that could fail due to the overutilization of resources (e.g., CPU, storage).

A special category of failures is related to *software aging*. The presence of these failures is manifested as either an increase in their rate or in performance and QoS degradations. Typical causes of software aging failures are elusive bugs, such as memory leaks, unterminated threads and unreleased locks. The effects of these bugs usually become evident whenever peaks and bursts appear in the workload. A common solution to cope with these problems is represented by *software rejuvenation*, that is, a cost-effective software maintenance technique based on preventive rollbacks of continuously running applications. A recent survey [2] presents an interesting classification of the most common approaches used in this framework (see Fig. 6). A detailed

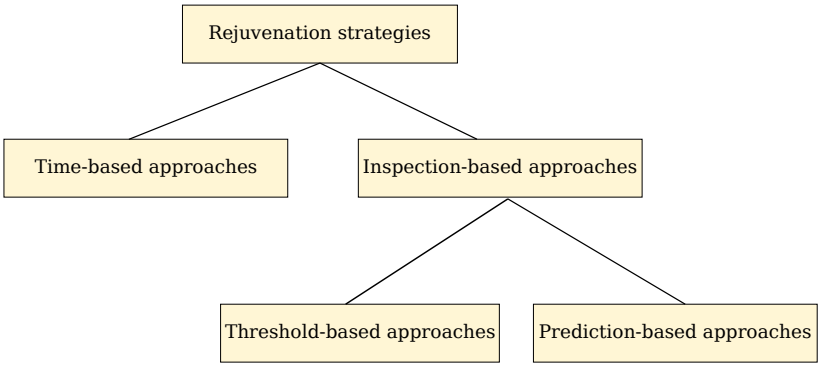

Fig. 6 Classification of the main rejuvenation strategies.

overview of the analysis techniques proposed in the literature for software aging and rejuvenation (e.g., stochastic processes, time series analysis, machine learning) is offered in [23]. In particular, in cloud environments, software rejuvenation can be applied to either individual VMs and to the hypervisor. Techniques based on live VM migrations and checkpointing are often exploited to reduce downtimes due to failures [12]. Similarly, to reduce the

downtime during the rejuvenation, time series approaches are used to predict the proper time to trigger the process [3]. In detail, to guarantee a safe scheduling of rejuvenation actions, the resource-aware rejuvenation policy introduced in the paper considers multiple thresholds referring to the resource usages (e.g., virtual memory).

Despite the effort already put in the domain of cloud failure analysis and prediction, some open challenges remain to be investigated. In particular, to improve workload reliability in the clouds, failure awareness resource provisioning and integration of failure prediction mechanisms in the schedulers should be devised.

8 Conclusions

A deep understanding of workload properties and behavior is essential for an effective deployment of cloud technologies and for achieving the desired service levels. In this chapter we discussed the main issues related to the entire lifecycle of the workloads in the clouds, starting with their characterization at the design time (i.e., workload categories, structures and patterns), their matching at the deployment phase (i.e., resource requirements and scheduling) and the issues in the execution phase (i.e., failure analysis and prediction).

The list of topics and issues related to cloud workloads presented in this chapter does not pretend to be exhaustive. However, the snapshot of the state of the art gathers in one place the pointers to many different approaches and can be therefore seen as a starting point in the design of a comprehensive framework dealing with all stages of the workload lifecycle. In particular, the analysis of the literature suggests some interesting research challenges dealing with the design and the development of:

- Portable frameworks for workload monitoring and profiling.
- Systematic approaches towards workload characterization to be exploited in resource management strategies.
- Management systems for workload scheduling in real cloud environments that address the heterogeneity and variability in the resource requirements.
- Failure-aware resource provisioning and scheduling mechanisms that improve workload reliability.

Finally, a major issue faced by the research in cloud environments is the lack of publicly available large-scale workload measurements. In general, providers and users are very reluctant to disclose data about their workloads to avoid leakage of competitive and confidential information. Nevertheless, the availability of this data would be very beneficial for accelerating cloud deployments.

