Computers and Electrical Engineering 47 (2015) 69–81

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Workload modeling for resource usage analysis and simulation in cloud computing

Deborah Magalhães [a,b,*], Rodrigo N. Calheiros [b], Rajkumar Buyya [b], Danielo G. Gomes [a]

[a] Group of Computer Networks, Software Engineering, and Systems (GREat) Departamento de Engenharia de Teleinformática, Universidade Federal do Ceará, Fortaleza-Brazil
[b] Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, The University of Melbourne, Australia

ARTICLE INFO

Article history:
Received 26 December 2014
Revised 25 August 2015
Accepted 27 August 2015

Keywords:
Cloud computing
Workload modeling and characterization
Simulation
Distribution analysis
Web applications

ABSTRACT

Workload modeling enables performance analysis and simulation of cloud resource management policies, which allows cloud providers to improve their systems' Quality of Service (QoS) and researchers to evaluate new policies without deploying expensive large scale environments. However, workload modeling is challenging in the context of cloud computing due to the virtualization layer overhead, insufficient tracelogs available for analysis, and complex workloads. These factors contribute to a lack of methodologies and models to characterize applications hosted in the cloud. To tackle the above issues, we propose a web application model to capture the behavioral patterns of different user profiles and to support analysis and simulation of resources utilization in cloud environments. A model validation was performed using graphic and statistical hypothesis methods. An implementation of our model is provided as an extension of the CloudSim simulator.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction

Clouds are being used as a platform for various types of applications with different Quality of Service (QoS) aspects, such as performance, availability and reliability. These aspects are specified in a Service Level Agreement (SLA) negotiated between cloud providers and customers. The failure to comply with QoS aspects can compromise the responsiveness and availability of service and incur SLA violations, resulting in penalties to the cloud provider. The development of resource management policies that support QoS is challenging and the evaluation of these policies is even more challenging because clouds observe varying demand, their physical infrastructure has different sizes, software stacks, and physical resources configurations, and users have different profiles and QoS requirements [1]. In addition, reproduction of conditions under which the policies are evaluated and control of evaluation conditions are difficult tasks.

In this context, workload modeling enables performance analysis and simulation, which brings benefits to cloud providers and researchers. Thereby, the evaluation and adjustment of policies can be performed without deployment of expensive large scale environments. Workload models have the advantage of allowing workload adjustment to fit particular situations, controlled

* Corresponding author at.: Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, The University of Melbourne, Australia. Tel.: +61 415722721.
E-mail address: deborah.vm@gmail.com, deborah@great.ufc.br (D. Magalhães).

http://dx.doi.org/10.1016/j.compeleceng.2015.08.016
0045-7906/© 2015 Elsevier Ltd. All rights reserved.

modification of parameters, repetition of evaluation conditions, inclusion of additional features, and generalization of patterns found in the application [2], providing a controlled input for researchers. For cloud providers, the evaluation and simulation of resource management policies allow the improvement of their systems' QoS. Finally, the simulation of workloads based on realistic scenarios enables the production of tracelogs, scarce in cloud environments because of business and confidentiality concerns [3,4].

Workload modeling and characterization is especially challenging when applied in a highly dynamic environment, such as cloud data centers, for different reasons:(i) heterogeneous hardware is present in a single data center and the virtualization layer incurs overhead caused by I/O processing and interactions with the Virtual Machine Monitor (VMM); and (ii) complex workloads are composed of a wide variety of applications submitted at any time, and with different characteristics and user profiles. These factors contribute to a lack of methodologies to characterize the different behavioral patterns of cloud applications.

To tackle the above issues, a web application model able to capture the behavioral patterns of different user profiles is proposed to support analysis and simulation of resources utilization in cloud environments. The proposed model supports the construction of performance models used by several research domains. Performance models improve resource management because they allow the prediction of how application patterns will change. Thus, resources can be dynamically scaled to meet the expected demand. This is critical to cloud providers that need to provision resources quickly to meet a growing resource demand by their applications.

In this context, the main contribution of this paper is a model capable of representing resource demand of Web application supported by different user profiles in a context of cloud environment. The workload patterns are modeled in the form of statistical distributions. Therefore, the patterns fluctuate based on realistic parameters in order to represent dynamic environments. A model validation is provided through graphical and analytical methods in order to show that the model effectively represents the observed patterns. A secondary contribution of this paper is the validation and implementation of the proposed model as an extension of the CloudSim simulator [1], making the model available for the cloud research community.

The rest of the paper is organized as follows: Section 2 presents the challenges and importance of workload modeling in clouds. Section 3 describes related works. Section 4 details the adopted methodology and how it was achieved. Section 5 presents and discusses the modeling and simulation results. Section 6 concludes and defines future research directions.

2. Problem statement and motivation

Workload characterization and modeling problems have been addressed over the last years, resulting in models for generation of synthetic workloads similar to those observed on real systems [2]. The main objective of such models is enabling the behavior patterns detection on the collected data.

2.1. Challenges of workload modeling in clouds

Workload modeling and characterization is especially challenging when applied in a highly dynamic environment such as a cloud, for various reasons, as discussed below:

1. Hardware platforms heterogeneity: Information Technology (IT) managers update about 20% to 25% of their platforms every year [5], resulting in the combination of different hardware in the same data center. Besides, the virtualization layer promotes an overhead caused by I/O processing and interactions with the Virtual Machine Manager (VMM). This overhead depends on the hardware platform.
2. Business and confidentiality: due to business and confidentiality reasons, there are few cloud tracelogs available for analysis. Thus, there is a lack of methodologies to characterize the different behavioral patterns of cloud applications [3,4]. Nevertheless, recent efforts in this direction, such as, Google TraceLog [6] and Yahoo!, enable data analysis and characterization for specific scenarios [7].
3. Workload complexity: the cloud hosts a wide variety of applications submitted at any time, with different characteristics and user profiles, which have heterogeneous and competing QoS requirements [1]. This leads to complex workloads depending on users' behavior and resource consumption. Thus, it is challenging to predict workload patterns over time.

2.2. Importance of workloads modeling in cloud

Workload modeling increases the understanding of typical cloud workload patterns and leads to more informed decisions to better manage resources [3,7]. From the workload characterization, performance models can be constructed to support research topics such as energy-efficiency and resource management and to answer important research questions, such as: how are overall cloud data center utilization levels affected by user behavior? How cloud data center energy efficiency can be improved while the QoS is maintained?

In addition, workload modeling in cloud computing enables performance analysis and simulation, which brings benefits to cloud providers and researchers as it allows: (i) the evaluation, through simulation, of resource management policies allowing the improvement of cloud services' QoS; (ii) the evaluation of these policies without deployment and execution of the applications in expensive large-scale environments; and (iii) the simulation of realistic cloud environments with controlled modification, adjustment, and repetition. The use of simulation models enables the production of tracelogs based on realistic scenarios, filling the gap previously identified in the cloud computing area [3].

Table 1
Summary of related work.

Authors	Modeling approach	Application type	Exploratory analysis	Parameter estimation	GoF test	Validation
Chen et al. [4]	Distribution analysis	MapReduce	Yes	No	No	No
Ganapathi et al. [8]	Distribution analysis	MapReduce	No	No	No	No
Kavulya et al. [7]	Distribution analysis	MapReduce	Yes	Yes (single)	Yes	No
Moreno et al. [3]	Distribution analysis and cluster	MapReduce	Yes	No	No	Yes
Grozev and Buyya [9]	Analytical model	Web	No	No	No	Yes
Our proposal	**Distribution analysis**	**Web**	**Yes**	**Yes (multi)**	**Yes**	**Yes**

3. Related work

The conflict of priorities between cloud providers, aiming high resource utilization with low operating costs, and MapReduce applications users addressing small execution time, led to characterization and modeling of this application type [4,7,8]. Chen et al. [4] developed a tool for generation of realistic workloads with the goal of analyzing the tradeoff between latency and resources utilization. In attempt to obtain resource utilization data, statistical models were used to generate the job stream processed in the environment provided by Amazon's Elastic Cloud Computing (EC2). However, the authors do not present information concerning the distributions, parameters and Goodness of Fit (GoF) tests. Thus, the reproduction of the model by other researchers is infeasible.

