IaaS Cloud Benchmarking: Approaches, Challenges, and Experience

Alexandru Iosup, Radu Prodan, and Dick Epema

Abstract Infrastructure-as-a-Service (IaaS) cloud computing is an emerging commercial infrastructure paradigm under which clients (users) can lease resources when and for how long needed, under a cost model that reflects the actual usage of resources by the client. For IaaS clouds to become mainstream technology and for current cost models to become more clientfriendly, benchmarking and comparing the non-functional system properties of various IaaS clouds is important, especially for the cloud users. In this article we focus on the IaaS cloud-specific elements of benchmarking, from a user's perspective. We propose a generic approach for IaaS cloud benchmarking, discuss numerous challenges in developing this approach, and summarize our experience towards benchmarking IaaS clouds. We argue for an experimental approach that requires, among others, new techniques for experiment compression, new benchmarking methods that go beyond blackbox and isolated-user testing, new benchmark designs that are domain-specific, and new metrics for elasticity and variability.

1 Introduction

Infrastructure-as-a-Service (IaaS) clouds are becoming a rich and active branch of commercial ICT services. Users of IaaS clouds can provision "processing, storage, networks, and other fundamental resources" [51] on-demand, that is, when needed, for as long as needed, and paying only for what is actually consumed. For the past five years, commercial IaaS clouds such as Amazon's EC2 have gained an increasing user base, from small and medium businesses [3] to scientific HPC users [14, 43]. However, the increased adoption of clouds and perhaps even the pricing models depend on the ability of (prospective) cloud users to benchmark and compare

A. Iosup (✉) • D. Epema
Delft University of Technology, EEMCS-Room HB07.050,
Mekelweg 4, 2628 CD Delft, The Netherlands
e-mail: A.Iosup@tudelft.nl; D.H.J.Epema@tudelft.nl

R. Prodan
Parallel and Distributed Systems, University of Innsbruck, Innsbruck, Austria
e-mail: Radu@dps.uibk.ac.at

© Springer Science+Business Media New York 2014

X. Li, J. Qiu (eds.), *Cloud Computing for Data-Intensive Applications*,
DOI 10.1007/978-1-4939-1905-5_4

commercial cloud services. In this chapter, we investigate the IaaS cloud-specific elements of benchmarking from the user perspective.

An important characteristic of IaaS clouds is good performance, which needs to be ensured on-demand and sustained when needed over a long period of time. However, as we have witnessed happening with several other new technologies while still in their infancy, notably with grid computing in the 1990s, it is likely that IaaS clouds will also undergo a period of changing performance management practices. In particular, we foresee that the branch of performance management that focuses on measuring the performance will evolve from traditional practices to meet the requirements of cloud operators and customers.

Benchmarking is a traditional approach to verify that the performance of a system meets the requirements. When benchmarking results are published, for example through mixed consumer-provider organizations such as SPEC and TPC, the consumers can easily compare products and put pressure on the providers to use best-practices and perhaps lower costs. Currently, the use of clouds is fragmented across many different application areas, including hosting applications, media, games, and web sites, E-commerce, On-Demand Workforce and CRM, high-performance computing, search, and raw resources for various usage. Each application area has its own (de facto) performance standards that have to be met by commercial clouds, and some have even developed benchmarks (e.g., BioBench [1] for Bioinformatics and RUBiS [63] for online business).

For IaaS clouds, we conjecture that the probable characteristics of current and near-future workloads can be derived from three major trends emerging from the last decade of grid and large-scale computing. First, individual jobs are now predominantly split into smaller compute or data-intensive tasks (many tasks [58]); there are almost no tightly coupled parallel jobs. Second, the duration of individual tasks is diminishing with every year; few tasks are still running for longer than 1 h and a majority require only a few minutes to complete. Third, compute-intensive jobs are split either into bags-of-tasks (BoTs) or DAG-based workflows, but data-intensive jobs may use a variety of programming models, from MapReduce to general dataflow.

Cloud benchmarking is not a straightforward application of older benchmarking techniques. In the past, there have been several large-scale computing environments that have similarities with clouds. Already decades ago, institutes such as CERN and the IBM T.J. Watson Research Center had large numbers of mainframes (using virtualization through the Virtual Machine operating system!) that also used multi-tenancy across their departments. Similarly, some vendors had large-scale installations for paid use by customers through Remote Job Entry facilities. In these environments, benchmarking and capacity planning were performed in close collaboration between owners and customers. An important difference, and advantage, for customers wishing to benchmark their prospective computing environments is that they can simply use access by credit card to deploy and benchmark their applications in the cloud: clouds do not only offer elasticity on demand, they also offer (resources for) capacity planning and benchmarking on demand. The new challenge is that customers will have to gain, through benchmarking, sufficient trust

in the performance, the elasticity, the stability, and the resilience of clouds, to rely on them for the operation of their businesses. As a matter of fact, cloud customers may want to benchmark both when migrating to the cloud, and, after migration, to assess continuously the operation of their applications in the cloud. Thus, of great importance is the ability of cloud benchmarks to allow users to gain trust without requiring long setups and costly operation.

We discuss in this chapter a focused, community-based approach to IaaS cloud benchmarking in which the main challenges are jointly identified, and best-practice and experiences can be easily shared. Although we have seen in the past few years numerous approaches to benchmarking and performance evaluation of various systems, there is no unified view of the main challenges facing researchers and practitioners in the field of benchmarking. This chapter aims at providing this unified view and should thus be useful in system procurement and performance management. From traditional benchmarking, the unified view borrows from earlier efforts on benchmarking middleware [8,9], on benchmarking databases [24], on the performance evaluation of grid and parallel-system schedulers [10, 15, 20, 35], and on benchmarking systems in general [2, 44].

The unified view includes a generic architecture for IaaS cloud benchmarking. We have designed the architecture so that it can be familiar to existing practitioners, yet provide new, cloud-specific functionality. For example, current IaaS cloud operators lease to their customers resources, but leave the selection of resource types and the selection of the lease/release moments as a customer task; because such selection can impact significantly the performance of the system built to use the leased resources, the generic benchmarking architecture must include policies for provisioning and allocation of resources.

In additional to traditional benchmarking elements and the generic architecture, the unified view introduced in this chapter focuses on ten important methodological, system-, workload-, and metrics-related issues. For example, how should cloud-bursting systems, that is, systems that lease resources to complement the customer's own resources, be benchmarked? What could be realistic models for the workloads of IaaS clouds? For IaaS clouds that share resources between multiple customers, how to benchmark their ability to isolate user-environments and thus to prevent performance variability [39]? etc.