References

1. Alhamazani, K., Ranjan, R., Mitra, K., Rabhi, F., Jayaraman, P., Khan, S., Guabtni, A., Bhatnagar, V.: An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art. Computing **97**(4), 357–377 (2015)
2. Alonso, J., Trivedi, K.: Software Rejuvenation and its Application in Distributed Systems. In: D. Bruneo, S. Distefano (eds.) Quantitative Assessments of Distributed Systems: Methodologies and Techniques, pp. 301–325. Wiley (2015)
3. Araujo, J., Matos, R., Alves, V., Maciel, P., de Souza, F.V., Matias, R.J., Trivedi, K.: Software Aging in the Eucalyptus Cloud Computing Infrastructure: Characterization and Rejuvenation. ACM Journal on Emerging Technologies in Computing Systems **10**(1), 11:1–11:22 (2014)
4. Ardagna, C., Damiani, E., Frati, F., Rebeccani, D., Ughetti, M.: Scalability Patterns for Platform-as-a-Service. In: Proc. of the 5th Int. Conf. on Cloud Computing - CLOUD'12, pp. 718–725. IEEE (2012)
5. Armbrust, M., Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: A View of Cloud Computing. Communications of the ACM **53**(4), 50–58 (2010)
6. Atikoglu, B., Xu, Y., Frachtenberg, E., Jiang, S., Paleczny, M.: Workload Analysis of a Large-scale Key-value Store. In: Proc. of the 12th ACM SIGMETRICS/PERFORMANCE Joint Int. Conf. on Measurement and Modeling of Computer Systems, pp. 53–64. ACM (2012)
7. Azmandian, F., Moffie, M., Dy, J., Aslam, J., Kaeli, D.: Workload Characterization at the Virtualization Layer. In: Proc. of the 19th Int. Symp. on Modeling, Analysis Simulation of Computer and Telecommunication Systems - MASCOTS'11, pp. 63–72. IEEE (2011)
8. Bala, A., Chana, I.: Intelligent failure prediction models for scientific workflows. Expert Systems with Applications **42**(3), 980–989 (2015)
9. Beaumont, O., Eyraud-Dubois, L., Lorenzo del Castillo, J.: Analyzing Real Cluster Data for Formulating Allocation Algorithms in Cloud Platforms. In: Proc. of the 26th Int. Symp. on Computer Architecture and High Performance Computing - SBAC-PAD, pp. 302–309. IEEE (2014)
10. Bi, J., Zhu, Z., Tian, R., Wang, Q.: Dynamic Provisioning Modeling for Virtualized Multi-tier Applications in Cloud Data Center. In: Proc. of the 3rd Int. Conf. on Cloud Computing - CLOUD'10, pp. 370–377. IEEE (2010)
11. Birke, R., Chen, L., Smirni, E.: Multi-Resource Characterization and their (In)dependencies in Production Datacenters. In: Proc. of the Network Operations and Management Symposium - NOMS'14. IEEE (2014)
12. Bruneo, D., Distefano, S., Longo, F., Puliafito, A., Scarpa, M.: Workload-Based Software Rejuvenation in Cloud Systems. IEEE Transactions on Computers **62**(6), 1072–1085 (2013)
13. Buyya, R., Yeo, C., Venugopal, S., Broberg, J., Brandic, I.: Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems **25**(6), 599–616 (2009)
14. Byun, E.K., Kee, Y.S., Kim, J.S., Maeng, S.: Cost optimized provisioning of elastic resources for application workflows. Future Generation Computer Systems **27**(8), 1011–1026 (2011)
15. Calero, J., Aguado, J.G.: Comparative analysis of architectures for monitoring cloud computing infrastructures. Future Generation Computer Systems **47**, 16–30 (2015)
16. Chen, S., Ghorbani, M., Wang, Y., Bogdan, P., Pedram, M.: Trace-Based Analysis and Prediction of Cloud Computing User Behavior Using the Fractal Modeling Technique. In: Proc. of the 7th Int. Congress on Big Data, pp. 733–739. IEEE (2014)
17. Chen, W., Ferreira da Silva, R., Deelman, E., Fahringer, T.: Dynamic and Fault - Tolerant Clustering for Scientific Workflows. IEEE Transactions on Cloud Computing (2015)