Ganapathi et al. [8] proposed a model to predict the execution time of MapReduce jobs in order to maximize performance while minimizing costs. A workload generator based on statistical models was used to guide the prediction. However, the authors did not consider the full distribution to build the model, instead, the distribution is estimated from the 1st, 25th, 75th and 99th percentiles, causing information loss that compromises the accuracy of the model.

Kavulya et al. [7] characterized resource utilization patterns and sources of failures to predict job completion times in MapReduce applications. They did not present an analysis of data justifying the distributions used. Furthermore, they use only the method of Maximum Likelihood Estimation (MLE), which is sensitive to outliers. Our approach uses different estimation methods and the results show that the estimator has great influence on the predictor accuracy. Kavulya et al. [7] presented Kolmogorov–Smirnov (KS) values with no critical value. Thus, it is not clear which set of distributions and parameters provide a good fit to the modeled data.

Most works discussed in this session focus on resource utilization imposed by jobs, tasks, and requests. However, different types of users directly impact the cloud workload as observed by the cloud provider. In view of this, Moreno et al. [3] and Grozev and Buyya [9] created models based on resource utilization and users' behavioral patterns. Moreno et al. [3] proposed a new approach for characterization of the Google trace log in the context of both user and task in order to derive a model to capture resource utilization patterns. However, although the model parameters are presented, estimation methods and the results of the goodness-of-fit tests are omitted. These factors compromise the use of the model by other researchers.

Grozev and Buyya [9] also made use of the Rice University Bidding System (RUBiS) workload and implemented the model in the CloudSim. The Central Processing Unit (CPU) load is modeled in terms of average number of instructions required by a user session. However, the average is not robust, which makes it easily influenced by peak usage. Also, the authors adopt a non-statistical approach for modeling, thus, they did not analyze dispersion and shape measurements, which are relevant for a statistical workload characterization.

From these related works, we observe that the application characterization and modeling supports researchers and cloud providers in understanding complex tradeoffs and predicting applications behavior. This understanding leads to more efficient techniques for resources management. However, the lack of a well-defined methodology, containing steps to achieve distributions, estimate parameters, and goodness-of-fit tests, prevents reproduction and usage of models. In this scenario, our proposal consists in a Web application model achieved through a well-defined methodology. A summary of the related work is presented in Table 1.

4. Material and methods

4.1. Methodology

Fig. 1 presents the methodology [2] used to create the proposed models. This methodology begins with users submitting requests to cloud environment while the operational metrics are measured and stored in data logs.

After the monitoring and tracing, the user activity and performance modeling is carried out based on three steps: (i) statistical analysis, which analyzes the data characteristics, determines if some data transformation is necessary and defines the candidate distributions to represent the model; (ii) parameter estimation, which, given the distribution selected in advance, uses estimation methods to set the parameters of the model based on the collected samples; and (iii) GoF tests, which are methods to evaluate whether the distributions and their respective parameters, provide satisfactory approximation to the empirical data.

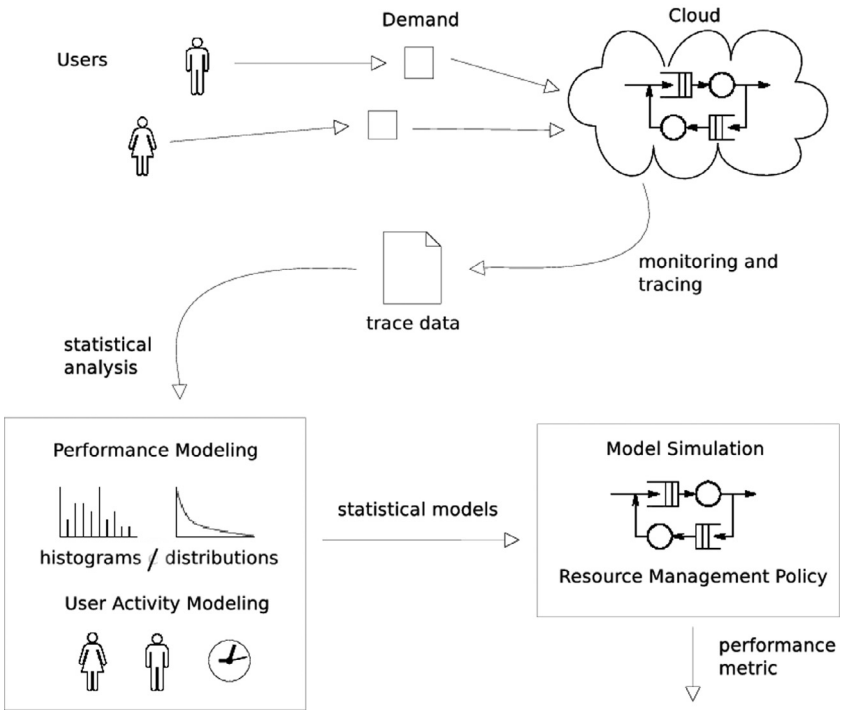

Fig. 1. User activity and performance modeling methodology for analysis and simulation in cloud environments (adapted from [2]).

Table 2
Physical sever and VMs specifications.

Instance name	Configuration
RUBiS client	m1.small | 2 GB RAM | 1 VCPU | 20 GB Disk
servers	m1.medium | 4 GB RAM | 2 VCPU | 40 GB Disk
Physical server	64 GB RAM | 12 VCPU (Xeon 6C E5-2620) | 2.1TB Disk

4.2. Workload

The cloud environment is suitable for interactive real-time applications. However, recent works have been focusing on provisioning and scheduling techniques for batch applications, while cloud resource management in interactive applications has not received much attention so far [9]. In view of this, we utilize the widely used RUBiS [10] benchmark in order to evaluate the impact of the user in the resource consumption patterns.

In the context of cloud computing, RUBiS has been employed in the construction of a performance model of a 3-tier application in cloud environments [9] and on the evaluation of a Benchmark-as-a-Service platform, enabling the determination of the bottleneck tier and to tune the application servers to improve application performance [11]. The used methodology is not limited to RUBiS and thus it can be extrapolated to other workload categories [2].

The RUBiS benchmark is an auction site, based on eBay.com, implementing functionalities such as selling, browsing, and bidding. It provides two user profiles: browsing profile, including only read interactions with the database, and the bidding profile, which includes 85% read and 15% reading and writing in a database [10].

4.3. Monitoring and tracing

A series of decisions must be made before the application modeling process is carried out. This includes the definition of the experimental environment and interest metrics. The experimental environment consisted of a private cloud comprising three high-performance servers interconnected through a Gigabit Ethernet switch. The physical resources management was accomplished via OpenStack (Grizzly) [12]. Our experiments had two types of VMs, small and medium, whose hardware configurations are described in Table 2. The server had twice the Random Access Memory (RAM) existed on the client. For compatibility with the monitoring tool Bro [13], RUBiS version 1.4.3 was used.

The client VM was configured with the load generator. The server VM hosts a Linux, Apache, MySQL and Hypertext Preprocessor (PHP) (LAMP) stack. The Linux version on the server was the same used on the client (Ubuntu 12.04.2 LTS). The MySQL

Table 3
Experiment # 1 – RUBiS client configuration.

