Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 68 (2015) 67 – 77

Procedia Computer Science

HOLACONF - Cloud Forward: From Distributed to Complete Computing

A Joint Benchmark-Analytic Approach For Design-Time Assessment of Multi-Cloud Applications

Athanasia Evangelinou[a,*], Michele Ciavotta[b], George Kousiouris[a], Danilo Ardagna[b]

[a]National Technical University of Athens, Athens, Greece
[b]Politecnico di Milano, Milan, Italy

Abstract

Verifying that a software system shows certain non-functional properties is a primary concern for cloud applications. Given the heterogeneous technology offer and the pricing models currently available in the cloud market it is extremely complex to find the deployment that fits the application requirements and provides the best Quality of Service (QoS) and cost trade-offs. This task can be very challenging, even infeasible if performed manually, since the number of solutions may become extremely large depending on the number of possible providers and available technology stacks. Furthermore, with the increasing adoption of cloud computing, there is a need for fair evaluation of cloud systems. Today's cloud services differ among others by cost, performance, consistency guarantees, load-balancing, caching, fault tolerance, and SLAs. Moreover, cloud systems are inherently multi-tenant and their performance can vary over time, depending on the congestion level, provider policies, and the competition among running applications. System architects and developers are challenged with this variety of services and trade-offs. Hence, the purpose of a cloud benchmark should be to help developers when choosing the right architecture and services for their applications. In this paper we propose a joint benchmarking and optimization methodology to support the design and migration of legacy applications to Cloud. Our approach is effective in identifying the deployment of minimum costs, which provide also QoS guarantees.

Peer-review under responsibility of Institute of Communication and Computer Systems.

Keywords: Benchmarking, Cloud applications, QoS, Model Driven Design.

1. Introduction

Beyond doubt, one of the events that most impacted on the ICT world has been the advent and rise of cloud computing, which has had a tremendous repercussion on both industry and academia. The success of the Cloud has

Peer-review under responsibility of Institute of Communication and Computer Systems.
doi:10.1016/j.procs.2015.09.224

been seen by many as a paradigm shift of paramount importance that, in recent years, has also affected the social behavior of a large part of the population[1]. Suffice it to say, in fact, that everyday services like Dropbox, Netflix or Whatsapp owe part of their success to the flexibility the cloud supplies.

There is an undeniable excitement around to the possibilities offered by the wide adoption of cloud computing and this led to the flourishing of a market, where a large number of players compete with heterogeneous and constantly evolving technological offers[2]. As a consequence, the process of software design and implementation experienced a deep change. On the one hand, the Cloud has meant for developers having a set of new tools and abstractions that simplify the development and operation processes. Dynamic systems capable to react to workload fluctuations by adapting themselves in order to keep the performance unchanged can easily built and run delegating to the cloud provider the intensive tasks of infrastructure management and maintenance. On the other hand, new important issues were introduced. Vendor lock-in and performance variability in the Cloud are just some of the problems that the developer has to face. In fact, the increasing size and complexity of software systems, combined with the wide range of services and prices available in the market, puts the designer before the necessity to evaluate a combinatorially growing number of design alternatives[3] with the goal of finding a minimum-cost configuration that suits the application Quality of Service (QoS) requirements. To carry out such a task, the system designer should consider a large number of alternatives and should be able to evaluate costs (that often depends on the application dynamics) and performance for each of them. This can be very challenging, even infeasible if performed manually, since the number of solutions may become extremely large depending on the number of possible providers and available technology stacks. What is more, in many cases the performance of these service offerings may vary depending on the application types that are intended to be executed upon them and their characteristics in terms of resource usage. Intensive RAM, CPU, disk or GPU usage may be toggled between applications that have different goals, such as graphics environments, scientific components, database instances or front ends. Evaluating a specific and arbitrary application component over the entire range of offerings becomes a daunting task, especially when the deployment of the former may be subject to provider-specific or component-specific actions or code in each case. However, by compacting the necessary measurements to an abstracted level, by using indicative, categorized and generic benchmarks, one may reduce the effort needed in this task.

Furthermore, cloud systems are inherently multi-tenant and their performance may change over time, depending on the congestion level, policies implemented by the cloud provider, and competition among running applications. Assessing the performance of an application in the Cloud is a complex process that requires unbiased data and specialized models often implying skills that go beyond those commonly exhibited by software engineers. This situation calls for analytical techniques, models and benchmarks that simplify the process of performance evaluation at design-time in order to support the user in the decision making process.

This work aims at addressing and solving this problem proposing the combined use of cloud benchmarking tools to acquire service ratings over a wide set of application types based on widely used benchmarks, and an automated tool able to efficiently explore the space of design alternatives. Having such information available at design-time enables designers to make more informed decisions about the technology to adopt and the architecture to implement to migrate legacy applications and fully exploit the potentiality offered by the cloud infrastructure.