18. Chen, X., Lu, C., Pattabiraman, K.: Failure Analysis of Jobs in Compute Clouds: A Google Cluster Case Study. In: Proc. of the 25th Int. Symp. on Software Reliability Engineering - ISSRE'14, pp. 167–177. IEEE (2014)
19. Chen, X., Lu, C.D., Pattabiraman, K.: Failure Prediction of Jobs in Compute Clouds: A Google Cluster Case Study. In: Proc. of the IEEE Int. Symp. on Software Reliability Engineering Workshops - ISSREW'14, pp. 341–346 (2014)
20. Chen, Y., Alspaugh, S., Katz, R.: Interactive Analytical Processing in Big Data Systems: A Cross-industry Study of MapReduce Workloads. Proceedings of the VLDB Endowment **5**(12), 1802–1813 (2012)
21. Chen, Y., Ganapathi, A., Griffith, R., Katz, R.: Analysis and Lessons from a Publicly Available Google Cluster Trace. Tech. Rep. UCB/EECS-2010-95, EECS Department, University of California, Berkeley (2010)
22. Cheng, D., Jiang, C., Zhou, X.: Heterogeneity-Aware Workload Placement and Migration in Distributed Sustainable Datacenters. In: Proc. of the 28th Int. Symp. on Parallel and Distributed Processing - IPDP'14, pp. 307–316. IEEE (2014)
23. Cotroneo, D., Natella, R., Pietrantuono, R., Russo, S.: A Survey of Software Aging and Rejuvenation Studies. ACM Journal on Emerging Technologies in Computing Systems **10**(1), 8:1–8:34 (2014)
24. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM **51**(1), 107–113 (2008)
25. Di, S., Kondo, D., Cappello, F.: Characterizing and modeling cloud applications/jobs on a Google data center. The Journal of Supercomputing **69**(1), 139–160 (2014)
26. Di Martino, C., Kalbarczyk, Z., Iyer, R., Goel, G., Sarkar, S., Ganesan, R.: Characterization of Operational Failures from a Business Data Processing SaaS Platform. In: Proc. of the 36th Int. Conf. on Software Engineering Companion - ICSE'14, pp. 195–204. ACM (2014)
27. Do, A., Chen, J., Wang, C., Lee, Y., Zomaya, A., Zhou, B.B.: Profiling applications for virtual machine placement in clouds. In: Proc. of the 4th Int. Conf. on Cloud Computing - CLOUD'11, pp. 660–667. IEEE (2011)
28. Du, J., Sehrawat, N., Zwaenepoel, W.: Performance Profiling of Virtual Machines. ACM SIGPLAN Notices **46**(7), 3–14 (2011)
29. Duan, R., Prodan, R., Li, X.: Multi-Objective Game Theoretic Scheduling of Bag-of-Tasks Workflows on Hybrid Clouds. IEEE Transactions on Cloud Computing **2**(1), 29–42 (2014)
30. Dutta, S., Gera, S., Verma, A., Viswanathan, B.: Smartscale: automatic application scaling in enterprise clouds. In: Proc. of the 5th Int. Conf. on Cloud Computing - CLOUD'12, pp. 221–228. IEEE (2012)
31. Fatema, K., Emeakaroha, V., Healy, P., Morrison, J., Lynn, T.: A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives. Journal of Parallel and Distributed Computing **74**(10), 2918–2933 (2014)
32. Fiondella, L., Gokhale, S., Mendiratta, V.: Cloud Incident Data: An Empirical Analysis. In: Proc. of the Int. Conf. on Cloud Engineering - IC2E'13, pp. 241–249. IEEE (2013)
33. Ganapathi, A., Yanpei, C., Fox, A., Katz, R., Patterson, D.: Statistics-driven workload modeling for the Cloud. In: Proc. of the 26th Int. Conf. on Data Engineering Workshops - ICDEW'10, pp. 87–92. IEEE (2010)
34. Garraghan, P., Townend, P., Xu, J.: An Empirical Failure-Analysis of a Large-Scale Cloud Computing Environment. In: Proc. of the 15th Int. Symp. on High-Assurance Systems Engineering - HASE'14, pp. 113–120. IEEE (2014)
35. Ghorbani, M., Wang, Y., Xue, Y., Pedram, M., Bogdan, P.: Prediction and Control of Bursty Cloud Workloads: A Fractal Framework. In: Proc. of the Int. Conf. on Hardware/Software Codesign and System Synthesis - CODES'14, pp. 12:1–12:9. ACM (2014)