This chapter has evolved from a number of regular articles [19, 26, 27] and a series invited talks given by the authors between 2012 and 2014, including talks at MTAGS 2012 [41], HotTopiCS 2013 [32], etc.[1] This work has also benefited from valuable discussion in the SPEC Research Group's Cloud Working Group (see also Sect. 5.1).

[1]In inverse chronological order: Lecture at the Linked Data Benchmark Council's Fourth TUC Meeting 2014, Amsterdam, May 2014. Lecture at Intel, Haifa, Israel, June 2013. Lecture at IBM Research Labs, Haifa, Israel, May 2013. Lecture at IBM T.J. Watson, Yorktown Heights, NY, USA, May 2013. Lecture at Technion, Haifa, Israel, May 2013.

The remainder of this chapter is structured as follows. In Sect. 2, we present a primer on benchmarking computer systems. Then, we introduce a generic approach for IaaS cloud benchmarking, in Sect. 3. In Sect. 4, we discuss numerous challenges in developing our and other approaches for cloud benchmarking, with focus on methodological, system-, workload-, and metrics-related issues. We summarize our experience towards benchmarking IaaS clouds in Sect. 5. Our summary focuses on the initiatives of the SPEC Research Group and its Cloud Working Group, of which the authors are members, and our own experience with building models and tools that can become useful building blocks for IaaS cloud benchmarking. Last, we conclude in "Conclusion" section.

2 A Primer on Benchmarking Computer Systems

We review in this section the main reasons for benchmarking and the main elements of the typical benchmarking process, which are basically unchanged since the early 1990s. For more detail, we refer to canonical texts on benchmarking [24] and performance evaluation [44] of computer systems.

2.1 Why Benchmarking?

Benchmarking computer systems is the process of evaluating their performance and other non-functional characteristics with the purpose of comparing them with other systems or with industry-agreed standards. Traditionally, the main use of benchmarking has been to facilitate the informed procurement of computer systems through the publication of verifiable results by system vendors and third-parties. However, benchmarking has grown as a support process for several other situations, which we review in the following.

Use in System Design, Tuning, and Operation Benchmarking has been shown to increase pressure on vendors to design better systems, as has been for example the experience of the TPC-D benchmark [24, Ch. 3, Sec. IV]. For this benchmark, insisting on the use of SQL has driven the wide acceptance of the ANSI SQL-92; furthermore, the complexity of a majority of the queries has lead to numerous improvements in the design of aggregate functions and support for them. This benchmark also led to a wide adoption of the geometric mean for aggregating normalized results [2]. The tuning of the DAS multi-cluster system has benefited from the benchmarking activity of some of the authors of this chapter, developed in the mid-2000s [33]; then, our distributed computing benchmarks exposed various (fixable) problems of the in-operation system.

Use in Training One of the important impediments in the adoption of a new technology is the lack of expertise of potential users. Market shortages of qualified

personnel in computer science are a major cause of concern for the European Union and the US. Benchmarks, through their open-source nature and representation of industry-accepted standards, can represent best-practices and thus be valuable training material.

On Alternatives to Benchmarking Several alternative methods have been used for the purposes described earlier in this section, among them empirical performance evaluation, simulation, and even mathematical analysis. We view benchmarking as an empirical evaluation of performance that follows a set of accepted procedures and best-practices. Thus, the use of empirical performance evaluation is valuable, but perhaps without the representativeness of a (de facto) standard benchmark. We see a role for (statistical) simulation [17, 22, 55] and mathematical analysis when the behavior of the system is well-understood and for long-running evaluations that would be impractical otherwise. However, simulating new technology, such as cloud computing, requires careful (and time-consuming) validation of assumptions and models.

2.2 Elements of Benchmarking

Inspired by canonical texts [24, 44], we review here the main elements of a benchmarking process. The main requirements of a benchmark—relevance, portability, scalability, and simplicity—have been discussed extensively in related literature, for example in [24, Ch. 1].

The *System Under Test (SUT)* is the system that is being evaluated. A *white box* system exposes its full operation, whereas a *black box* system does not expose operational details and is evaluated only through its outputs.

The *workload* is the operational load to which the SUT is subjected. Starting from the empirical observation that "20 % of the code consumes 80 % of the resources", simple *microbenchmarks* (*kernel benchmarks* [24, Ch. 9]) are simplified or reduced-size codes designed to stress potential system bottlenecks. Using the methodology of Saavedra et al. [59] and later refinements such as Sharkawi et al. [61], the results of microbenchmarks can be combined with application profiles to provide credible performance predictions for any platform. *Synthetic* and even *real-world (complex) applications* are also used for benchmarking purposes, as a response to system improvements that make microbenchmarks run fast but do not affect the performance of much larger codes. For distributed and large-scale systems such as IaaS clouds, *simple workloads* comprised of a single application and a (realistic) job arrival process represent better the typical system load and have been used for benchmarking [33]. *Complex workloads*, that is, the combined simple workloads of multiple users, possibly with different applications and job characteristics, have started to be used in the evaluation of distributed systems [33, 65]; we see an important role for them in benchmarking.

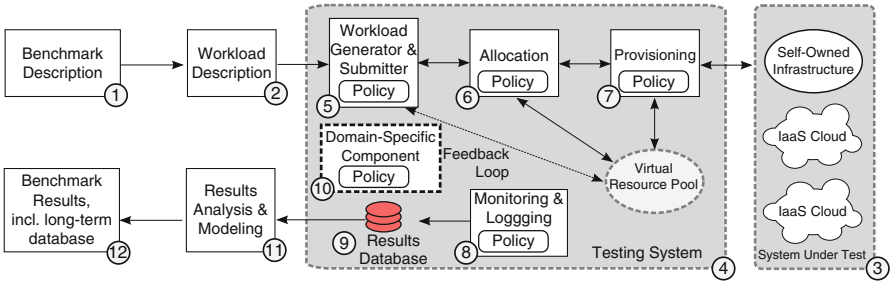

Fig. 1 Overview of our generic architecture for IaaS cloud benchmarking

The *Benchmarking Process* consists of the set of rules, prior knowledge (invariants), and procedures used to subject the SUT to the benchmark workload, and to collect and report the results.

3 A Generic Architecture for IaaS Cloud Benchmarking

We propose in this section a generic architecture for IaaS cloud benchmarking. Our architecture focuses on conducting benchmarks as sets of (real-world) experiments that lead to results with high statistical confidence, on considering and evaluating IaaS clouds as evolving black-box systems, on employing complex workloads that represent multi-tenancy scenarios, on domain-specific scenarios, and on a combination of traditional and cloud-specific metrics.