Configuration	Value
Rounds	100
User profile	Browsing/Bidding
Session run time	15 min
Think time	7–8 s
Up ramp time	2 min
Down ramp time	1 min

relational database (14.14) was configured with a maximum number of connections equal to the number of virtual CPUs on the server VM. Finally, we used version 5.3.10 of the PHP scripting language.

To accurately model the user behavior impact on the processor, we monitored the total number of executed instructions and how they arrived in the processor during a user session through the CPU utilization, which was the second metric analyzed in this study.

RUBiS randomly generates the size of user sessions from a negative exponential distribution. This value is static for any CPU that processes the session. Converting such a value to time, according to the CPU clock, is reasonable if all CPUs have the same characteristics (architecture, core number, manufacturer, etc.). When CPU characteristics are heterogeneous, estimation of the execution time is a difficult task. Thus, the instructions number is captured with the aim of creating a model that characterizes the user session in terms of number of executed instructions. Therefore, the session execution time varies according to the hardware's processing capacity.

The third and fourth metrics analyzed in this study were memory and disk utilization. The latter one is measured in terms of Transactions Per Second (TPS), which represents the number of reads and writes transfers per second performed on the disk. This measure is more easily extrapolated to different disk hardware settings when compared to percentage of disk utilization.

The fifth metric considered was the response time, which was monitored and traced to characterize the Quality of Service (QoS) provided by RUBiS to its users. In this work, the user response time definition consists of the sum of the reply time and transfer time. The reply time is the amount of time between the instant at which the client receives the first reply packet, and the time at which the client issued the HTTP request. Transfer time is the time taken for the entire reply to be transferred to the client [14]. Since the bidding profile offers a more realistic mix of user actions (registration, bid on items and consult current bids, rating, and comments left by other users), its response time provides a more meaningful indication of performance than the response time of the Browsing profile.

Nonetheless, Hashemian et al. [14] verified that RUBiS introduces significant errors in the measured end-user response times. For this reason, this work adopts their monitoring approach, where the tcpdump tool [15] is used to capture all HTTP packets. At the end of the user session, a modified version of the Bro tool [13] captures the tcpdump output file and calculates the reply times and transfer times of HTTP requests and records them. The Bro was executed in an off-line manner to avoid the overhead on the client.

The metrics samples were obtained through 100 executions of the same experiment, which consists of requests submission from a single client to Apache and MySQL servers. The data was captured throughout the processing of the user session on the server. RUBiS was configured using the parameters presented in Table 3.

Empirically, we observed that 100 repetitions of the experiment are large enough to capture most of the patterns found in the user sessions. To avoid interference of outliers, our statistical analysis is performed on the median of the repetitions. In the experiments, both RUBiS user profiles are used. RUBiS reproduces the think time via negative exponential distribution with mean between 7 and 8 s, as defined by TCP-W benchmark [16].

Most of e-commerce sessions last less than 16.66 min [17] and the reasonable longest time for user Web browsing are 15 min [18]. In this context, the session time uses the negative exponential distribution with the mean equals to 15 min, consistent with the RUBiS specifications [10].

4.4. Performance modeling

The patterns identified in collected data were modeled in the form of statistical distributions. Then, given n observations, $x_1, x_2, \ldots, x_n$, representing a metric from an unknown population, our goal was to find the Probability Density Function (PDF) that represents the data adequately. The PDF has the form $f(x, \theta)$, where θ is the vector of parameters estimated from the available samples.

Understanding the characteristics of such samples is important to determine which distributions better approximate the collected data. To this end, we performed a data analysis through test statistics and graphs, focusing on statistics that characterize the shape of the data: skewness and kurtosis.

4.4.1. Parameters estimation

Since the characteristics of the sample are known, the distribution candidates for the metrics *number of instructions, CPU utilization, memory utilization*, and *disk transactions per second* were selected considering both user profiles supported by RUBiS.

Additionally, the parameters of *response time* metric were estimated considering the Bidding profile. The Generalized Lambda (GL) distribution is able to represent different shapes, including those with negative skewness. Therefore, we used it as an alternative to represent the observed number of instructions. The GL is a generalization of the four parameters of lambda distribution family [19]: the first parameter is the location, represented by the median (μ), the second parameter is the scale, represented by the inter-quartile range (σ), and the last two parameters are related to shape, featuring the skewness (α_3) and the steepness of the distribution (α_4), respectively.

Another alternative selected is the Generalized Extreme Value (GEV) distribution [20]. This distribution has three parameters, namely location (μ), scale (σ), and shape (ξ) and can represent a variety of forms that include three different distributions: (i) Gumbel, when $\xi = 0$, (ii) Frechet, when $\xi < 0$, and (iii) Weibull, when $\xi > 0$.

To represent the CPU, memory and disk utilization, 21 different continuous distributions were analyzed for the user profiles. Among them, we highlight the Generalized Weibull Distribution (GWD) [21] and 3-parameter Error distribution [22]. Both have the location (μ), scale (σ) and shape (ξ) parameters. As GL and GEV, these distributions can also be specialized to represent other distributions, which makes them extremely flexible in fitting different kinds of data.

After choosing the distributions, it is necessary to estimate the parameters of the models. One option is to calculate the moments of the sample and use them as estimators for the moments of the distribution. However, this approach is highly sensitive to outliers, especially for the third and fourth moments. This limits its utilization in the case of distributions used in this work [2,23]. As noted in Table 5, five different estimation methods were used [23]: Maximum Log-Likelihood (mle), Histogram Fitting (hist), Quantile Matching (quant), Probability Weighted Moments (pwm) [24], and Maximum Product of Spacing Estimator (mps) [25].

The CPU utilization metric proved much harder to adjust when compared to the other metrics. Even compared to a higher number of distributions (total = 21), we obtained high values of D and values of p-Value lower than the critical value. Therefore, a parameter estimation previous phase called pre-fitting was performed. This step consists in changing the data scale to make the normality assumption plausible. Thereby, the following mathematical transformations were applied to this metric: log, square-root, and inverse.

4.4.2. Goodness of fit

Once the selected parameters and their distributions are known, the next step was the determination of the models that fit to the data through GoF tests. The GoF statistics verifies if the empirical and theoretical data belong to the same distribution. In this study, two methods were used to assess if the selected distributions provide good fit to the data: one graphic method using Quantile–Quantile (Q–Q) plots, and one analytical method using the KS test.

The Q–Q plots technique consists in the calculation of empirical and theoretical distribution quantiles and plotting one in terms of the other [2]. It allows verifying if two data sets belong to the same distribution and other aspects simultaneously, for example, the presence of outliers.

The KS test evaluates the hypothesis that the observed data belongs to a population that follows one or more probability distributions. This test can be defined as the hypotheses: (i) *Null Hypothesis (H_0):* the observed data follows the specified distribution and (ii) *Alternative Hypothesis (H_a):* the observed data does not follow the specified distribution.

The KS test has an important limitation: the parameters from $F(x)$ must be known in advance rather than estimated from the observed data, as performed in this work. In order to circumvent this limitation, we used the bootstrapping method [26].

4.5. Model simulation

In this section, we detail the implementation of the web application modeling as an extension of the CloudSim simulator [1]. The application modeling was developed in CloudSim because this simulator contains abstractions for representing cloud infrastructures and power consumption.

4.5.1. Workload generation and validation

The process of generating the load simulator is shown in Fig. 2. At the start of the simulation, the *ECommerceApp* class is instantiated with the parameters number of users and arrival rate of users, which can be a fixed value or generated by a distribution, and, finally, the user profiles (browsing and bidding). With these parameters, the *ECommerceApp* instantiates the *UBehavior* class responsible for encapsulating the users' behavior. This behavior is defined by the statistical distributions that represent the total number of instructions, CPU, memory and disk utilization, and response time. The models are developed using the R statistical language [27] that communicates with CloudSim through the *REngine* library.