36. Grozev, N., Buyya, R.: Performance Modelling and Simulation of Three-Tier Applications in Cloud and Multi-Cloud Environments. The Computer Journal **58**(1), 1–22 (2015)
37. Han, R., Ghanem, M., Guo, L., Guo, Y., Osmond, M.: Enabling cost-aware and adaptive elasticity of multi-tier cloud applications. Future Generation Computer Systems **32**, 82–98 (2014)
38. Huang, D., He, B., Miao, C.: A Survey of Management in Multi-Tier Web Applications. IEEE Communications Surveys Tutorials **16**(3), 1574–1590 (2014)
39. Huang, J., Nicol, D.: Trust mechanisms for cloud computing. Journal of Cloud Computing **2**(1), 1–14 (2013)
40. Iosup, A., Ostermann, S., Yigitbasi, M., Prodan, R., Fahringer, T., Epema, D.: Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing. IEEE Transactions on Parallel and Distributed Systems **22**(6), 931–945 (2011)
41. Jayasinghe, D., Malkowski, S., Li, J., Wang, Q., Wang, Z., Pu, C.: Variations in Performance and Scalability: An Experimental Study in IaaS Clouds Using Multi-Tier Workloads. IEEE Transactions on Services Computing **7**(2), 293–306 (2014)
42. Jennings, B. and Stadler, R.: Resource Management in Clouds: Survey and Research Challenges. Journal of Network and Systems Management **23**(3), 567–619 (2015)
43. Jiang, H.J., Huang, K.C., Chang, H.Y., Gu, D.S., Shih, P.J.: Scheduling Concurrent Workflows in HPC Cloud through Exploiting Schedule Gaps. In: Y. Xiang, A. Cuzzocrea, M. Hobbs, W. Zhou (eds.) Algorithms and Architectures for Parallel Processing, *Lecture Notes in Computer Science*, vol. 7016, pp. 282–293. Springer (2011)
44. Jiang, T., Zhang, Q., Hou, R., Chai, L., Mckee, S., Jia, Z., Sun, N.: Understanding the behavior of in-memory computing workloads. In: Proc. of the Int. Symp. on Workload Characterization - IISWC'14, pp. 22–30. IEEE (2014)
45. Johnson, S., Huizenga, G., Pulavarty, B.: Performance Tuning for Linux Servers. IBM RedBooks (2005)
46. Juan, D.C., Li, L., Peng, H.K., Marculescu, D., Faloutsos, C.: Beyond Poisson: Modeling Inter-Arrival Time of Requests in a Datacenter. In: V. Tseng, T. Ho, Z.H. Zhou, A. Chen, H.Y. Kao (eds.) Advances in Knowledge Discovery and Data Mining, *Lecture Notes in Computer Science*, vol. 8444, pp. 198–209. Springer (2014)
47. Juve, G., Chervenak, A., Deelman, E., Bharathi, S., Mehta, G., Vahi, K.: Characterizing and profiling scientific workflows. Future Generation Computer Systems **29**(3), 682–692 (2013)
48. Kaviani, N., Wohlstadter, E., Lea, R.: Profiling-as-a-Service: Adaptive Scalable Resource Profiling for the Cloud in the Cloud. In: G. Kappel, Z. Maamar, H. Motahari-Nezhad (eds.) Service-Oriented Computing, *Lecture Notes in Computer Science*, vol. 7084, pp. 157–171. Springer (2011)
49. Kessaci, Y., Melab, N., Talbi, E.G.: A Pareto-based metaheuristic for scheduling HPC applications on a geographically distributed cloud federation. Cluster Computing **16**(3), 451–468 (2012)
50. Khan, A., Yan, X., Tao, S., Anerousis, N.: Workload Characterization and Prediction in the Cloud: A Multiple Time Series Approach. In: Proc. of the 13th Network Operations and Management Symposium - NOMS'12, pp. 1287–1294. IEEE (2012)
51. Khazaei, H., Misic, J., Misic, V.: Performance Analysis of Cloud Computing Centers Using $M/G/m/m+r$ Queuing Systems. IEEE Transactions on Parallel and Distributed Systems **23**(5), 936–943 (2012)
52. Li, S., Ren, S., Yu, Y., Wang, X., Wang, L., Quan, G.: Profit and Penalty Aware Scheduling for Real-Time Online Services. IEEE Transactions on Industrial Informatics **8**(1), 78–89 (2012)
53. Lim, H., Babu, S., Chase, J., Parekh, S.: Automated Control in Cloud Computing: Challenges and Opportunities. In: Proc. of the 1st Workshop on Automated Control for Datacenters and Clouds - ACDC'09, pp. 13–18. ACM (2009)
54. Liu, S., Quan, G., Ren, S.: On-Line Scheduling of Real-Time Services for Cloud Computing. In: Proc. of the 6th World Congress on Services, pp. 459–464. IEEE (2010)

