Chapter

A Resource Allocation Model Driven through QoC for Distributed Systems

André Luiz Tinassi D'Amato and Wellington Oliveira de Andrade

Abstract

The trend of fog computing has generated challenges to establish resource allocation provided by this type of environment, since, in fog environments, the computing resource setting occurs on demand and at the edge of the network. Thus, ensuring both environment performance and providing user satisfaction imposes a severe technical problem. Since distributed systems are context-aware systems, the quality of context design can be applied to manage customer service, which aims to improve QoS, and provides system performance, for a given context. So, in this chapter, we propose a model to obtain runtime improvement for individual users and improve the global system performance using the quality of context in fog computing environment. The contribution of this proposal is to provide a resource allocation model, and metrics, based on QoC to deal with different distributed computing scenarios, in order to coordinate and enhance the environmental performance and user satisfaction. Experimental results show that our model improves system performance and users' satisfaction. For measuring workloads, estimates of users' satisfaction were performed. The proposed model obtained average results between 80 and 100% of users' satisfaction acceptance, and a standard deviation adherent to a flat surface for workloads with a large number of tasks.

Keywords: distributed system, fog computing, resource allocation, quality of experience, throughput, quality of context, users satisfaction

1. Introduction

The fog computing approach is an alternative to the cloud computing solution, once this paradigm reduces the amount of transmitted data on the network and the computational complexity required in the cloud. However, some approaches in the computing field try to take advantage of both approaches simultaneously. The degree of freedom presented by this new branch focuses mainly on the internet of things landscape, which needs an infrastructure that encompasses all its requirements, a situation in which fog computing fits, which allows the main focus on decision-making and data management locally [1, 2].

In fog computing, part of the data processing, which would be sent to a cloud, can take place between nearby personal devices situated at the edge network. Thus the latency problem can be mitigated, as part of the processing takes place close to the users' devices. In the fog computing model, edge devices could be set as small local data centers supporting multi-tenancy and [3] elasticity. Therefore, we can say that fog computing allows reducing the amount of data sent to the cloud, and consequently reducing the communication latency and the amount of data processed by it. Although fog computing is a good solution for dealing with the problems arising from cloud computing, this paradigm presents several challenges.

Fog computing is a solution designed to deal mainly with Internet of Things applications (IoT) [3], and this type of application tends to deal with the processing of information collected from one or more sources in real-time. From there, it is necessary to make decisions to satisfy the users' needs [3] while maintaining QoS and consequently QoE. However, relying exclusively on edge resources is not always possible, as some computing and data storage requirements may exceed the capacity of those of edge devices. In addition, a user resource configuration may not have enough capacity to meet user's request due to availability or even memory and processing limitations.

In addition, the technological diversity of edge computing devices, and the growth in user demand, generate difficulties to establish resource allocation in order to favor the environment and the applications individual. Edge devices impose a very high level of heterogeneity, making it difficult to allocate resources and establish technologies capable of dealing with different types of different devices. When performing an allocation of resources in any data center, it is important to meet the demands of the user; however, it is of fundamental importance to perform this task maintaining as much load balancing as possible so that the resources can be shared by other users. Therefore, resource scheduling in a fog environment must deal with the best fit between QoE and load balancing.

The related works considered in this article aim to establish techniques, to deal with computational resources management in distributed systems, focusing on system performance or user satisfaction. Due to the difficulty in establishing the trade-off between performance/satisfaction, the related works tried to focus on one of these parameters. Among the related works, knowledge models based on artificial intelligence and ontology are applied. Our proposal presents an approach to address this gap, considering the performance/satisfaction trade-off by developing and applying parameters of context and quality of experience. In this sense, this chapter is proposed a Quality of Context (QoC) based approach aiming at the user's QoE considered from jobs attendance time (makespan).

1.1 Paper organization

The remainder of this paper is organized as follows: Section 2 addresses the basics concepts used by the proposed model in Section 3, and 4; Section 5 discusses the experiment conducted and presents results; Section 6 addresses related works, while final considerations are presented in Section 7.

2. The relationship between quality of context and quality of experience

Considering computational or network resources, the performance perspective between humans and service providers is technically distinct. Service providers consider QoS parameters while, for users, the quality of experience determines the perceived performance [4]. The QoS metrics, for example, are determined from technical parameters: throughput; delay; network performance; loss packets rate, etc.

To evaluate the user QoE of a particular service provided by the network, opinion tests are applied in controlled environments. This type of test is known by the community as a *mean opinion score*. This technique is generally applied for evaluating multimedia systems [5]. However, due to the large number and diversity of applications, opinion tests are not the best alternative. In addition, opinion tests are criticized by some authors [6] and their criticisms are related to scoring scales used in opinion tests. The scale of opinion tests is considered by some authors such as [7] inaccurate and not representative. This occurs because scales used in MOS tests do not consider cultural differences in interpretation. According to [8], the MOS test scores determine absolute values obtained in controlled environments, which do not accurately represent real environments by not considering the influence of context variables.

According to [9], QoC describes context metrics, which can be applied to enhance application or service performance. Thus the QoC is used to establish the reliability of provided services. QoC modeling based on context parameters makes it possible to quantify, or predict, the quality of a service provided.