Once the users' behavior (number of requests, their arrival time, their size in terms of number of instructions, memory and disk demands, and response time) is known, a *USession* is instantiated for each user. Afterwards, *USession* instantiates the set of *Request* that will compose it. In each simulation step, one or more requests are processed until all are completed.

Graphical and statistical hypothesis test approaches were used to evaluate the accuracy of the model in the simulator by comparing the simulated against the observed data. The Wilcox Mann–Whitney (WMW) hypothesis test [28] consists in the evaluation of the hypothesis that the simulated data belongs to the population that follows the probability distribution of the observed data. If p-Value $> \alpha$, the hypothesis cannot be rejected. Otherwise, the null hypothesis is rejected. The level of significance was set at $\alpha = 0.05$ for all the tests.

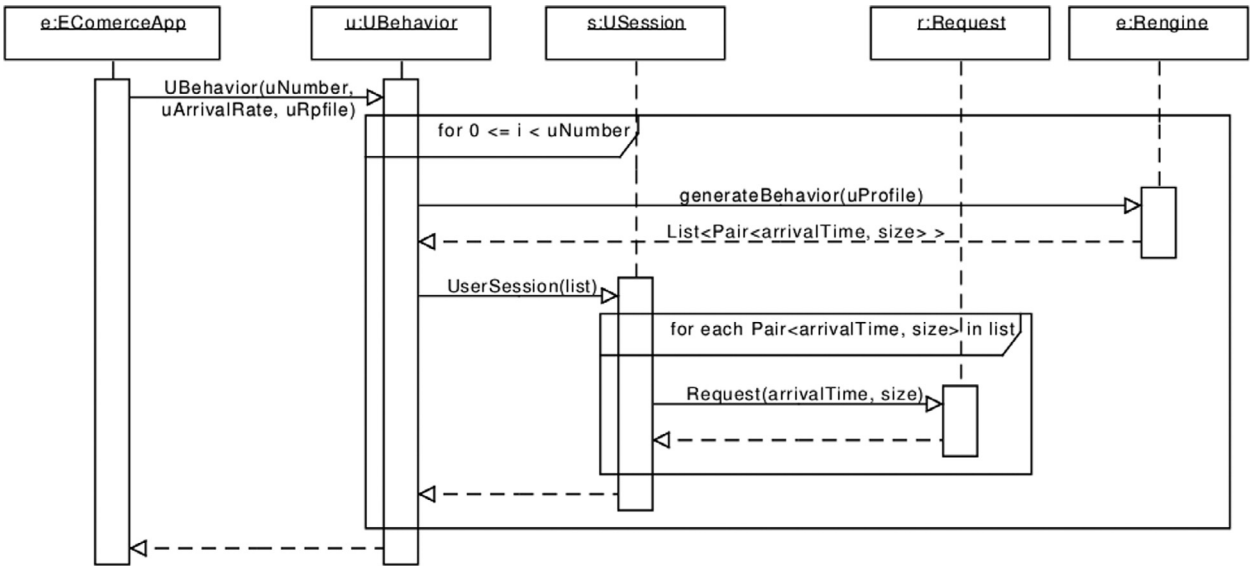

Fig. 2. Workload generation.

Table 4
Observed metrics statistics: number of instructions, CPU, memory, and disk utilization, and response time.

Statistic	Type	Browsing				Bidding				
		Instructions	CPU (%)	RAM (%)	Disk (TPS)	Instructions	CPU (%)	RAM (%)	Disk (TPS)	Resp. time (ms)
Minimum	Location	4.00e + 08	0	18.92	0	4.26e + 08	0	58.17	0	0.74
1st Quartile	Location	4.69e + 08	0	19.97	0	4.72e + 08	0	58.76	0	0.85
Median	Location	4.87e + 08	0	20.02	0	4.87e + 08	0	58.84	0	0.87
Mean	Location	4.75e + 08	0.59	20.01	0.11	4.79e + 08	0.55	58.82	0.08	1.00
3rd Quartile	Location	4.89e + 08	0.50	20.08	0	4.89e + 08	0.50	58.92	0	0.90
Maximum	Location	4.92e + 08	82.50	20.13	26	4.91e + 08	79.90	59.01	8	9.42
Std. deviation	Dispersion	2.02e + 07	4.99	0.11	1	1.38e + 07	4.80	0.16	0.43	0.68
Skewness	Shape	−1.67	12.28	−3.10	18.05	−1.52	13.00	−2.05	8.28	8.80
Kurtosis	Shape	2.25	156.89	17.21	418.48	2.07	175.84	5.06	107.78	91.97

5. Results and discussion

5.1. Statistical analysis

Table 4 presents the descriptive statistics related to the sum of the number of instructions consumed by Apache and MySQL services, CPU, memory and disk utilization for both user profiles, and response time for Biding profile. Regarding the number of instructions and memory, the negative value of skewness is reinforced because the median is greater than the average. This characteristic is clearly observed in the histograms of the number of instructions consumed by Apache and MySQL services (Figs. 3a and 4a) through the long left tail relative the right tail. The negative skewness was primordial in the choice of distributions used to fit the number of instructions.

Table 4 also shows high kurtosis values for CPU, disk and response time. This characteristic is seen in Figs. 3a and 4a through a well pronounced peak, near the median. These metrics have many time intervals equal or close to zero. Therefore, the non-zero values promote a large scale difference contributing to the presence of peaks.

The change from browsing to bidding profile implies in a memory consumption increment. However, the disk consumption is much lower compared to memory consumption because, in general, e-commerce applications are in-memory, i.e., the information is transferred from disk to memory (cache) to avoid slow Web response times.

Figs. 3b and 4b show the number of instructions executed by the CPU concerning the Apache and MySQL services for each of the 100 user sessions performed during the experiment. In both profiles, MySQL requires more processing than Apache. Furthermore, none of these services are bottlenecks for the application in this experimental setting.

Fig. 5a depicts the scatterplot of percentage of CPU utilization over a user session, where we found a higher CPU consumption at the beginning of the session. This consumption decays rapidly to zero or close to zero and so continues until the end of the session. The concentration of data in a single well-pronounced peak near the median with fast decay reinforced the high kurtosis presented in Table 4.

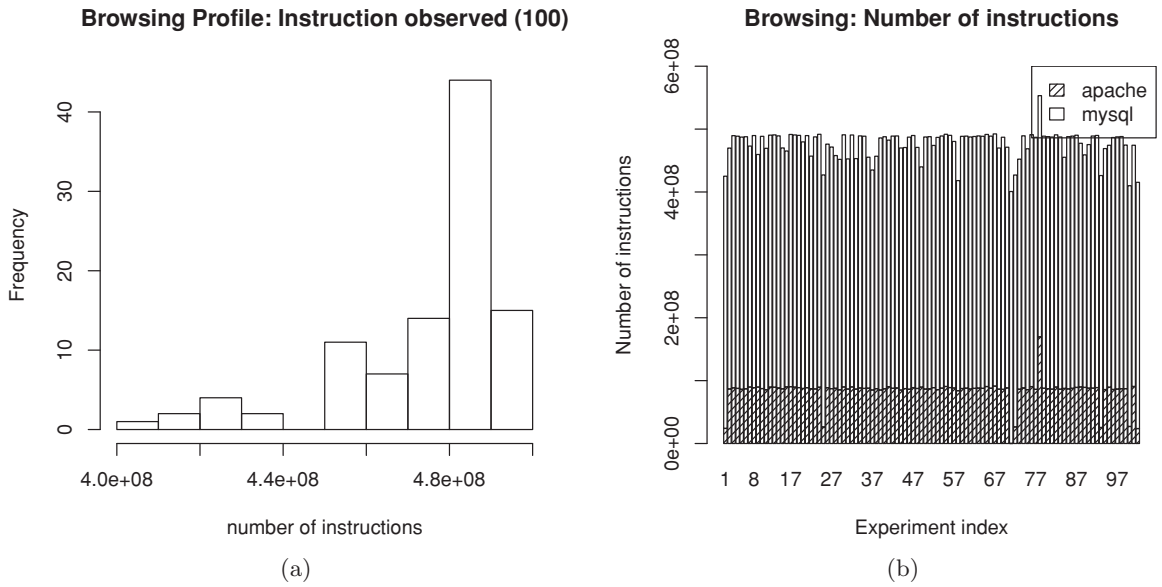

Fig. 3. Statistical analysis of number of instructions. (a) Histogram of Browsing profile (median of 100 rounds). (b) Number of instruction per user session for Apache and MySQL.