We introduce in Sect. 4 the main challenges that need to be addressed for our architecture to be realizable. In Sect. 5.2, we discuss a partial implementation of this architecture that has already achieved good results in practice [65].

3.1 Overview

Our main design principle is to adapt the proven designs for benchmarking to IaaS clouds at scale. Thus, we design an architecture that builds on our GrenchMark framework for grid benchmarking [33], as presented in Fig. 1.

The *Benchmarking Process* consists of the set of rules, prior knowledge (invariants), and procedures used to subject the SUT to the benchmark workload, and to collect and report the results. In our architecture, the process begins with the user (e.g., a prospective IaaS cloud user) defining the benchmark configuration, that is, the complex workloads that define the user's preferred scenario (component 1 in Fig. 1). The scenario may focus on processing as much of the workload as possible during a fixed test period or on processing a fixed-size workload as quickly or

cheaply as possible. The benchmarking system converts (component 2) the scenario into a set of workload descriptions, one per (repeated) execution. The workload may be defined before the benchmarking process, or change (in particular, increase) during the benchmarking process. To increase the statistical confidence in obtained results, subjecting the SUT to a workload may be *repeated* or the workload may be *long-running*.

After the preparation of the workload, the SUT (component 3 in Fig. 1) is subjected to the workload through the job and resource management services provided by the testing system (component 4, which includes components 5–10). In our benchmarking architecture, the SUT can be comprised of one or several self-owned infrastructures, and public and private IaaS clouds. The SUT provides resources for the execution of the workload; these resources are grouped into a *Virtual Resource Pool*. The results produced during the operation of the system may be used to provide a *feedback loop* from the Virtual Resource Pool back into the Workload Generator and Submitter (component 5); thus, our architecture can implement open and closed feedback loops [60].

As a last important sequence of process steps, per-experiment results are combined into higher-level aggregates, first aggregates per workload execution (component 11 in Fig. 1), then aggregates per benchmark (component 12). The reporting of metrics should try to avoid the common pitfalls of performance evaluation; see for example [2, 23]. For large-scale distributed systems, it is particularly important to report not only the basic statistics, but also some of the outliers, and full distributions or at least the higher percentiles of the distribution (95-th, 99-th, etc.). We also envision the creation of a general database of results collected by the entire community and shared freely. The organization and operation of such a database is within the scope of future work.

3.2 Distinguishing Design Features

We present in the remainder of this section several of the distinguishing features of this architecture.

In comparison with traditional grid environments, commercial IaaS clouds do not provide services for managing the incoming stream of requests (components 5, 6, and 8 in Fig. 1) or the resources leased from the cloud (components 7 and 8). Our architecture supports various policies for provisioning and allocation of resources (components 6 and 7, respectively). In contrast to GrenchMark, our generic cloud-benchmarking architecture also includes support for evolving black-box systems (components 9, 11, and 12), complex workloads and multi-tenancy scenarios (components 1, 2, and 5), domain-specific components (component 10), etc.

Experiments conducted on large-scale infrastructure should be designed to minimize the time spent effectively using resources. The interplay between components 1, 2, and 5 in Fig. 1 can play a non-trivial role in resolving this challenge, through automatic selection and refinement of complex test workloads that balance the

trade-off between accuracy of results and benchmark cost; the main element in a dynamic tuning of this trade-off is the policy present in component 5. The same interplay enables multi-tenancy benchmarks.

Several of the possible SUTs expose complete or partial operational information, acting as white or partially white boxes. Our architecture allows exploiting this information, combining results from black-box and white-box testing. Moreover, the presence of the increasingly higher-level aggregations (components 11 and 12 in Fig. 1) permits both the long-term evaluation of the system, and the combination of short-term and long-term results. The policy for monitoring and logging in component 8 allows the user to customize what information is processed and stored in the results database. We conclude that our architecture goes far beyond simple black-box testing.

Supports domain-specific benchmarks is twofold in our architecture. First, components 5–7 support complex workloads and feedback loops, and policy-based resource and job management. Second, we include in our architecture a domain-specific component (component 10) that can be useful in supporting cloud programming models such as the compute-intensive workflows and bags-of-tasks, and the data-intensive MapReduce and Pregel. The policy element in component 10 allows this component to play a dynamic, intelligent role in the benchmarking process.

4 Open Challenges in IaaS Cloud Benchmarking

We introduce in this section an open list of surmountable challenges in IaaS cloud benchmarking.

4.1 Methodological

Challenge 1. Experiment compression.

Long setup times, for example of over a day, and/or long periods of continuous evaluation, for example of more than a day per result, reduce the usefulness of a benchmark for the general user. This is a general problem with any experimental approach, but for IaaS clouds it has the added disadvantage of greatly and visibly increasing the cost of benchmarking. We argue that research is needed to reduce the setup and operational time of benchmarks for IaaS clouds. This can be achieved through reduced input and application sets, a clever setup of the experiments, and sharing of results across the community. We also envision the use of combined

experimental approaches, in which real-world experiments are combined with emulation [64,66] or simulation. Our vision for experiment compression represents an extension of the concept of statistical simulation [17,22,55], which has been used for computer architecture studies, to real-world experimentation.

Reduced benchmark input and application sets can be obtained by refining input workloads from real complex workloads, using theoretically sound methods (e.g., statistical models and goodness-of-fit tests). Such reduced benchmark inputs will contrast with traditional synthetic benchmarks, which incorporate many human-friendly parameter values (e.g., "10 % queries of type A, 90 % queries of type B") and thus may lack theoretical guarantees for representativeness.

Challenge 2. Beyond black-box testing through testing short-term dynamics and long-term evolution.

Similarly to multi-cluster grids, which frequently added clusters or individual nodes to the distributed infrastructure, clouds are continuously extended and tuned by their operators. Moreover, commercial clouds such as Amazon EC2 add frequently new functionality to their systems. Thus, the benchmarking results obtained at any given time may be unrepresentative for the future behavior of the system. We argue that IaaS clouds should not be benchmarked only using traditional black-box and even white-box testing, for which the system under test does not change in size and functionality, but also through new benchmarking methods that evaluate the impact of short-term dynamics and long-term evolution. Specifically, short-term dynamics characterize system changes occurring over short periods (at most hours), and long-term evolution characterizes system changes occurring over long periods (months, years).