55. Liu, Z., Cho, S.: Characterizing Machines and Workloads on a Google Cluster. In: Proc. of the 41st Int. Conf. on Parallel Processing Workshops - ICPPW'12, pp. 397–403. IEEE (2012)
56. Mao, M., Humphrey, M.: Scaling and Scheduling to Maximize Application Performance within Budget Constraints in Cloud Workflows. In: Proc. of the 27th Int. Symp. on Parallel and Distributed Processing - IPDPS'13, pp. 67–78. IEEE (2013)
57. Marshall, P., Keahey, K., Freeman, T.: Elastic Site: Using Clouds to Elastically Extend Site Resources. In: Proc. of the 10th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing - CCGrid'10, pp. 43–52. IEEE (2010)
58. Mauch, V., Kunze, M., Hillenbrand, M.: High performance cloud computing. Future Generation Computer Systems **29**(6), 1408–1416 (2013)
59. Meng, S., Liu, L.: Enhanced Monitoring-as-a-Service for Effective Cloud Management. IEEE Transactions on Computers **62**(9), 1705–1720 (2013)
60. Mishra, A., Hellerstein, J., Cirne, W., Das, C.: Towards Characterizing Cloud Backend Workloads: Insights from Google Compute Clusters. ACM SIGMETRICS Performance Evaluation Review **37**(4), 34–41 (2010)
61. Moreno-Vozmediano, R., Montero, R., Llorente, I.: Multicloud Deployment of Computing Clusters for Loosely Coupled MTC Applications. IEEE Transactions on Parallel and Distributed Systems **22**(6), 924–930 (2011)
62. Mueller, J., Palma, D., Landi, G., Soares, J., Parreira, B., Metsch, T., Gray, P., Georgiev, A., Al-Hazmi, Y., Magedanz, T., Simoes, P.: Monitoring as a Service for Cloud Environments. In: Proc. of the 5th Int. Conf. on Communications and Electronics - ICCE'14, pp. 174–179. IEEE (2014)
63. de Oliveira, G., Ribeiro, E., Ferreira, D., Araujo, A., Holanda, M., Walter, M.: ACOsched: A scheduling algorithm in a federated cloud infrastructure for bioinformatics applications. In: Proc. of the Int. Conf. on Bioinformatics and Biomedicine - BIBM'13, pp. 8–14. IEEE (2013)
64. Pacheco-Sanchez, S., Casale, G., Scotney, B., McClean, S., Parr, G., Dawson, S.: Markovian Workload Characterization for QoS Prediction in the Cloud. In: Proc. of the 4th Int. Conf. on Cloud Computing - CLOUD'11, pp. 147–154. IEEE (2011)
65. Pandey, S., Wu, L., Guru, S., Buyya, R.: A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments. In: Proc. of the 24th Int. Conf. on Advanced Information Networking and Applications - AINA'10, pp. 400–407. IEEE (2010)
66. Patel, J., Jindal, V., Yen, I., Bastani, F., Xu, J., Garraghan, P.: Workload Estimation for Improving Resource Management Decisions in the Cloud. In: Proc. of the 12th Int. Symp. on Autonomous Decentralized Systems - ISADS'15, pp. 25–32. IEEE (2015)
67. Raicu, I.: Many-task Computing: Bridging the Gap Between High-throughput Computing and High-performance Computing. Ph.D. thesis, University of Chicago (2009)
68. Reiss, C., Tumanov, A., Ganger, G., Katz, R., Kozuch, M.: Heterogeneity and Dynamicity of Clouds at Scale: Google Trace Analysis. In: Proc. of the 3rd Symp. on Cloud Computing - SoCC'12, pp. 7:1–7:13. ACM (2012)
69. Reiss, C., Wilkes, J., Hellerstein, J.L.: Google cluster-usage traces: format+ schema. Google Inc. (2011)
70. Ren, G., Tune, E., Moseley, T., Shi, Y., Rus, S., Hundt, R.: Google-wide Profiling: A Continuous Profiling Infrastructure for Data Centers. IEEE Micro **30**(4), 65–79 (2010)
71. Ren, Z., Xu, X., Wan, J., Shi, W., Zhou, M.: Workload characterization on a production Hadoop cluster: A case study on Taobao. In: Proc. of the Int. Symp. on Workload Characterization - IISWC'12, pp. 3–13. IEEE (2012)
72. Sadooghi, I., Palur, S., Anthony, A., Kapur, I., Belagodu, K., Purandare, P., Ramamurty, K., Wang, K., Raicu, I.: Achieving Efficient Distributed Scheduling with Message Queues in the Cloud for Many-Task Computing and High-Performance Computing. In: Proc. of the 14th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing - CCGrid'14, pp. 404–413. IEEE (2014)