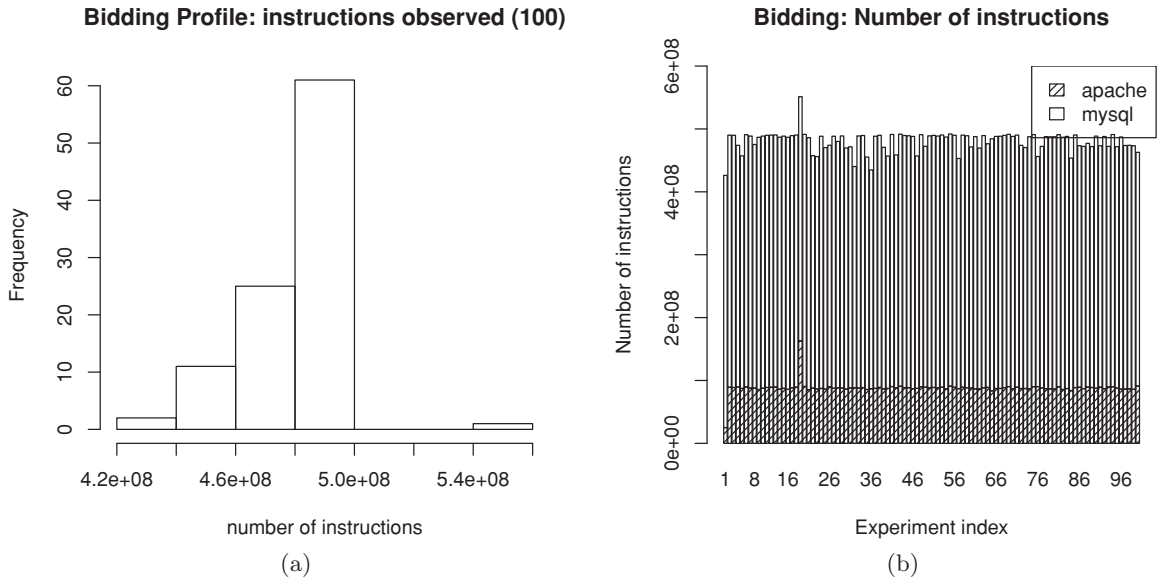

Fig. 4. Statistical analysis of instructions number. (a) Histogram of Bidding profile (median of 100 rounds). (b) Number of instruction per user session for Apache and MySQL.

Due to the large difference in scale between the percentages of CPU utilization reflected in Fig. 5a, the values of the axes are limited in order to observe the CPU utilization behavior when it is equal or close to zero, as shown in Fig. 5b. The same pattern of behavior is identified in the bidding profile.

Fig. 5b shows that instructions arrive to the processor in bursts followed by periods of inactivity, because of think times of users, for both profiles. However, the number of instructions in each cycle and the frequency with which they occur varies over time, so there is no pattern about where the peaks and troughs of cycles will happen, indicating a stationary time series. Thus, the data are subjected to a statistical hypothesis test of stationarity, where the Augmented Dickey–Fuller (ADF) [29] and Kwiatkowski–Phillips–Schmidt–Shin (KPSS) [30] tests are computed. In these tests, the significance level are fixed at $\alpha = 0.05$. Both tests showed that the data are stationary: for ADF, p-Value $= 0.01$ and, for KPSS, p-Value $= 0.1$.

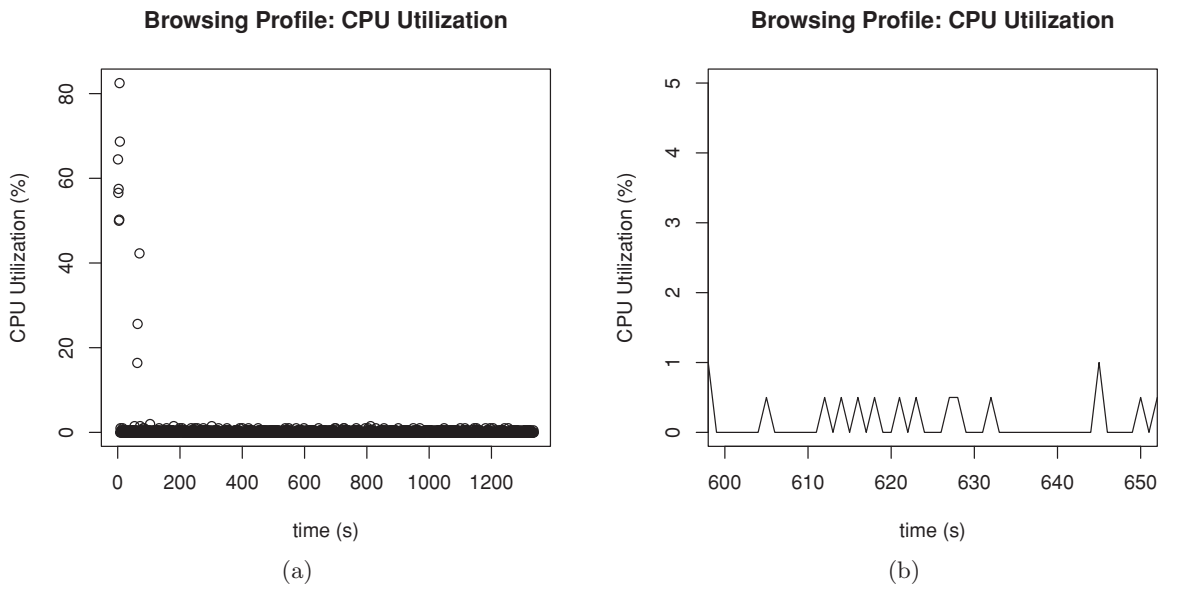

Fig. 5. Exploratory analysis of the observed CPU utilization for a browsing user session. (A) Browsing profile: scatterplot of CPU utilization. (b) Browsing profile: limited axis with interval between 600 and 650 s.

Table 5
Distributions and parameters for number of instructions.

Distribution	Estimation method	Estimated parameters	
		Browsing	Bidding
GL	mle	$\hat{\alpha_3} = -9.509e-01$, $\hat{\alpha_4} = 8.055e-01$	$\hat{\alpha_3} = -9.414e-01$, $\hat{\alpha_4} = 9.814e-01$
	hist	$\hat{\alpha_3} = -9.635e-01$, $\hat{\alpha_4} = 3.264e-02$	$\hat{\alpha_3} = -4.442e-01$, $\hat{\alpha_4} = 4.790e-01$
	quant	$\hat{\alpha_3} = -9.744e-01$, $\hat{\alpha_4} = 1.347e-02$	$\hat{\alpha_3} = -9.539e-01$, $\hat{\alpha_4} = 2.102e-02$
	mps	$\hat{\alpha_3} = -9.504e-01$, $\hat{\alpha_4} = 9.834e-01$	$\hat{\alpha_3} = -9.483e-01$, $\hat{\alpha_4} = 9.777e-01$
GEV	mle	$\hat{\xi} = -3.015e+00$, $\hat{\sigma} = 4.230e+07$	$\hat{\xi} = 7.116e-01$, $\hat{\sigma} = 6.077e+07$
	pwm	$\hat{\xi} = -1.324e+00$, $\hat{\sigma} = 1.839e+07$	$\hat{\xi} = -8.767e-01$, $\hat{\sigma} = 1.523e+07$

5.2. Parameters estimation

Table 5 shows the values of the estimated parameters for the selected distributions in combination with the different estimation methods for the number of instructions considering both user profiles. The GL distribution has four estimated parameters, because the sample is the same for all estimation methods. The median values for the browsing profile ($\hat{\mu} = 4.873e+08$) and for the bidding profile ($\hat{\mu} = 4.877e+08$), as well as inter-quartile range for the browsing profile ($\hat{\sigma} = 1.919e+07$) and for the bidding profile ($\hat{\sigma} = 1.735e+07$) remain constant for all the combinations.