A straightforward approach to benchmark both short-term dynamics and long-term evolution is to measure the system under test periodically, with judiciously chosen frequencies [40]. However, this approach increases the pressure of the so-far unresolved Challenge 1.

Challenge 3. Impact of middleware.

IaaS clouds are built on several layers of middleware, from the guest operating system of the VM to the data-center resource manager. Each of these layers adds new complexity to testing and possibly also visible or invisible performance bottlenecks. One of the key issues in benchmarking IaaS clouds is to measure the performance of each layer of the middleware in isolation. We argue that a solution for this problem may not be possible under the current assumption of black-box testing, and propose instead to focus on a new methodology that accounts for imprecision in the isolation of root causes of performance.

We believe that good steps towards understanding the performance of various middleware layers can be and have already been taken [8], for example in assessing the impact of virtualization, but that more work is needed to reconcile the results (the situation presented in Challenge 2, where IaaS clouds change over time, may be a source of conflicting experimental results). We have surveyed in our previous work [39, 40] over ten performance studies that use common benchmarks to assess the virtualization overhead on computation (5–15 %), I/O (10–30 %), and HPC kernels (results vary). We have shown in a recent study of four commercial IaaS clouds [39] that virtualized resources obtained from public clouds can have a much lower performance than the theoretical peak, possibly because of the performance of the middleware layer.

4.2 System Properties

Challenge 4. Reliability, availability, and related system properties.

One of the factors affecting the behavior of large-scale systems is the presence of failures, which are likely inevitable at scale. We have found endemic presence of failures in many popular large-scale systems, from grids [36] to DNS and other distributed services [47]. Benchmarking reliability and related systems properties is difficult, not in the least because of Challenge 2.

Challenge 5. Massive scale, multi-site benchmarking.

One of the main product features of IaaS clouds is the promise of seemingly infinite capacity. We argue that benchmarking this promise is difficult, very time-consuming, and very costly. We have seen in our previous work that testing tools can be built to test infrastructures of thousands of cores [33], but performance evaluation tools that work at much larger scale in heterogeneous IaaS clouds have yet to be proven in practice. An important challenge here may be the ability to generate massive-scale workloads.

We have already had experience with companies building *hybrid clouds* [51] out of their own infrastructure and resources leased from IaaS clouds (this process is also referred to as *cloud-bursting*, for example by NIST and Microsoft). Other cloud deployment models require the use of multiple sites, because the application functionality requires it [12], to improve load balancing or performance [62], to fulfill reliability targets, to avoid vendor lock-in [5], etc. We and others [53] expect multi-site cloud use to increase, as more companies, with or without existing

computational capacity, try out or even decide to use cloud services. We also expect multi-site cloud use to reuse mechanisms of traditional co-allocation, that is, simultaneous allocation of resources across several organizational components with (wide) geographical spread. We argue that benchmarking across multiple sites raises additional challenges, not in the least the combined availability for testing and scalability of the infrastructure, and the increased cost.

Challenge 6. Performance isolation.

The negative effects of the interaction between running jobs in a complex workload have been observed in distributed environments since at least the mid-1990s [6]. Following early work [30, 49], we argue that quantifying the level of isolation provided by an IaaS cloud is a new and important challenge.

Moreover, as IaaS clouds become more international, their ability to isolate performance may suffer most during periods of peak activity [30]. Thus, studying the time patterns of performance interference, and their impact on the targets of performance isolation, is worthwhile.

4.3 Workload

Challenge 7. Realistic yet tunable models of workloads and of system performance.

Statistical workload modeling is the general technique of producing synthetic models from workload traces collected from real-world systems that are statistically similar to the real-world traces, yet may be sufficiently easy to tune for a community of non-expert users. We argue that building such statistical models raises important challenges, from data collection to trace processing, from finding good models to testing the validity of the models. We also see as an open challenge the derivation of statistical performance models, perhaps through linear regression, from already existing measurements.

We envision that IaaS clouds will also be built for specific, even niche application domains, charging premium rates for the expertise required to run specific classes of applications. This is similar to the appearance of domain-specific grids, such as BioGrid, in the early 2000s; and of domain-specific database-related technology, such as transaction-processing and data warehousing solutions, in the early 1990s [24, Ch.1]. We argue that IaaS cloud benchmarking should begin with domain-specific benchmarks, before transiting to general benchmarks.

Besides regular user workloads, most commercial IaaS clouds offer value-adding features such as backup, upgrade, (live) migration, load-balancing, scheduling and message queues, publish/subscribe-based communication services, etc. These value-adding features generate additional, cloud-internal workloads.

Toward building domain-specific benchmarks, we argue for building statistical models of domain-specific or at least programming model-specific workloads. We have conducted in the past extensive research in grid workloads [34], with results in modeling BoTs [38], and in characterizing scientific and engineering workflows [34]. Several studies [11, 21, 46, 68, 69], including our own study of four large MapReduce clusters [13], have focused on characterizing workloads of MapReduce, which is one of the most popular programming models for data processing in the loud. Open challenges in this context are the formulation of realistic models for workflows, MapReduce, and other programming models for data processing. We also find that the many-task programming model [58] is worthwhile for investigation in this context. We also refer to a recent survey of challenges associated with large-scale log analysis [54].

Challenge 8. Benchmarking performance isolation under different multi-tenancy models.

Unlike traditional system benchmarking, where interference of different elements that affect performance—multiple users competing for resources, stressing multiple system resources at the same time—is generally avoided, the expected cloud workload is complex. We argue that for IaaS clouds interference should be expected and benchmarked. Specific focus for this challenge, as an extension of Challenge 8, is to benchmark under a specific multi-tenancy model, from the shared-nothing approach of multi-cluster grids, to shared-hardware and shared-virtualized machine approaches prevalent in today's commercial clouds [48, 52], and possibly others.

4.4 Metrics

Challenge 9. Beyond traditional performance.

Traditional performance metrics—such as utilization, throughput, and makespan—have been defined for statically-sized, homogeneous systems. We have raised in our previous work [35] the challenge of adapting these metrics for distributed on-demand systems, such as the contemporary multi-cluster grids and commercial

IaaS clouds. IaaS clouds raise new challenges in defining cloud-related metrics, such as elasticity [7, 29, 42]; they also require revisiting traditional metrics, including dependability-related [31].

We also argue for revisiting the analysis of results and their refinement into metrics. For example, due to their change over time and imperfect performance isolation, IaaS clouds may require revisiting the concept of variability, beyond the traditional mean (or median) and standard deviation. Our preliminary work [40] on the variability of performance in IaaS and other types of clouds indicates that variability can be high and may vary with time.