The remainder of the paper is organized as follows. In Section 2, the benchmarking approach followed in the context of the ARTIST* project is presented, while Section 3 introduces the design time exploration tool developed within MODAClouds†. In Section 4 the experimental results achieved on a case study by combining and integrating the two approaches are presented. In Section 5 related work in the respective fields is described, while the paper concludes in Section 6.

2. Cloud Benchmarking

For a successful migration of a legacy system on cloud environments one should take into account the varying performance issues of cloud services. Today's cloud services differ among others by cost, performance, consistency guarantees, load-balancing, caching, fault tolerance, SLA and programming languages. System architects and

* http://www.artist-project.eu
† http://www.modaclouds.eu

developers have to tackle with this variety of services and trade-offs. Moreover, in some cases cloud providers offer their own metrics for evaluating and guaranteeing cloud QoS.

Hence, in order to measure performance aspects and select the cloud Services that fit best to the application type that is to be migrated, an abstracted process is implemented by using suitable tools and tests, namely benchmarking. The first step of this process is to define a set of performance stereotypes based on different application categories. The main goal of these stereotypes is to extract a number of performance characteristics of the provider that are necessary for meeting QoS requirements of the migrated cloud applications. The source of these characteristics are common application types that correspond to various popular applications and have been linked to respective benchmarks that can be used to indicate a specific offering ability to solve real-life computational problems. Thus, tests have been identified with specific workload patterns that can be mapped to concrete real world applications. Benefits of such a categorization include the ability to abstract an offering performance capabilities on an application description level, thus being easily ranked according to user interests for a specific category. Concerning the characterization of a service ability from a performance point of view, ARTIST European project provides the Benchmarking Suite for benchmarking cloud platforms in order to extract performance-related data[16]. This framework is a fully automated solution for managing the benchmarking process, based on a set of third party benchmarking tools that cover as much as possible the entire application field. The set of the application types used in ARTIST is reported in Table 1.

Table 1. Benchmark Tests used in the ARTIST platform.

Benchmark Test	Application Type
YCSB	Databases
Cloudsuite	Common web aps like streaming, web serving etc.
Filebench	File system and storage
DaCapo	JVM applications

Through connectors, the benchmarking suite is able to provision cloud resources, install and execute benchmarking tools, retrieve results and, eventually, destroy provisioned resources. Compared to other available benchmarking solutions for Cloud, our approach drastically reduces the manual intervention in the benchmarking process allowing massive and large-scale tests.

Such suite allows carrying out tests and getting performance metrics in a homogeneous and independent way. Metrics can then be used to collect, over the time, quantitative information about the performance offered and constitute the baseline for the selection of cloud services during the second optimization and design-time optimization phase. The benchmarking suite comes with a GUI Web front-end for retrieving service information based on the identified application type, as well as incorporating cost aspects in a relative ranking based on user interest, with the ability to insert interest weights on either cost or performance. Results may be ranked either based on performance of the benchmark or by a combined efficiency index with cost, appearing in the following equation:

$$SE = \frac{\#Clients}{w_1 * \text{delay} + w_2 * Cost}$$

3. QoS Assessment and Optimization

Design-time exploration is supported by the SPACE4Cloud (System PerformAnce and Cost Evaluation on Cloud) a multi-platform open source tool for the specification, assessment and optimization of QoS characteristics for Cloud applications. It allows software architects to describe, analyze and optimize cloud applications following the Model-Driven Development approach. SPACE4Cloud supports MODACloudsML meta-models[4], which express cloud-specific attributes. Among other things, such models include architectural and QoS constraints (e.g., VM utilization or average application response time are lower than given thresholds), and a user-defined workload[5], necessary to assess both performance and cost of the application under different load conditions.

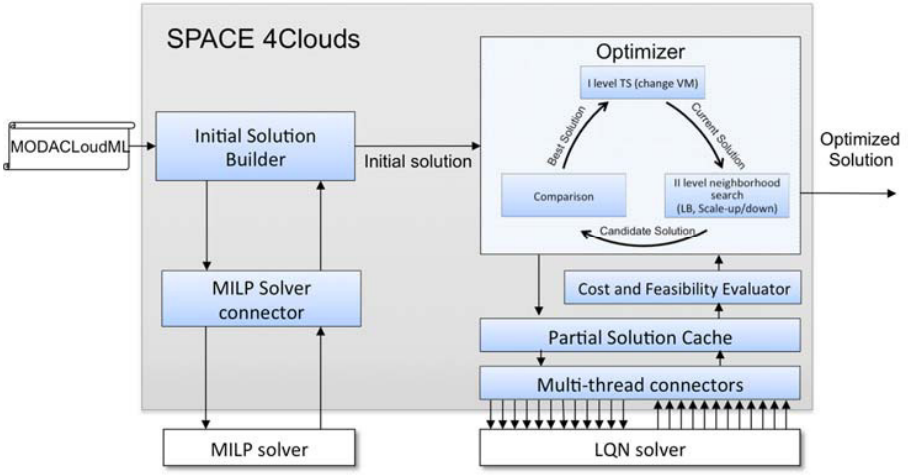


Fig. 1. Architecture of SPACE4Cloud tool.