In contrast, the shape parameters (α_3) and (α_4) of the GL distribution have their values influenced by the estimation method. Similarly, the value of the location parameter for the browsing profile ($\hat{\mu} = 4.783e+08$) and for the bidding profile ($\hat{\mu} = 4.820e+08$) of the GEV distribution remains constant, while the scale parameters (σ) and shape (ξ) are influenced by the estimation method. Therefore, if the selected fitting distributions have shape parameters, it is important to verify the existence of outliers in the sample in order to choose the appropriated estimation method. Fig. 6 shows how sensitive the *mle* and *pwm* estimation methods are to the presence of outliers.

Table 6 contains the parameters estimated through the *mle* method, for the distributions that represent the CPU, memory and disk utilization, and response time metrics and offer the best fit for the data, according to the KS test. The GEV distribution best fits CPU (Bidding), memory (Browsing), disk (Browsing), and response time (Bidding) enhancing the results found in Moreno et al. [3] that uses GEV to model the consumption of CPU and memory from the data provided by the Google Cloud TraceLog [6]. The other scenarios are covered by the GWD and Error(3P). All these distributions have a shape parameter that allows a better fitting.

5.3. Goodness of fit

Fig. 6 shows the Q–Q graphs for the following pairs distribution/estimation method: GL/*mle* and GEV/*pwm*. For each graph, the reference line is plotted. It can be noticed that the pair GEV/*pwm* quantiles (Fig. 6b), in terms of the observed data quantiles, have greater proximity to the reference line. Thus, this pair is a strong candidate to represent the observed number of instructions behavior. It is interesting to compare the pairs GEV/*pwm* and GL/*mle* (Fig. 6a). So, it can be observed how sensitive the *mle* estimation method is to the presence of outliers.

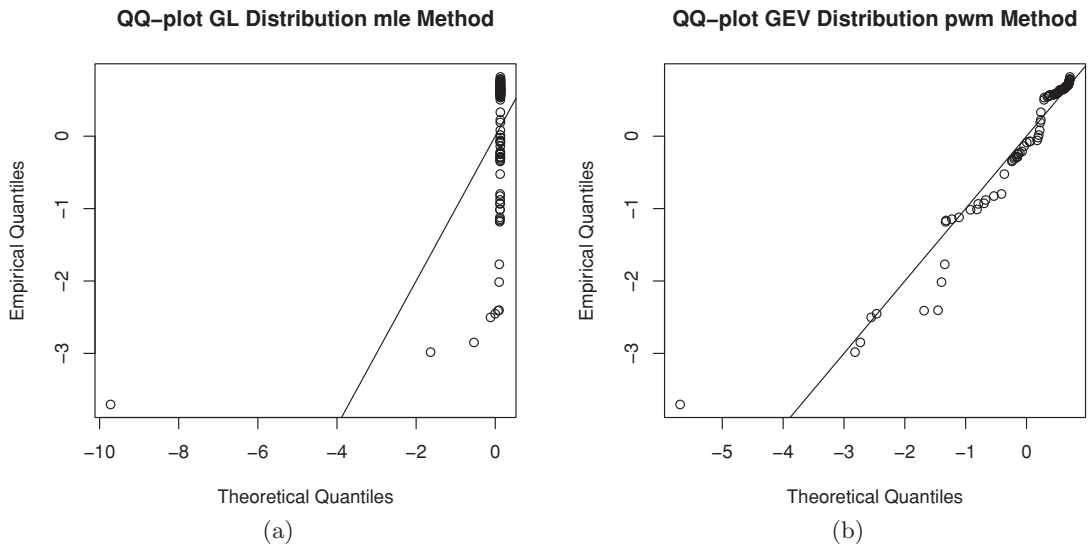

Fig. 6. Quantile–Quantile plots. (a) GL with the mle method; (b) GEV with the pwm method.

Table 6
mle parameters estimation of response time and CPU, memory and disk utilization metrics.

Metric	User profile	Distribution	Estimated parameters
CPU utilization	Browsing	GWD	$\hat{\xi} = 5.386$, $\hat{\mu} = 0.976$, $\hat{\sigma} = 0.014$
	Bidding	GEV	$\hat{\xi} = -0.259$, $\hat{\mu} = 0.576$, $\hat{\sigma} = 0.004$
Memory utilization	Browsing	GEV	$\hat{\xi} = -0.221$, $\hat{\mu} = 0.049$, $\hat{\sigma} = 8.526e-06$
	Bidding	Error(3P)	$\hat{\xi} = 2.062$, $\hat{\mu} = 7.669$, $\hat{\sigma} = 3.0985e-04$
Disk utilization	Browsing	GEV	$\hat{\xi} = -0.369$, $\hat{\mu} = 0.977$, $\hat{\sigma} = 0.003$
	Bidding	Error(3P)	$\hat{\xi} = 2.042$, $\hat{\mu} = 0.967$, $\hat{\sigma} = 0.003$
Response time	Bidding	GEV	$\hat{\xi} = -0.290$, $\hat{\mu} = 1086.7$, $\hat{\sigma} = 10.898$

Table 7
The Kolmogorov–Smirnov test: number of instructions.

Distribution/estimation method	Browsing		Bidding	
	D	*p*-Value	D	*p*-Value
GL/mle	0.130	0.368	0.120	0.431
GL/hist	0.210	0.016	0.240	0.009
GL/quant	0.230	0.012	0.240	0.006
GL/mps	0.120	0.465	0.120	0.433
GEV/mle	0.200	0.030	0.450	2.2e-16
GEV/pwm	0.100	0.675	0.160	0.146

Table 7 presents the values of D and p-Value test statistics for the number of instructions observed to the distributions of probability specified in Section 4.4.1. Considering the browsing profile, there are only 3 cases where the null hypothesis cannot be rejected because the p-Value > 0.05: GL/*mle*, GL/*mps* and GEV/*pwm*. Considering the profile bidding, there are 2 cases where the null hypothesis cannot be rejected: GL/*mle* and GL/*mps*. Other important information that can be inferred from the table is that the results are sensitive to the applied estimation method. The GL distribution with *hist* and *quant* estimation methods presents performance near or below to the symmetric and positive asymmetric distributions.

The estimation methods *pwm* and *mps* perform better than the *mle* method for both GEV and GL distributions, for both profiles. This can be justified by the fact that the *mle* method is equivalent to maximizing the geometric mean. Therefore, it is highly sensitive to outliers. Fig. 6a shows the presence of outliers in the data. Furthermore, the *mle* method provides good results for a small sample size, while the *pwm* and *mps* are more robust methods.

The GEV distribution with parameters estimated using the *pwm* method presents the best fit to browsing profile. In contrast, the GL distribution with parameters estimated using the *mps* method presents the best fit for the bidding profile.

In accordance with the presented results, the models are defined representing the number of instructions executed during a user session for both user profiles. These models aim to characterize user session based on the number of instructions instead of session runtime. Thus, the session runtime will vary according to the characteristics of the CPU. On the other hand, if the

Table 8
Kolmogorov-Smirnov test: response time and cpu, memory and disk utilization.