73. Samak, T., Gunter, D., Goode, M., Deelman, E., Juve, G., Silva, F., Vahi, K.: Failure Analysis of Distributed Scientific Workflows Executing in the Cloud. In: Proc. of the 8th Int. Conf. on Network and System Management - CNSM'12, pp. 46–54. IEEE (2012)
74. Schad, J., Dittrich, J., Quiané-Ruiz, J.A.: Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance. Proceedings of the VLDB Endowment **3**(1-2), 460–471 (2010)
75. Sharma, U., Shenoy, P., Towsley, D.: Provisioning Multi-tier Cloud Applications Using Statistical Bounds on Sojourn Time. In: Proc. of the 9th Int. Conf. on Autonomic Computing - ICAC'12, pp. 43–52. ACM (2012)
76. Shen, S., van Beek, V., Iosup, A.: Statistical Characterization of Business-Critical Workloads Hosted in Cloud Datacenters. In: Proc. of the 15th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing - CCGrid'15. IEEE (2015)
77. Singh, R., Sharma, U., Cecchet, E., Shenoy, P.: Autonomic Mix-aware Provisioning for Non-stationary Data Center Workloads. In: Proc. of the 7th Int. Conf. on Autonomic Computing - ICAC'10, pp. 21–30. ACM (2010)
78. Solis Moreno, I., Garraghan, P., Townend, P., Xu, J.: Analysis, Modeling and Simulation of Workload Patterns in a Large-Scale Utility Cloud. IEEE Transactions on Cloud Computing **2**(2), 208–221 (2014)
79. Somasundaram, T., Govindarajan, K.: CLOUDRB: A framework for scheduling and managing High-Performance Computing (HPC) applications in science cloud. Future Generation Computer Systems **34**, 47–65 (2014)
80. Spicuglia, S., Björkqvist, M., Chen, L., Serazzi, G., Binder, W., Smirni, E.: On Load Balancing: A Mix-aware Algorithm for Heterogeneous Systems. In: Proc. of the 4th Int. Conf. on Performance Engineering - ICPE'13, pp. 71–76. ACM (2013)
81. Tai, J., Zhang, J., Li, J., Meleis, W., Mi, N.: ArA: Adaptive Resource Allocation for Cloud Computing Environments under Bursty Workloads. In: Proc. of the 30th Int. Conf. on Performance Computing and Communications - IPCCC'11, pp. 1–8. IEEE (2011)
82. Tickoo, O., Iyer, R., Illikkal, R., Newell, D.: Modeling Virtual Machine Performance: Challenges and Approaches. SIGMETRICS Performance Evaluation Review **37**(3), 55–60 (2010)