Users can specify the models defining the application using Creator 4Clouds, an open-source IDE[33], while information about the performance of the considered cloud resources is stored in a SQL database to decouple its evolution from the one of the tool.

SPACE4Cloud can be used either to assess the cost and performance of a full-described solution (i.e., application and cloud configuration) according to a specific cost model[5] or, providing only the application model, to find a suitable (even multi-cloud) configuration that minimizes the running cost while meeting QoS requirements. In order to assess the performance of the application under development, SPACE4Cloud translates the design models, which are an extension of Palladio Component Model set (PCM)[6], into Layered Queuing Networks (LQNs)[7], performance model that are eventually solved by appropriate tools (in particular LINE[8] or LQNS[9]).

Fig. 1 highlights the main elements of SPACE4Cloud as well as the links with other third-party components. The Initial Solution Builder is in charge of generating an initial deployment by solving a Mixed Integer Linear Program (MILP) built on approximate performance models and solved by a third-party solver. All experiments reported in this paper were conducted using CPLEX[10]. The resulting solution is then used to bias the Optimizer toward promising zones of the solution space[11]. A fast and effective local-search exploration procedure that implements elements from both Tabu[12] and Iterated Local search[13] paradigms is the core of this component. The rationale of the core metaheuristic algorithm is to improve iteratively the current solution by means of local moves, which are essentially transformation actions that, starting form a solution, lead to a new, possibly better, one. The new solution is then evaluated in terms of cost and performance; part of the assessment process consists in deriving a set of performance models, namely LQN models, which are then analyzed by an external solver. The Cloud Resource Database provides information about cloud provider offers, in terms of VM costs and performance metrics, which are necessary to create the LQN models.

LQNs have been preferred to other performance models as they can be used to represent complex systems (e.g., multi-tier applications) and competition among application requests at software layer. In this work we adopted LINE[8] for all the experiments as, to the best of our knowledge, it is the only solver able to take into account cloud performance variability through random environments[34]. What is worth to be pointed out at this point is that, the evaluation of a single candidate solution of our Iterated Local Search is a time-consuming task resulting in a bottleneck for the entire optimization process. This happens because a solution comprises 24-hour deployment configurations, each of them leading to a different LQN model to be evaluated also in terms of cost and feasibility. For this reason, in order to speed up the evaluation process, we realized a multi-thread connector component managing the parallel evaluation of the 24 LQN models of a single solution and a cache-based proxy (Partial Solution Cache), to store and retrieve the evaluation of previous solutions for each hour in the time horizon. These additions greatly boost the overall evaluation process since the optimizer tends to generate similar solutions; thus by caching partial evaluations the tool is able to avoid unnecessary evaluations of performance models.

4. Experimental Analysis

The combined methodology of the two approaches appears in Fig. 2. Following the selection of the application categories and respective tests, the benchmarks are run and their results are used to populate the Raw Data DB of ARTIST. Such results are then imported within SPACE4Cloud resource DB and are exploited during the SPACE4Cloud candidate solution performance assessment to evaluate how the performance metrics of the target application changes by varying the type and size of the hosting resources. The set of benchmarks to be considered in the design-time exploration has to be configured a priori by the application developer. In the following section we report the results of our benchmarking activity. Section 4.2 shows the results we achieved for the optimal design of a social networking application case study.

4.1. Benchmarking process

In order to examine the performance aspects of cloud environments we proceeded to the execution of performance measurements on various cloud providers. The results from the benchmarking process were stored in an internal Raw data DB in order to be used during the migration of an application component for selecting the best Cloud environment in terms of performance to be hosted.

During the experimental process for investigating differences in VM performance, we utilized workloads from DaCapo[17], YCSB[18] and Filebench[19] benchmarking tools. DaCapo is designed to facilitate performance analysis of Java Virtual Machines, while YCSB and Filebench measures databases performance and file system and storage respectively. The selected workloads from each test were running on instances in three different cloud environments: Amazon EC2, Microsoft Azure, and Flexiant. For all three cases different types of VM instances were considered. A summary of the characteristics of the selected benchmarking workloads is reported in Table 2 and Table 3.

Table 2. Types of Workloads per Benchmark Category.