Metric	Browsing			Bidding		
	p-Value (max)	*p*-Value (min)	Error (%)	*p*-Value (max)	*p*-Value (min)	Error (%)
CPU utilization	0.901	0.007	6	0.982	0.008	1
Memory utilization	0.998	0.065	0	0.992	0.018	2
Disk utilization	0.994	0.032	1	0.994	0.044	1
Response time	–	–	–	0.983	0.036	1

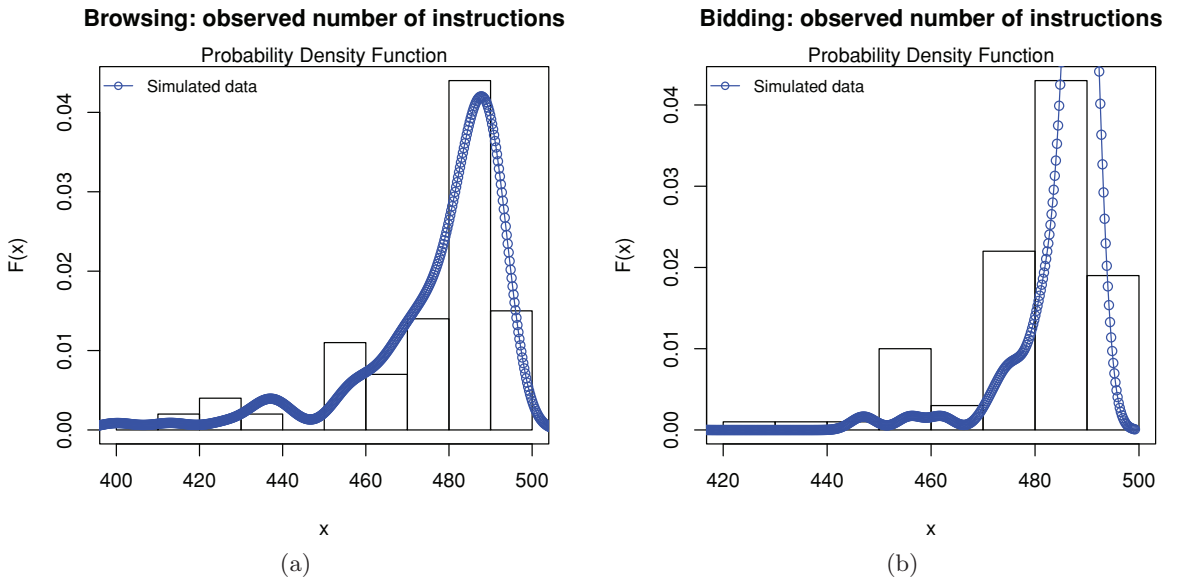

Fig. 7. Comparison of histogram of observed number of instructions and probability density function of simulated number of instructions. (a) Browsing profile. (b) Bidding profile.

model characterized the session runtime, this value would be constant regardless the processor. Also, establishing relationship between session runtime for different processors in a heterogeneous environment is a complex task, since several factors impact the runtime such as number of cores, clock frequency, and architecture.

The GoF tests defined the distribution/parameters pairs more apt to represent the observed metrics. These pairs are used to compose the model that represents the resources demand of the web application according to user profiles. Table 8 shows the *p*-Value(max), *p*-Value(min) and error(%) calculated based on 500 replications of KS test. An error is computed when sampled simulated data does not belong to the same sample data observed, i.e., *p*-Value < 0.05. No resource has an error greater than 6%, indicating that the models can correctly represent the collected data.

Therefore, the Browsing model is represented by the GEV distribution with $\hat{\xi} = -1.324e+00$, $\hat{\mu} = 4.873e+08$, $\hat{\sigma} = 1.839e+07$ parameters, representing the total number of instructions that will compose the user session, and distribution GWD with $\hat{\xi} = 5.386$, $\hat{\mu} = -0.976$, $\hat{\sigma} = 0.014$ parameters, representing how the total of instructions will be distributed throughout the session based on CPU utilization.

The Bidding model, on the other hand, is modeled by the GL distribution with $\hat{\mu} = 4.877e+08$, $\hat{\sigma} = 1.735e+07$, $\hat{\alpha_3} = -9.483e-01$, $\hat{\alpha_4} = 9.777e-01$ parameters, representing the total number of instructions, and the distribution GEV with $\hat{\xi} = -0.259$, $\hat{\mu} = 0.576$, $\hat{\sigma} = 0.004$ parameters, representing the CPU utilization. The models that represents the disk and memory demands of the two profiles and the response time experienced by the Bidding profile are presented in Table 6.

Due to trace or model unavailability, unrealistic assumptions are made in the literature about the workload [31], such as set of requisitions with fixed inter arrival times, simple Poisson models to represent instruction arrivals and exponentially distributed session time. However, in this work, we achieved different results from assumptions commonly made, such as: stationarity in the data observed, the distribution representing the arrival of instructions in the processor is GWD, and the distributions that represent the total number of instructions, which is a metric correlated with session time, are GEV and GL.

5.4. Simulation validation

The graphical validation is shown in Fig. 7, while the WMW test results are reported in Table 9. Fig. 7 shows a comparison of the histogram of observed data against probability density function of the simulated data to the number of instructions metric,

Table 9
WMW test

Metric	Browsing			Bidding		
	p-Value (max)	*p*-Value (min)	Error (%)	*p*-Value (max)	*p*-Value (min)	Error (%)
Number of instructions	0.999	0.033	3.000	0.995	0.012	9.000
CPU utilization	0.996	0.026	6.000	0.998	0.033	8.000
Memory utilization	0.992	0.046	4.000	0.999	0.016	1.000
Disk utilization	0.998	0.034	1.000	0.999	0.029	8.000
Response time	–	–	–	0.998	0.024	4.000

considering both user profiles. The simulated data are consistent with the observed data for both profiles. However, visually, the Browsing profile provides a better approximation to the observed data. This result is reinforced by Table 9, where the Browsing profile has an error three times smaller than the Bidding profile.

Table 9 shows the maximum and minimum *p*-Values obtained through the WMW test applied to a set of 100 samples of observed data and 100 samples of simulated data. For the total of 10,000 comparisons, the error was also calculated. The error is computed when sampled simulated data does not belong to the same sample data observed, i.e., p-Value < 0.05. The error for all metrics is less than 10%. Among the mathematical transformations applied to the CPU utilization metric, the inverse transformation offers the best results, reducing the Wilcox error rate of 34% to 6%, considering the Browsing profile.

Then, we can conclude the implementation of web application modeling in the CloudSim simulator is capable of producing data to accurately represent both user profiles. Thus, it can be used by researchers to build performance models and to produce tracelogs based on realistic scenarios and extrapolating the results with controlled modification of parameters such as number of users, software stack, and physical and virtual machine configuration. Furthermore, this implementation contributes to the development of performance models to support emerging cloud computing research domains, such as resource allocation in Mobile Cloud Computing (MCC) in which the trade-off between time and energy is a management challenge [32].

6. Conclusion

We applied a well-defined methodology to generate a Web application model for a cloud data center workload. It contains steps and justifications to achieve the distributions and parameters derived from application analyses and it can be extrapolated to other workload categories. Thereby, our model can be easily reproduced by researchers and cloud providers to support different research domains. It was implemented as an extension of the CloudSim simulator and its validation demonstrated that the Web application modeling can produce data to accurately represent different user profiles.