83. Van den Bossche, R., Vanmechelen, K., Broeckhove, J.: Cost-Optimal Scheduling in Hybrid IaaS Clouds for Deadline Constrained Workloads. In: Proc. of the 3rd Int. Conf. on Cloud Computing - CLOUD'10, pp. 228–235. IEEE (2010)
84. Warneke, D., Kao, O.: Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud. IEEE Transactions on Parallel and Distributed Systems **22**(6), 985–997 (2011)
85. Weingärtner, R., Bräscher, G., Westphall, C.: Cloud resource management: A survey on forecasting and profiling models. Journal of Network and Computer Applications **47**, 99–106 (2015)
86. Wu, F., Wu, Q., Tan, Y.: Workflow scheduling in cloud: a survey. The Journal of Supercomputing **71**(9), 3373–3418 (2015)
87. Yang, H., Luan, Z., Li, W., Qian, D.: MapReduce Workload Modeling with Statistical Approach. Journal of Grid Computing **10**(2), 279–310 (2012)
88. Yin, J., Lu, X., Chen, H., Zhao, X., Xiong, N.N.: System resource utilization analysis and prediction for cloud based applications under bursty workloads. Information Sciences **279**, 338–357 (2014)
89. Zhan, Z.H., Liu, X.F., Gong, Y.J., Zhang, J., Chung, H.S.H., Li, Y.: Cloud Computing Resource Scheduling and a Survey of Its Evolutionary Approaches. ACM Computing Surveys **47**(4), 63:1–63:33 (2015)
90. Zhang, F., Cao, J., Li, K., Khan, S., Hwang, K.: Multi-objective scheduling of tasks in cloud platforms. Future Generation Computer Systems **37**, 309–320 (2014)

91. Zhang, Q., Zhani, M., Boutaba, R., Hellerstein, J.: Dynamic Heterogeneity-Aware Resource Provisioning in the Cloud. IEEE Transactions on Cloud Computing **2**(1), 14–28 (2014)
92. Zhao, Y., Fei, X., Raicu, I., Lu, S.: Opportunities and Challenges in Running Scientific Workflows on the Cloud. In: Proc. of the Int. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discovery - CyberC'11, pp. 455–462. IEEE (2011)
93. Zhu, Q., Agrawal, G.: Resource Provisioning with Budget Constraints for Adaptive Applications in Cloud Environments. In: Proc. of the 19th Int. Symp. on High Performance Distributed Computing - HPDC'10, pp. 304–307. ACM (2010)