Traditionally, system warm-up is excluded from performance evaluation, leaving only the steady-state period of the system for study. However, especially for hybrid and other multi-site cloud architectures, we argue for the need to also measure the transitional period that occurs when a significant fraction of the system resources are in the process of being leased or released.

Challenge 10. The cost issue.

Although cost models were discussed in benchmarking and performance evaluation of both databases and grids, a variety of issues have not been addressed. Specifically, the sub-leasing cost model used in today's commercial IaaS clouds (e.g., Amazon's "spot" instances) provides a new focus. It is also unclear how to define costs for a hybrid cloud infrastructure, especially when the performance of the cloud—throughput, makespan, etc.—does not match the expectation [39, 67]. Last but not least, it is unclear how to define the source of budgets, for example either infrastructural or operational funds, a situation which affects a variety of economic metrics. Early approaches exist [14, 43].

5 Experience Towards IaaS Cloud Benchmarking

In this section, we present our experience in joining a community of experts working on benchmarking IaaS clouds and in conducting independent research on the topic.

5.1 Methodology: The SPEC Cloud Working Group

The SPEC Research Group[2] (RG) is a new group within the Standard Performance Evaluation Corporation (SPEC). Among other activities, the SPEC RG facilitates the interaction between academia and industry by co-organizing the Joint

[2] http://research.spec.org/.

ACM/SPEC International Conference on Performance Engineering (ICPE). The Cloud Working Group[3] (CWG) is a branch of the SPEC RG that aims to develop the methodological aspects of cloud benchmarking (**Challenges 1–3** in Sect. 4). In this section we summarize two initiatives of the SPEC RG and CWG.

Beyond Traditional Performance Traditional performance metrics such as utilization and normalized schedule length [50] have been defined for statically sized systems. Redefining these metrics for dynamic systems, especially in the context of black-box resources leased from clouds, is a topic of interest for the CWG (**Challenges 5 and 6**). Beyond performance, the CWG is also interested in other non-functional metrics, such as elasticity, utility, performance isolation, and dependability (**Challenges 4, 9, and 15**).

Reproducibility of Experiments (Orthogonal to Our Challenges) Being able to reproduce experimental results is critical for the validity and lifetime of obtained results. However, this goal is difficult to achieve when the system under test is complex, dynamic, or large-scale; IaaS clouds have all these characteristics. A recent initiative of the RG is to build a repository[4] that can be used to share experimental results, setups, and other meta-data. Moreover, the call for papers issued by ICPE 2013 includes a focus on reproducibility of experiments.

5.2 SkyMark: A Framework for IaaS Cloud Benchmarking

We have recently implemented a part of the architecture described in Sect. 3 as our SkyMark tool for IaaS cloud benchmarking [4]. SkyMark already implements two of the distinguishing features of our architecture (see Sect. 3.2). First, SkyMark provide services for managing the incoming stream of requests (jobs) and the resources leased from the cloud [65]. For the former, SkyMark provides single or multiple job queues, depending on the configuration of the experiment, and each queue supports a variety of simple scheduling policies (e.g., FCFS). For the latter, SkyMark supports several static and dynamic resource provisioning policies.

Second, SkyMark supports complex workloads (**Challenge 7**). Workloads are split into units. Each unit is defined by the characteristic resource to be stressed (e.g., through CPU-intensive jobs), the job arrival pattern (one of uniform, increasing, and bursty), and the job durations. SkyMark is able, for a given target configuration, to generate workloads that lead to a user-specified average utilization in the absence of system overheads.

Using SkyMark, we were able [65] to benchmark three IaaS clouds, including Amazon EC2. We have used in out benchmarks six provisioning policies and

[3] http://research.spec.org/working-groups/rg-cloud-working-group.html.

[4] ICPE Organizers, Reproducibility repository approved, http://icpe2013.ipd.kit.edu/news/single_view/article/reproducibility-repository-approved/.

three allocation policies, with provisioning and allocation policies considered either independently or together. We were also able [4] to evaluate, for our OpenNebula private clouds, the interference occurring in various multi-tenancy scenarios (**Challenge 8**).

5.3 Real-World Evaluation of IaaS Cloud Performance

Several of the challenges we formulated in Sect. 4 are the outcome of our previous research conducted from the past three years in benchmarking and understanding the performance of several cloud infrastructures. We summarize in the following some of our main results that motivated this classification.

Challenge 2 We have observed the long-term evolution in performance of clouds since 2007. Then, the acquisition of one EC2 cloud resource took an average time of 50 s, and constantly increased to 64 s in 2008 and 78 s in 2009. The EU S3 service shows pronounced daily patterns with lower transfer rates during night hours (7 PM to 2 AM), while the US S3 service exhibits a yearly pattern with lowest mean performance during the months January, September, and October. Other services have occasional decreases in performance, such as SDB in March 2009, which later steadily recovered until December [40]. Finally, EC2 spot prices typically follow a long-term step function [56].

Challenge 3 Depending on the provider and its middleware abstraction, several cloud overheads and performance metrics can have different interpretation and meaning. In IaaS clouds, resource acquisition is typically the sum of the installation time and boot times, and for Amazon EC2 has a stable value in the order of minutes [39]. Other IaaS providers, such as GoGrid, behave similarly to grids and offer highly variable resource acquisition times, i.e., one order magnitude higher than EC2. In contrast, the Google App Engine (GAE), which offers a higher-level PaaS abstraction, defines the acquisition overhead as the time between the issue of a HTTP request until the HTTP response is returned; the overhead of GAE is in the order of seconds [57], an order of magnitude lower than for EC2. The performance interpretations and differences can have similarly high variations depending on the middleware. The black-box execution approach in IaaS clouds of externally-compiled software encapsulated in VMs generates high degradations from the expected peak performance, up to six to eight times lower than the theoretical maximum of Amazon's "Elastic Compute Unit" (ECU, 4.4 GOPS) [39]. Parallel computing-wise, the performance of today's IaaS is below the theoretical peak of today's dedicated parallel supercomputers even for demanding conveniently parallel applications by 60–70 %. Furthermore, benchmarking the sustained performance of other infrastructures such as GAE is almost prohibited by the sandboxed environment that completely hides the underlying hardware on which the instance is started with no user control, raising the need for **Challenge 6** [57].