Filebench Workloads			
fileserver: A file system workload, similar to SPECsfs .	Varmail: A /var/mail NFS mail server emulation, following the workload of Postmark, but multi-threaded.	Videoserver: A mix of open/read/close operations on multiple files in a directory tree, plus a file append.	Webproxy: A mix of create/write/close, open/read/close, delete of multiple files in a directory tree, plus a file append

DaCapo Workloads							
xalan: transforms XML documents into HTML ones	tomcat: runs a set of queries against a tomcat server retrieving and verifying the resulting webpages	pmd: analyzes a set of Java classes for a range of source code problems	jython: interprets pybench Python benchmark	h2: executes a JDBC benchmark using a number of transactions against a banking model application	fop: parses/formats XSL-FO file and generates a PDF file	eclipse: executes jdt performance tests for the Eclipse IDE	avrora: simulates a number of programs running on a grid of AVR micro-controllers

YCSB Workloads					
A: Update heavy workload	B: Read mostly workload	C: Read only	D: Read latest workload	E: Short ranges	F: Read-modify-write

The execution of the tests took place at specific hours (at different time intervals) during a period of eight months (from July 2014 to February 2015) and the average values were extracted for each case. Moreover, the different time zones of the three different provider locations were taken into consideration so that the peak hours were the same in each zone. In Amazon EC2 case, the virtual machines were running in a North Virginia datacenter, while in the case

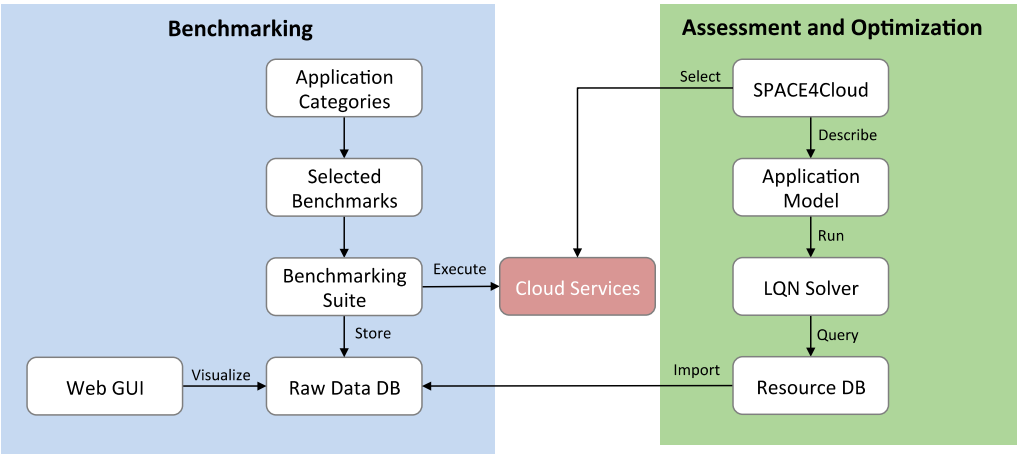


Fig. 2. Combined methodology of the two approaches.

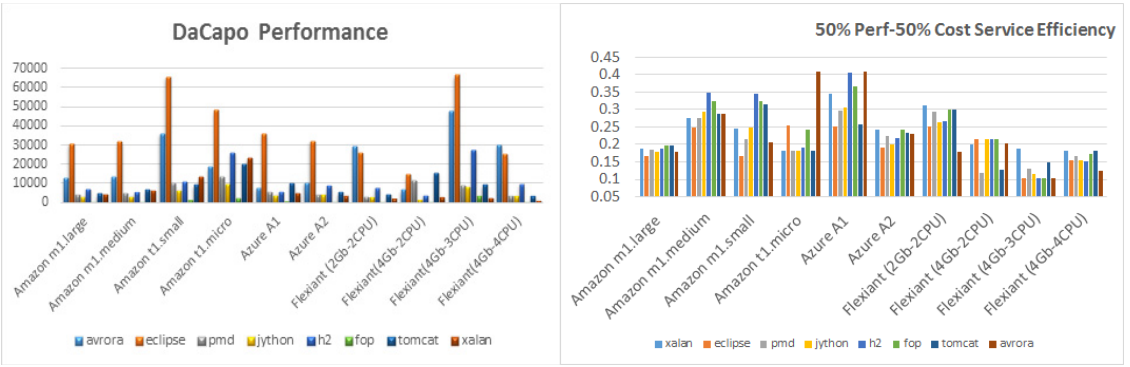
of Microsoft Azure and Flexiant the datacenters were located in Ireland and the United Kingdom, respectively. VM instance characteristics are summarized in Table 3. After completing the benchmarking process the results were retrieved from the local database, processed and the plots reported in Fig. 3 were obtained.

Table 3. Providers and VM instance types considered by the benchmarking process.