Based on our model and experiments, the following observations can be highlighted: (i) the user profile type (i.e., model of the user behavior) has a strong influence on resource utilization, so we need different statistical distributions to represent the total number of instructions and CPU, memory and disk utilization. Therefore, user behavior must be considered in workload modeling to reflect realistic conditions; and (ii) we observe the presence of stationarity instead of fixed arrival times to represent instructions arrival on the processor, GWD and GEV distributions instead of simple Poisson models to represent instruction arrivals, and Generalized Extreme Value and Generalized Lambda distributions instead of Exponential distribution to represent session time.

As future work, we are planning to (i) incorporate a model of user arrival including daily cycle characteristic; (ii) evaluate the impact of different sizes of user population on the observed metrics; and (iii) develop provisioning policies based on the proposed model to meet the web applications demand.

Acknowledgments

The authors thank Nikolay Grozev, Ph.D. candidate, for his valuable suggestions on the manuscript. Deborah M.V. Magalhães thanks the financial support from CAPES (Ph.D. scholarship) and CNPq (Doctorate Sandwich Abroad – SWE). This research was funded by the Australian Research Council through Future Fellowship program. This is also a partial result of the National Institute of Science and Technology – Medicine Assisted by Scientific Computing (INCT-MACC) and the SLA4Cloud project (STIC-AmSud program, CAPES process: 23038.010147/2013-17).

References

[1] Calheiros RN, Ranjan R, Beloglazov A, De Rose CAF, Buyya R. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw: Pract Exp 2011;41:23–50.
[2] Feitelson DG. Workload modeling for computer systems performance evaluation. Cambridge University Press; 2015. In press, available online at http://www.cs.huji.ac.il/~feit/wlmod/wlmod.pdf.
[3] Moreno I, Garraghan P, Townend P, Xu J. An approach for characterizing workloads in Google cloud to derive realistic resource utilization models. In: Proceedings of 7th international symposium on service oriented system engineering (SOSE). IEEE; 2013. p. 49–60.
[4] Chen Y, Ganapathi AS, Griffith R, Katz RH. Towards understanding cloud performance tradeoffs using statistical workload analysis and replay. Technical Report. University of California at Berkeley; 2010. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-81.html.
[5] Mulia WD, Sehgal N, Sohoni S, Acken JM, Stanberry CL, Fritz DJ. Cloud workload characterization. IETE Tech Rev 2013;30:382–97.

[6] Reiss C, Wilkes J, Hellerstein JL. Google cluster-usage traces: format + schema. Technical Report. Google Inc.; 2011. URL: http://code.google.com/p/googleclusterdata/wiki/TraceVersion2.
[7] Kavulya S, Tan J, Gandhi R, Narasimhan P. An analysis of traces from a production MapReduce cluster. In: Proceedings of 10th IEEE/ACM international conference on cluster, cloud and grid computing (CCGrid); 2010. p. 94–103.
[8] Ganapathi A, Chen Y, Fox A, Katz R, Patterson D. Statistics-driven workload modeling for the cloud. In: Proceedings of international conference on the data engineering workshops. IEEE; 2010. p. 87–92.
[9] Grozev N, Buyya R. Performance modelling and simulation of three-tier applications in cloud and multi-cloud environments. Comput J 2013. doi:10.1093/comjnl/bxt107.
[10] Rubis: Rice university bidding system. 2013. URL: http://rubis.ow2.org/.
[11] Tchana A, Dillenseger B, De Palma N, Etchevers X, Vincent J, Salmi N, et al. A self-scalable and auto-regulated request injection benchmarking tool for automatic saturation detection. IEEE Trans Cloud Comput 2014. doi:10.1109/TCC.2014.2321169.
[12] Openstack cloud software. 2013. URL: http://www.openstack.org/.
[13] Paxson V. Bro: a system for detecting network intruders in real-time. Comput Netw: Int J Comput Telecommun 1999;31:2435–63.
[14] Hashemian R, Krishnamurthy D, Arlitt M. Web workload generation challenges – an empirical investigation. Softw: Pract Exp 2012;42:629–47.
[15] Tcpdump/libpcap public repository. 2015. URL: http://www.tcpdump.org/.
[16] Menascé DA. TPC-W: a benchmark for e-commerce. Internet Comput 2002;6:83–7.
[17] Menascé DA, Almeida VA, Riedi R, Ribeiro F, Fonseca R, Meira W. A hierarchical and multiscale approach to analyze e-business workloads. Perform Evaluation 2003;54:33–57. http://dx.doi.org/10.1016/S0166-5316(02)00228-6.
[18] Prasad R, Dovrolis C. Measuring the congestion responsiveness of internet traffic. In: Passive and active network measurement. In: Volume 4427 of Lecture notes in computer science. Springer; 2007. p. 176–85. doi:10.1007/978-3-540-71617-4_18.
[19] Hastings C, Mosteller F, Tukey J, Winsor C. Low moments for small samples: a comparative study of statistics. Annal Math Stat 1947;18:413–26.
[20] Jenkinson AF. The frequency distribution of the annual maximum (or minimum) values of meteorological elements. Q J Royal Meteorol Soc 1955;87:158–71.
[21] Pham H, Lai C. On recent generalizations of the weibull distribution. IEEE Trans Reliab, 2007;56:454–8. doi:10.1109/TR.2007.903352.
[22] Mineo AM, Ruggieri M. A software tool for the exponential power distribution: the normalp package. J Stat Softw 2005;12:1–24.
[23] Chalabi Y, Würtz D, Troyer M. New directions in statistical distributions, parametric modeling and portfolio selection. ETH; 2012.
[24] Greenwood JA, Landwehr JM, Matalas NC, Wallis JR. Probability weighted moments: definition and relation to parameters of several distributions expressable in inverse form. Water Resour Res 1979;15:1049–54.
[25] Cheng RCH, Amin NAK. Estimating parameters in continuous univariate distributions with a shifted origin. J Royal Stat Soc 1983;45:394–403.
[26] Sekhon JS. Multivariate and propensity score matching software with automated balance optimization: the matching package for r. J Stat Softw 2011;42:1–52.
[27] TeamR.C.. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing; Vienna, Austria; 2013. http://www.R-project.org.
[28] Gold A. Understanding the mann-whitney test. J Prop Tax Assess Adm 2007;4:55.
[29] Said SE, Dickey DA. Testing for unit roots in autoregressive-moving average models of unknown order. Biometrika 1984;71:599–607.
[30] Kwiatkowski D, Phillips PC, Schmidt P, Shin Y. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? J Econom 1992;54:159–78.
[31] Li H. Realistic workload modeling and its performance impacts in large-scale escience grids. IEEE Trans Parallel Distributed Syst 2010;21:480–93.
[32] Ghasemi-Falavarjani S, Nematbakhsh M, Ghahfarokhi BS. Context-aware multi-objective resource allocation in mobile cloud. Comput Elect Eng 2015;44:218–40.

Deborah Magalhães is a Ph.D. student in the Department of Teleinformatics Engineering at the Federal University of Ceará, Brazil, and a previously visitor student in the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, The University of Melbourne, Australia. Her research interests include energy-efficiency provisioning, distributed systems simulation and workload modeling and characterization.

Rodrigo N. Calheiros is a Research Fellow in the Department of Computing and Information Systems, The University of Melbourne, Australia. His research interests also include virtualization, grid computing, and simulation and emulation of distributed systems.

Rajkumar Buyya is a Fellow of IEEE, Professor of Computer Science and Software Engineering; Future Fellow of the Australian Research Council; and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is also the founding CEO of Manjrasoft, a spin-off company of the University, commercializing its innovations in cloud computing.

Danielo G. Gomes is currently an assistant professor at Universidade Federal do Ceará, Brazil. He received his Ph.D. in Réseaux et Télécoms from the University of Evry, France. His research interests include Internet of Things, green computing, performance evaluation, mobile cloud computing, integration Cloud-IoT. Danielo is an editorial board member of Computers and Electrical Engineering, Computer Communications and Sustainable Computing.