The IaaS middleware has a significant impact on the PaaS environments researched on top. An interesting example is Amazon Simple Workflow (SWF)[5] that enables programming and executing workflow applications on the EC2 cloud. Our previous analysis[45] indicates that SWF represents an attractive environment for running traditional workflow applications, especially those consisting of numerous relatively short activities affected by the large grid middleware overheads. In contrast, porting existing grid workflow middleware environments such as ASKALON to the cloud, although effective, exhibit performance losses due to their high middleware stacks required for portability in supporting a wider range of distributed and heterogeneous cluster, grid, and cloud computing infrastructures, as opposed to the SWF restricted, but highly optimized for the EC2 infrastructure.

Challenge 4 With respect to reliability, the payment models and compensations in case of resource failures make clouds a more promising platform than traditional distributed systems, especially grids. Interesting from the reliability point of view are the EC2 spot instances that allow customers to bid on unused capacity and run those instances for as long as their bid exceeds the current spot price. Our analysis on this risk-reward problem between January 2011 and February 2012 demonstrates that spot instances may represent a cheaper but still reliable solution offering up to 99 % availability provided that users make slightly generous bids, such as $0.35 for `m1.large` instances [56].

Challenge 5 Although multi-cloud environments promise seemingly infinite scalability and performance, our experience revealed that this is not always the case for communicating non-embarrassingly parallel applications. For example, our study on using Amazon EC2 and GoGrid as independent providers[16] illustrated that multi-clouds can help in shortening the makespan for workflow applications which do not require transferring large amounts of data among activities. In situations when data transfers dominate the computation time, the workflow does not benefit from a federation of Clouds and performs better in a single provider configuration. A deeper analysis of the results also reveals that cheap schedules targeting cost minimisation rarely consider federated resources and rather use resources from a single provider. An explanation for this behavior is the hourly based price model offered by the providers, cheap solutions trying to increase resource utilisation instead of launching simultaneous instances.

Challenge 9 Regarding the importance of system warmup, an interesting case is the modern just-in-time (JIT) compilations of Java application running on GAE infrastructure which can boost the performance of interpreted Java byte code by a factor of four in a predictable manner (from the third request onwards in case of GAE) [57].

Challenge 10 The variety of cost models combined with performance variability makes the cloud provider selection a difficult problem for the cloud user. For

[5] https://aws.amazon.com/de/swf/.

example, our analysis in [57] shows that computing costs are lower on GAE than in EC2 for very short jobs, mostly due to the cycle-based payment granularity, as opposed to the hourly billing intervals of EC2. The cost model may also vary within one provider. For example, the EC2 reserved instances are cheaper than standard instances if their usage is of about 50 % for for one year reservations, and of about 30 % for three year reservations [56]. In contrast, spot instances on EC2 may represent a 60 % cheaper but equally reliable alternative to standard instances provided that a correct user bet is made [56].

5.4 Statistical Workload Models

Challenge 7 In our previous work, starting from multi-cluster grid traces, we have proposed statistical models of BoTs [38], and characterized BoTs [34, 38] and workflows [34]. We found, notably, that BoTs are the dominant programming model for *compute-intensive* workloads in grids—they account for 80–90 % of both number of tasks and resource consumption. We have characterized and modeled statistically MapReduce workloads, starting from four traces of large clusters, including Google's [13].

A recent trend in *data-intensive* processing is the increasing automation of work, as workflows of inter-dependent tasks. We have modeled conceptually and characterized empirically [28] the workflow of a class of MapReduce applications, where time-stamped data collected from super-nodes in a global-scale deployment of a hundred-million-node distributed computing system are analyzed. This MapReduce use case has challenging features for MapReduce systems such as Hadoop and its successor YARN: small (kilobytes) to large (hundreds of megabytes) data sizes per observed item, very poor (100:1) to excellent (1:1 million) output:input ratio, and short (seconds) to long (hours) individual-job duration. Our findings indicate that traditional benchmarks for MapReduce that rely on single applications, such as PUMA, HiBench, ClueWeb09, and Grid/PigMix, are well complemented by workflow-based benchmarking.

5.5 Open Data: Several Useful Archives

Challenge 7 Workload and operational trace archives are an important tool in developing benchmarks. Although IaaS clouds are new, several online archives could already provide interesting data.

General workload traces for parallel systems and multi-cluster grid are provided by the Parallel Workloads Archive [18] and the Grid Workloads Archive [37], respectively. For an example of domain-specific workload traces, the Game Trace Archive [25] publishes data representative for online gaming.

For operational traces, the Failure Trace Archive [47] and the P2P Trace Archive [70] provide operational information about general and domain-specific (peer-to-peer) distributed systems.

Conclusion

The importance of IaaS cloud benchmarking has grown proportionally to the increased adoption of this technology, from small and medium businesses to scientific HPC users. In contrast to the fragmented field of today, we discuss in this work a more focused, unified approach to IaaS benchmarking, in which the community can join into identifying the main challenges, and then share best-practices and experiences. This approach could greatly benefit (prospective) cloud users with system procurement and performance management.

The unified view includes a generic architecture for IaaS cloud benchmarking, and focuses on ten important methodological, system-, workload-, and metrics-related issues. In our generic architecture, resource and job management can be provided by the testing infrastructure, there is support for black-box systems that change rapidly and can evolve over time, tests are conducted with complex workloads, and various multi-tenancy scenarios can be investigated.

We also discuss four classes of challenges in developing this approach: methodological, system property-related, workload-related, and metric-related. We identify ten main challenges to benchmarking IaaS clouds:

1. Experiment compression. (Methodological)
2. Beyond black-box testing through testing short-term dynamics and long-term evolution. (Methodological)
3. Impact of middleware. (Methodological)
4. Reliability, availability, and related system properties. (System)
5. Massive scale, multi-site benchmarking. Cloud-bursting. Co-allocation. (System)
6. Performance isolation. (System)
7. Realistic yet tunable models of workloads and of system performance. (Workload)
8. Benchmarking performance isolation under different multi-tenancy models. (Workload)
9. Beyond traditional performance. Elasticity and variability. (Metric)
10. The cost issue. Relate with metrics such as utilization, throughput, and makespan. (Metric)

Last, we summarize our experience towards benchmarking IaaS clouds. We have initiated various community-wide efforts via our work in the SPEC Research Group and its Cloud Working Group. We also present here a summary of our work in building models and tools for IaaS cloud benchmarking.

Acknowledgements This work was partially supported by the STW/NWO Veni grant @larGe (11881), EU projects PEDCA and EYE, Austrian Science Fund (FWF) project TRP 237-N23, and the ENIAC Joint Undertaking (project eRAMP).