Cloud Provider	VM instance	Region
	t1.micro	N.Virginia
Amazon EC2	m1.small	
	m1.medium	
	m1.large	
Microsoft Azure	A1 small Standard	Ireland
	A2 medium Standard	
Flexiant	2GB RAM- 2CPU	UK
	4GB RAM- 3CPU	
	4GB RAM- 4CPU	
	4GB RAM- 2CPU	

In order to draw conclusions from the execution of the benchmarks, one should compare between same colour bars, indicating similar workloads (Fig. 3a). From the graphs it is evident that the performance for a specific workload varies and depends on both the type of workload and the VM instance size. For instance, for the DaCapo benchmark the workload performance across Azure (A2 Standard), Amazon (m1.large) and Amazon (m1.medium) is almost similar except for some cases where Amazon provides better results for the workload h2 while Azure was better for the workload avrora. Similar results for YCSB and Filebench have been obtained. What appears also from Fig. 3a, and is more evident in the rating produced through the application of the Service Efficiency formula in Fig. 3b, is the fact that in many cases lower capability VMs prove to be significantly more efficient. This may be due to the fact that the workload of the test does not drive the VM resources to the limit, thus when costs are included it is evident that higher capability VMs are not necessary for the specific case. Furthermore, from the absolute service ratings in Figure 3a, while in general results are reasonable (higher capability VMs produce better absolute results) a number of non-intuitive cases may be obtained, for example in the case of Avrora for Azure A1 and A2 or Flexiscale 4Gb-2CPU and 4GB-3CPU. In that case, the usage of a smaller VM not only is more efficient but portrays also enhanced absolute ratings. This may be attributed to a random process starting for example in the guest OS (which however is reduced by the repetition of the measurement process) or potentially to the way VMs are deployed, co-located and managed within the provider cluster. This co-location has been proven to cause significant performance degradation[36] and this may be true especially for large VMs that may be assigned and co-existing on the same physical host. In our case the experiments were repeated approximately 20 times, with each experiment running 10 times the respective test. In the

context of a normally operating benchmarking service of course, these numbers would need to be higher and repeated potentially in a periodic manner.



(a) (b)

Fig. 3. (a) Performance time in milliseconds for DaCapo test; (b) Service Efficiency metric for DaCapo test.

4.2. A Case Study: the MiC application

This section is devoted to present and analyze the results of a preliminary experiment with the joint benchmarking-analytic approach presented in this work. The aim of this study is to provide a comparison between results obtained using nominal values (CPU speed, number of cores, and memory) provided by cloud providers and those calculated considering benchmarking information. In this latter case, the user can even specify the workload (according to the application characteristics, e.g. CPU or I/O bound) for each application tier in order to get a more reliable assessment.

The experiment has been performed by considering a social network application called MiC (Meeting in the Cloud). MiC is a web application that lets the user to profile her/his topics of interest and to share them with similar users. Moreover, MiC identifies the most similar users in the network according to the registered users' preferences. More specifically, during the registration process, the new user selects her/his topics of interest from a set of alternatives, providing a preference for each of them in the range 1 to 5. At the end of the registration, MiC calculates the Pearson coefficient[14] based on the preferences expressed, identifies the users in the system with the most similar interests, and creates a list of contacts for the newcomer.

After the registration process, the user can log in into the MiC portal and interact with her/his best contacts by

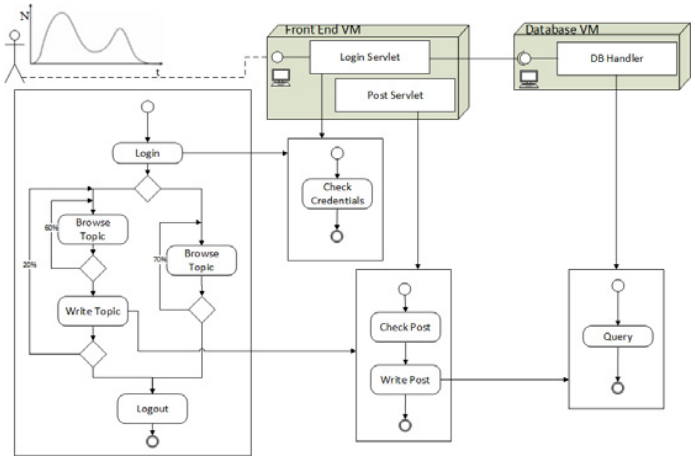


Fig. 4. Behavioral and deployment model of the MiC application.