References

1. Albayraktaroglu K, Jaleel A, Wu X, Franklin M, Jacob B, Tseng CW, Yeung D (2005) Biobench: A benchmark suite of bioinformatics applications. In: ISPASS, IEEE Computer Society, pp 2–9
2. Amaral JN (2012) How did this get published? Pitfalls in experimental evaluation of computing systems. LTES talk, [Online] Available: http://webdocs.cs.ualberta.ca/~amaral/Amaral-LCTES2012.pptx. Last accessed Oct 2012.
3. Amazon Web Services (2012) Case studies. Amazon web site, [Online] Available: http://aws.amazon.com/solutions/case-studies/. Last accessed Oct 2012.
4. Antoniou A, Iosup A (2012) Performance evaluation of cloud infrastructure using complex workloads. TU Delft MSc thesis, [Online] Available: http://repository.tudelft.nl/view/ir/uuid:d8eda846-7e93-4340-834a-de3e4aa93f8b/. Last accessed Oct 2012.
5. Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, Stoica I, Zaharia M (2010) A view of cloud computing. Commun ACM 53(4):50–58
6. Arpaci-Dusseau RH, Arpaci-Dusseau AC, Vahdat A, Liu LT, Anderson TE, Patterson DA (1995) The interaction of parallel and sequential workloads on a network of workstations. In: SIGMETRICS, pp 267–278
7. Brebner P (2012) Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (iaas) cloud applications. In: ICPE, pp 263–266
8. Brebner P, Cecchet E, Marguerite J, Tuma P, Ciuhandu O, Dufour B, Eeckhout L, Frénot S, Krishna AS, Murphy J, Verbrugge C (2005) Middleware benchmarking: approaches, results, experiences. Concurrency and Computation: Practice and Experience 17(15):1799–1805
9. Buble A, Bulej L, Tuma P (2003) Corba benchmarking: A course with hidden obstacles. In: IPDPS, p 279
10. Chapin SJ, Cirne W, Feitelson DG, Jones JP, Leutenegger ST, Schwiegelshohn U, Smith W, Talby D (1999) Benchmarks and standards for the evaluation of parallel job schedulers. In: JSSPP, pp 67–90
11. Chen Y, Ganapathi A, Griffith R, Katz RH (2011) The case for evaluating mapreduce performance using workload suites. In: MASCOTS, pp 390–399
12. Czajkowski K, Foster IT, Kesselman C (1999) Resource co-allocation in computational grids. In: HPDC
13. De Ruiter TA, Iosup A (2012) A workload model for MapReduce. TU Delft MSc thesis, [Online] Available: http://repository.tudelft.nl/view/ir/uuid:1647e1cb-84fd-46ca-b1e1-21aaf38ef30b/. Last accessed Oct 2012.
14. Deelman E, Singh G, Livny M, Berriman JB, Good J (2008) The cost of doing science on the cloud: the Montage example. In: SC, IEEE/ACM, p 50
15. Downey AB, Feitelson DG (1999) The elusive goal of workload characterization. SIGMETRICS Performance Evaluation Review 26(4):14–29
16. Durillo JJ, Prodan R (2014) Workflow scheduling on federated clouds. In: Euro-Par, Springer, LNCS
17. Eeckhout L, Nussbaum S, Smith JE, Bosschere KD (2003) Statistical simulation: Adding efficiency to the computer designer's toolbox. IEEE Micro 23(5):26–38
18. Feitelson D (2013) Parallel Workloads Archive. http://www.cs.huji.ac.il/labs/parallel/workload/
19. Folkerts E, Alexandrov A, Sachs K, Iosup A, Markl V, Tosun C (2012) Benchmarking in the cloud: What it should, can, and cannot be. In: TPCTC, pp 173–188

20. Frachtenberg E, Feitelson DG (2005) Pitfalls in parallel job scheduling evaluation. In: JSSPP, pp 257–282
21. Ganapathi A, Chen Y, Fox A, Katz RH, Patterson DA (2010) Statistics-driven workload modeling for the cloud. In: ICDE Workshops, pp 87–92
22. Genbrugge D, Eeckhout L (2009) Chip multiprocessor design space exploration through statistical simulation. IEEE Trans Computers 58(12):1668–1681
23. Georges A, Buytaert D, Eeckhout L (2007) Statistically rigorous java performance evaluation. In: OOPSLA, pp 57–76
24. Gray J (ed) (1993) The Benchmark Handbook for Database and Transaction Systems, 2nd edn. Mergan Kaufmann
25. Guo Y, Iosup A (2012) The Game Trace Archive. In: NETGAMES, pp 1–6
26. Guo Y, Biczak M, Varbanescu AL, Iosup A, Martella C, Willke TL (2014) How well do graph-processing platforms perform? an empirical performance evaluation and analysis. In: IPDPS
27. Guo Y, Varbanescu AL, Iosup A, Martella C, Willke TL (2014) Benchmarking graph-processing platforms: a vision. In: ICPE, pp 289–292
28. Hegeman T, Ghit B, Capota M, Hidders J, Epema DHJ, Iosup A (2013) The btworld use case for big data analytics: Description, mapreduce logical workflow, and empirical evaluation. In: BigData Conference, pp 622–630
29. Herbst NR, Kounev S, Reussner R (2013) Elasticity in Cloud Computing: What it is, and What it is Not. In: Proceedings of the 10th International Conference on Autonomic Computing (ICAC 2013), San Jose, CA, June 24–28, USENIX, preliminary Version
30. Huber N, von Quast M, Hauck M, Kounev S (2011) Evaluating and modeling virtualization performance overhead for cloud environments. In: CLOSER, pp 563–573
31. Huber N, Brosig F, Dingle N, Joshi K, Kounev S (2012) Providing Dependability and Performance in the Cloud: Case Studies. In: Wolter K, Avritzer A, Vieira M, van Moorsel A (eds) Resilience Assessment and Evaluation of Computing Systems, XVIII, Springer-Verlag, Berlin, Heidelberg, URL http://www.springer.com/computer/communication+networks/book/978-3-642-29031-2, iSBN: 978-3-642-29031-2
32. Iosup A (2013) Iaas cloud benchmarking: approaches, challenges, and experience. In: HotTopiCS, pp 1–2
33. Iosup A, Epema DHJ (2006) GrenchMark: A framework for analyzing, testing, and comparing grids. In: CCGrid, pp 313–320
34. Iosup A, Epema DHJ (2011) Grid computing workloads. IEEE Internet Computing 15(2):19–26
35. Iosup A, Epema DHJ, Franke C, Papaspyrou A, Schley L, Song B, Yahyapour R (2006) On grid performance evaluation using synthetic workloads. In: JSSPP, pp 232–255
36. Iosup A, Jan M, Sonmez OO, Epema DHJ (2007) On the dynamic resource availability in grids. In: GRID, IEEE, pp 26–33
37. Iosup A, Li H, Jan M, Anoep S, Dumitrescu C, Wolters L, Epema DHJ (2008) The grid workloads archive. Future Gener Comput Syst 24(7):672–686
38. Iosup A, Sonmez OO, Anoep S, Epema DHJ (2008) The performance of bags-of-tasks in large-scale distributed systems. In: HPDC, ACM, pp 97–108
39. Iosup A, Ostermann S, Yigitbasi N, Prodan R, Fahringer T, Epema DHJ (2011) Performance analysis of cloud computing services for many-tasks scientific computing. IEEE Trans Par Dist Syst 22(6):931–945
40. Iosup A, Yigitbasi N, Epema DHJ (2011) On the performance variability of production cloud services. In: CCGRID, pp 104–113
41. Iosup A, Prodan R, Epema DHJ (2012) Iaas cloud benchmarking: approaches, challenges, and experience. In: SC Companion/MTAGS
42. Islam S, Lee K, Fekete A, Liu A (2012) How a consumer can measure elasticity for cloud platforms. In: ICPE, pp 85–96
43. Jackson KR, Muriki K, Ramakrishnan L, Runge KJ, Thomas RC (2011) Performance and cost analysis of the supernova factory on the amazon aws cloud. Scientific Programming 19(2–3):107–119

44. Jain R (ed) (1991) The Art of Computer Systems Performance Analysis. John Wiley and Sons Inc.
45. Janetschek M, Prodan R, Ostermann S (2013) Bringing scientific workflows to amazon swf. In: 2013 39th Euromicro Conference Series on Software Engineering and Advanced Applications, IEEE, pp 389–396, DOI 10.1109/SEAA.2013.13
46. Kim K, Jeon K, Han H, Kim SG, Jung H, Yeom HY (2008) Mrbench: A benchmark for mapreduce framework. In: ICPADS, pp 11–18
47. Kondo D, Javadi B, Iosup A, Epema DHJ (2010) The failure trace archive: Enabling comparative analysis of failures in diverse distributed systems. In: CCGrid, pp 398–407
48. Krebs R, Momm C, Kounev S (2012) Architectural concerns in multi-tenant saas applications. In: CLOSER, pp 426–431
49. Krebs R, Momm C, Kounev S (2012) Metrics and techniques for quantifying performance isolation in cloud environments. In: Int'l. ACM SIGSOFT conference Quality of Software Architectures (QoSA), pp 91–100
50. Kwok YK, Ahmad I (1999) Benchmarking and comparison of the task graph scheduling algorithms. J Parallel Distrib Comput 59(3):381–422
51. Mell P, Grance T (2011) The NIST definition of cloud computing. National Institute of Standards and Technology (NIST) Special Publication 800-145, [Online] Available: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf. Last accessed Oct 2012.
52. Momm C, Krebs R (2011) A qualitative discussion of different approaches for implementing multi-tenant saas offerings. In: Software Engineering (Workshops), pp 139–150
53. Moreno-Vozmediano R, Montero RS, Llorente IM (2011) Multicloud deployment of computing clusters for loosely coupled mtc applications. IEEE Trans Parallel Distrib Syst 22(6):924–930
54. Oliner AJ, Ganapathi A, Xu W (2012) Advances and challenges in log analysis. Commun ACM 55(2):55–61
55. Oskin M, Chong FT, Farrens MK (2000) Hls: combining statistical and symbolic simulation to guide microprocessor designs. In: ISCA, pp 71–82
56. Ostermann S, Prodan R (2012) Impact of variable priced cloud resources on scientific workflow scheduling. In: Euro-Par 2012 – Parallel Processing, Springer, Lecture Notes in Computer Science, vol 7484, pp 350–362
57. Prodan R, Sperk M, Ostermann S (2012) Evaluating high-performance computing on google app engine. IEEE Software 29(2):52–58
58. Raicu I, Zhang Z, Wilde M, Foster IT, Beckman PH, Iskra K, Clifford B (2008) Toward loosely coupled programming on petascale systems. In: SC, ACM, p 22
59. Saavedra RH, Smith AJ (1996) Analysis of benchmark characteristics and benchmark performance prediction. ACM Trans Comput Syst 14(4):344–384
60. Schroeder B, Wierman A, Harchol-Balter M (2006) Open versus closed: A cautionary tale. In: NSDI
61. Sharkawi S, DeSota D, Panda R, Indukuru R, Stevens S, Taylor VE, Wu X (2009) Performance projection of hpc applications using spec cfp2006 benchmarks. In: IPDPS, pp 1–12
62. Sonmez OO, Mohamed HH, Epema DHJ (2010) On the benefit of processor coallocation in multicluster grid systems. IEEE Trans Parallel Distrib Syst 21(6):778–789
63. Spacco J, Pugh W (2005) Rubis revisited: Why j2ee benchmarking is hard. Stud Inform Univ 4(1):25–30
64. Vahdat A, Yocum K, Walsh K, Mahadevan P, Kostic D, Chase JS, Becker D (2002) Scalability and accuracy in a large-scale network emulator. In: OSDI
65. Villegas D, Antoniou A, Sadjadi SM, Iosup A (2012) An analysis of provisioning and allocation policies for infrastructure-as-a-service clouds. In: CCGrid, pp 612–619
66. Vishwanath KV, Vahdat A, Yocum K, Gupta D (2009) Modelnet: Towards a datacenter emulation environment. In: Peer-to-Peer Computing, pp 81–82
67. Walker E (2009) The real cost of a cpu hour. IEEE Computer 42(4):35–41
68. Wang G, Butt AR, Pandey P, Gupta K (2009) Using realistic simulation for performance analysis of MapReduce setups. In: HPDC Workshops, pp 19–26

69. Zaharia M, Borthakur D, Sarma JS, Elmeleegy K, Shenker S, Stoica I (2010) Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In: EuroSys, pp 265–278
70. Zhang B, Iosup A, Pouwelse J, Epema D (2010) The Peer-to-Peer Trace Archive: design and comparative trace analysis. In: Proceedings of the ACM CoNEXT Student Workshop, CoNEXT '10 Student Workshop, pp 21:1–21:2