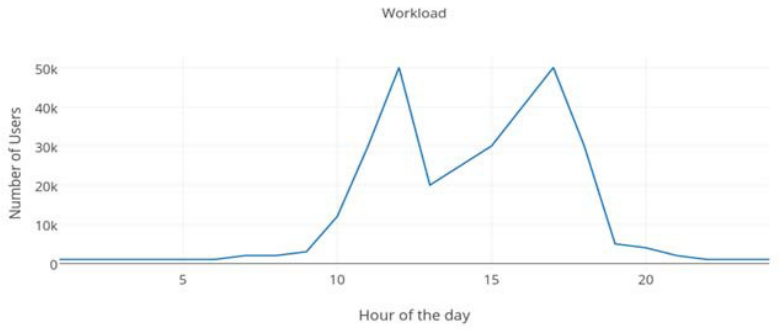


Fig. 5. Workload adopted for the experiment.

writing and reading posts on the selected topics. Users can also change their interests refining their profiles; in this case the system reacts re-evaluating the similarity and updating the list of recommended contacts. The application, whose main elements are depicted in Fig. 4, comprises two tiers, a Frontend to process the incoming http requests (via the Login and Post servlet) and a Backend running a Mysql database to stores users' profiles, messages, and best contacts lists (see Fig. 4). A PCM model for MiC application has been realized (freely available[35]), and the resource demands for each exposed functionality has been profiled within MODAClouds project.

We devised an experiment in two phases; in the first phase we analyze the SPACE4Cloud outcomes for MiC application and two cloud providers, namely Amazon and Microsoft, which are among the most important players in the market, considering the nominal performance information available on the online catalogs. A constraint of 7ms predicating on the average response time of every functionality exposed by MiC has been considered. In the second phase we first import into SPACE4Cloud resource database performance values obtained from the benchmarking activity described in the previous section. In particular, we used the results from DaCapo benchmark for the web Frontend, whereas Filebench has been selected to assess the performance of the Backend tier hosting the Mysql database. In this way SPACE4Cloud can retrieve the information that better fit the behavior of each tier.

In both phases of the experiment we optimized VM allocation for MiC application subject to a bi-modal workload with 50,000 users at its peaks, which are located in the central part of the day (see Fig. 5).

Results of this analysis are depicted in Fig. 6. We can notice that, as was to be expected, all traces follow the trend defined by the workload. Moreover, in the analysis performed using the benchmark values on average a lower number of machines is needed to fulfil the QoS requirements. The average number of VMs for each tier is reported in Fig. 6. Nonetheless, SPACE4Cloud returns a different and much more powerful VM type for the benchmark-supported case and this choice is eventually reflected on the cost per tier, which is largely increased.

The results of this analysis demonstrated that the performance advertised by cloud providers must be considered only as some guidelines for the choice of the VM and then to be able to get a reliable estimate of the performance of an application in the Cloud is necessary to resort to to more accurate benchmark results.

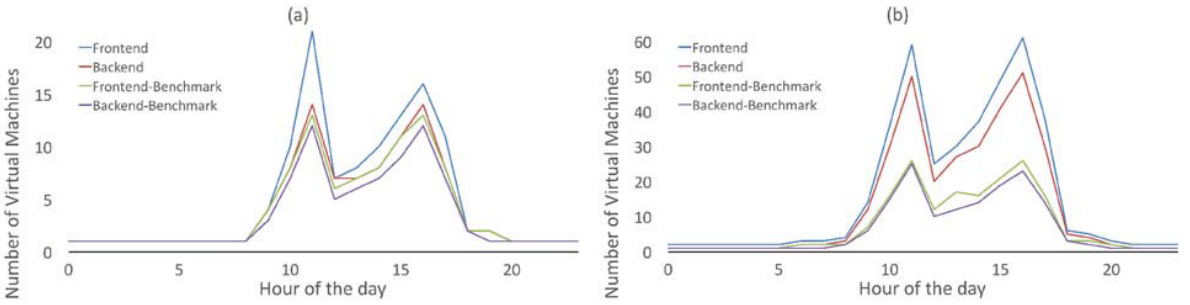


Fig. 6. Number of VMs running during the day for Amazon (a) and Microsoft (b).

Table 4. Results of the experiment.

	Amazon EC2	Microsoft Azure
	VM Type	VM Type
Frontend	c1.medium	Preview Extra Small Instance
Frontend-Bench	m1.large	A2
Backend	c1.medium	Preview Extra Small Instance
Backend-bench	m1.large	A2

5. Related Work

Our work is related mainly with three research areas: cloud benchmarking, cloud applications performance assessment, and cloud applications design space exploration. With relation to benchmarking of cloud services, CloudHarmony[21] and CloudSleuth[22] are performance measurement tools that archive the test results and make them available for access through the web API. The former offers a vast number of customizable benchmarks and provides various performance metrics with focus on application, CPU, Disk I/O, Memory I/O etc. for various cloud providers online. However, there is one aspect that could be ameliorated with regard to this approach. Since a large number of benchmarking tests is included in the list it would be desirable to limit the scope of tests to interesting shift in measurements. CloudSleuth can built up a cloud application benchmark which provides availability and response time of various cloud providers online by continuously monitoring a sample application running on top cloud computing providers however the focus in on web-based applications.

With regard to performance frameworks, PerfKit Benchmarker[23] is a living open source tool for benchmarking cloud, allowing developers to get a transparent view of application throughput, latency, variance and overhead. This framework includes popular benchmarking workloads that can be executed across multiple cloud providers. However, the only limitation is that PerfKit tools are currently support only Amazon AWS, Microsoft Azure clouds and Google Compute Engine. Finally, Skymark[24] is an extensible and portable performance analysis framework for IaaS clouds. It enables the generation and submission of real or synthetic complex workloads across IaaS cloud environments and it can analyze the impact of individual provision and allocation policy specified by the user, prior to the initiation of the experiment. Through the accumulation of statistical information regarding the workload execution, the framework is able to carry out a performance analysis of the underlying IaaS systems.

As far as quality assessment is concerned, the Object Management Group (OMG) introduced for this purpose two UML profiles specially tailored to model QoS, called Schedulability, Performance and Time (SPT)[25] and Modeling and Analysis of Real-Time and Embedded Systems MARTE[26]. These profiles allow to express some performance characteristics but still lack the proper support to model the heterogeneity of the cloud infrastructure. A similar approach led to PCM[27], a language that can be used to model an application and its non-functional properties. Once an application is fully described performance models can be automatically derived and solved in order to obtain a prediction on the application behavior. However, since the space of design alternatives for a single application can be very large, the task of finding the most suitable one is often arduous and time demanding; for this reason solutions able to guide the user have been proposed. The majority of them leverage particular algorithms to efficiently explore the design space in seeking for solutions that optimize particular quality metrics. Examples of techniques usually adopted are evolutionary algorithms and integer linear programming. Both ArcheOpterix[15,28] and PerOpterix framework[3,29] use genetic algorithms to generate candidate solutions. Other work presents an efficient tabu search (TS) heuristic[30] that has been used to derive component allocation in the context of embedded systems considering availability constraints. The SASSY[31] framework starts from a model of a service-oriented architecture, performs service selection and applies patterns like replication and load balancing in order to fulfill quality requirements. Finally, Frey et al.[32] proposed a combined metaheuristic-simulation approach based on a genetic algorithm to derive deployment architecture and runtime reconfiguration rules to move a legacy application to the Cloud.

6. Conclusions

The ability to measure cloud services on a variety of different application types enabled us to abstract the process of service measurement and selection, avoiding repeating such analysis for each and every individual application component one needs to deploy. However, the optimal selection for a specific application deployment may include the selection from a multitude of providers and consider the performance individual VM types available. Thus the combination of the two approaches (ARTIST Benchmarking Suite and SPACE4Cloud) enables the ability to apply this optimal decision on the entire application chain level, while taking under consideration user interests in terms of cost and performance.

This has been demonstrated by the MiC use case where different resources have been selected.

As future work, since in many cases even the application owner may not be aware of the overall VM behavior, especially in cases when different types of components are grouped in the same VM, it will be necessary to undergo a step further, i.e., a profiling and classification step[20]. During such step, the overall behavior of a VM consisting of an arbitrary number of cooperating components, will be measured and categorized according to the behavior of the selected benchmarks, in order to classify the former to one of the latter based on the dominant behavior and potential bottlenecks at the system level in terms of resource utilization.

Acknowledgements

The research leading to these results is partially supported by the European Community Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 317859, in the context of the ARTIST Project and under grant agreement no 318484 in the context of the MODACLOUDS project

References

1. Ferkoun M. IBM Thoughts on Cloud, http://www.thoughtsoncloud.com/2013/04/how-cloud-computing-is-impacting-everyday-life/ 2013
2. Gartner Group. Hype Cycle for Cloud Computing, https://www.gartner.com/doc/2807621/hype-cycle-cloud-computing-, 2014.
3. Koziolek A., Koziolek H., Reussner R. PerOpteryx: Automated Application of Tactics in Multi-objective Software Architecture Optimization. In QoSA 2011 Proc., QoSA-ISARCS '11, pages 33–42, New York, NY, USA, 2011. ACM.
4. Ferry N., Rossini A., Chauvel F., Morin B., Solberg A. Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems. In IEEE CLOUD 2013 Proc., pages 887–894. IEEE Computer Society, 2013.
5. Franceschelli D., Ardagna D., Ciavotta M., Di Nitto E. Space4cloud: a tool for system performance and cost evaluation of cloud systems. In Proc. MultiCloud '13, pages 27–34, New York, NY, USA, 2013. ACM.
6. Becker S., Koziolek H., Reussner R. The palladio component model for model-driven performance prediction. Journal of Systems and Software, 82(1):3–22, 2009.
7. Rolia J., Sevcik K. The method of layers. Software Engineering, IEEE Transactions on, 21(8):689–700, 1995.
8. Perez J., Casale G. Assessing SLA Compliance from Palladio Component Models. In SYNASC 2013 Proc., pages 409–416, Sept 2013.
9. Franks G., Al-Omari T., Woodside M., Das O., Derisavi S. Enhanced modeling and solution of layered queueing networks. Software Engineering, IEEE Transactions on, 35(2):148–161, March 2009.
10. IBM ILOG CPLEX optimization studio. http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud/.
11. Ardagna D., Gibilisco G., Ciavotta M., Lavrentev A. A multi-model optimization framework for the model driven design of cloud applications. In SBSE 2014 Proc. 2014.
12. Glover F. Tabu search: part i. ORSA Journal on computing, 1(3):190–206, 1989.
13. Lourenço H. R., Martin O. C., Stützle T. Iterated local search: Framework and applications. In Handbook of Metaheuristics, pages 363–397. Springer, 2010.
14. Pearson K. Note on regression and inheritance in the case of two parents. Proc. of the Royal Society of London, 58:pp. 240–242, 1895.
15. Aleti A., Björnander S., Grunske L., Meedeniya I. Archeopterix: An extendable tool for architecture optimization of aadl models. In Proc. of Workshop MOMPES 2009.
16. Kousiouris G., Giammatteo G., Evangelinou A., Galante N., Kevani E., Stampoltas C., Menychtas A., Kopaneli A., Ramasamy Balraj K., Kyriazis D., Varvarigou T., Stuer P., Orue-Echevarria Arrieta L. A Multi-Cloud Framework for Measuring and Describing Performance Aspects of Cloud Services Across Different Application Types, In Proc. of MultiCloud 2014.
17. DaCapo Benchmarking Suite:http://www.dacapobench.org/
18. Cooper, B.F., et al. Benchmarking cloud serving systems with YCSB. In Proceedings of the 1st ACM symposium on Cloud computing, pp. 143-154. ACM, 2010.
19. Filebench Benchmarking Suite: http://filebench.sourceforge.net/
20. ARTIST Deliverable D7.4 "Classification methods and tools", ICCS and other partners, March 2015
21. CloudHarmony.com TM. Cloudharmony services. Online, July 2015, available at:http://cloudharmony.com/services/.

22. Compuware Cloudsleuth Platform, available at: https://cloudsleuth.net/
23. PerfKitBenchmark, avaiable at: https://github.com/GoogleCloudPlatform/PerfKitBenchmarker
24. Antoniou A. Performance evaluation of cloud infrastructure using complex workloads. Master's thesis, Delft University of Technology, 2012.
25. OMG. UML Profile for Schedulability, Performance, and Time Specification, 2005.
26. OMG. A uml profile for marte: Modeling and analysis of real-time embedded systems, 2008.
27. Becker S., Koziolek H., Reussner R. The palladio component model for model-driven performance prediction. Journal of Systems and Software, 82(1):3–22, 2009.
28. Meedeniya I, Buhnova B, Aleti A, Grunske L. Architecture-driven reliability and energy optimization for complex embedded systems. In QoSA 2010.
29. Koziolek A. Automated Improvement of Software Architecture Models for Performance and Other Quality Attributes. PhD thesis, Germany, 2011.
30. Ouzineb,M, Nourelfath, M, Gendreau, M. Tabu search for the redundancy allocation problem of homogenous series-parallel multi-state systems. Reliability Engineering & System Safety 93(8), 1257–1272 (2008)
31. Menascé, D. A, Ewing, J. M, Gomaa, H, Malek, S, Sousa, J. P. A framework for utility-based service oriented design in SASSY. In: WOSP/SIPEW 2010.
32. Frey, S., Fittkau, F., Hasselbring, W. Search-based genetic optimization for deployment and reconfiguration of software in the cloud. In: ICSE 2013.
33. Almeida da Silva M. A., Ardagna D., Ferry N., Pérez J. F. Model-Driven Design of Cloud Applications with Quality-of-Service Guarantees: The MODAClouds Approach, MICAS Tutorial. MICAS-SYNASC 2014.
34. Casale G., Tribastone M., Harrison P.G. Blending randomness in closed queueing network models. Performance. Evaluation. 82: 15-38 2014.
35. MiC application PCM model: https://github.com/deib-polimi/modaclouds-models/tree/master/MiCforJSS-2tier
36. Kousiouris G., Cucinotta T., Varvarigou T., The Effects of Scheduling, Workload Type and Consolidation Scenarios on Virtual Machine Performance and their Prediction through Optimized Artificial Neural Networks, The Journal of Systems and Software, 84, 8, pages 1270-1291, 2011.