3. Proposed model

This Section discusses the proposed model addressing all proposed model components. First will address basic concepts of the model. Section 3.1 addresses the QoC metric and Section 4 addresses the QoE metric used in this work. In our work, it is proposed that the QoE is considered a utility function determined from QoC correctness parameter.

The model proposed in this work approaches QoE based on the concepts of resource utility when an application is submitted to a distributed system. The concepts of QoE applied, and the relationship between QoE and utility, are explained in Section 4. Is important to mention that application classes have a strong relationship with the concept of utility since each class has different needs regarding which resources they use on a larger scale. In this sense, the main information that a user describes refers to the application class. The classes referring to the application features are presented as follows: serial applications; parallel applications; network-oriented applications; CPU-oriented applications; I/O and storage-oriented applications.

3.1 QoC parameters

3.1.1 Correctness

In our work, the QoC is used to verify how much a given context is in accordance with what is being demanded by the application. As mentioned earlier in the introductory section of this chapter, the model proposed in this thesis meets different classes (or categories) of applications, each class is characterized by a different need for resources. Therefore, the parameter *correctness*, or correctness, provides a quantitative reference to determine if the current context of a particular resource is suitable to serve an application taking into account its category.

To obtain the value of QoC, an approach based on Bayesian probability theory is proposed. This approach is inspired and adapted from the solution proposed by [10]. The Bayesian probability combines information in a way that relates observed events to hypotheses. In other words, Bayes' theorem is used to calculate the probability conditional that a given event occurs given an observation. For example, calculation of the probability of overhead in allocating an amount cp_i of processes in a given node of a computational grid, in which an amount cp_j of processes available in that node was observed.

Proposed approach, aims to calculate the QoC using conditional probability from the contextual information specified in each submission of *job*, and from the status system update. A *job* is understood as a set of specifications for the execution of a certain task, these specifications being conditional on each other. For example, if context information reveals that the system has processes available in order to provide a processing rate x to execute a *job1*, it is also necessary to know the exact number of processes available to meet the application's execution flow. In other words, if the context manager determines the availability of a certain CPU rate for processing, it is also necessary to know if there are processes available to fulfill the request of *job1*.

In addition to the application demand, the approach proposed in this work aims to manage the workload exerted on the system. Therefore, one of the main functions delegated to the context manager is to collaborate for context-driven load balancing. Eq. (1) determines the probability that a given demand for a resource will be met without generating overload. Eq. (1) expresses the probability that a given demand cp_i for a resource i will overload a capacity node cp_j. Thus, the probability of the resource executing properly is determined by the completeness of Eq. (1). Therefore, the QoC for the resource i considering the available capacity cp_j is given by Eq. (2). The QoC defined in Eq. (2) is actually the partial correctness, considering only an amount of resource of type i on a single node j, named as $QoC_{i|j}$ just to facilitate understanding. For partial correctness, calculation $QoC_{i|j}$ is not considered the application class; therefore, the context for all resources described in *job* are calculated in the same way.

$$P\left(cp_i|cp_j\right) = \frac{P\left(cp_j|cp_i\right) * P(cp_i)}{P\left(cp_j\right)} \tag{1}$$

$$QoC_{(i|j)} = 1 - P\left(cp_i|cp_j\right) \tag{2}$$

3.1.2 Probability for each required resource

The probability for each required resource CP_i is given in Eq. (3), where $ENV_{resource}$ is the total capacity of the environment to provide the required resource. The $ENV_{resource}$ is given by Eq. (4), where n is the number of provider nodes in distributed resource facilities.

$$P(cp_i) = \frac{cp_i}{ENV_{resource}} \tag{3}$$

$$ENV_{resource} = \sum_{j=0}^{n} CP_j \tag{4}$$

The probability for each resource CP_j is given in Eq. (5), where $AVR_{resource}$ is the average of total capacity of the environment to provides the required resource. The $AVR_{resource}$ is given by Eq. (6), where n is the number of provider nodes in distributed resource facilities.

$$P\left(cp_j\right) = \frac{cp_j}{AVR_{resource}} \tag{5}$$

$$AVR_{resource} = \frac{\sum_{j=0}^{n} CP_j}{n} \tag{6}$$

3.1.3 Conditional probability

The conditional probability that associates the required and the available amount of resource is defined in Eq. (7). The probability of a required resource cp_i given an amount cp_j is defined by the magnitude correlation between what was requested and what is available.

$$P\left(cp_j|cp_i\right) = \frac{cp_i}{cp_j} \tag{7}$$

3.1.4 Conditional context parameter

Eq. (1) determines context evaluation from Dependent Context Parameters (DCP) set. Thus, is possible to achieve Eq. (8), where n is the applications context parameters (cp_i). The set CCP Conditional Context Parameters resulting from the Eq. (8).

$$CCP(cp_i) = \left\{P\left(cp_i|cp_j\right); j = 1..n \cap cp_j \varepsilon PDC_i\right\} \tag{8}$$

The correctness is calculated correlating CP_j, CP_i, and n for all context parameters that characterize cp_i. The $QoC(cp_i)_k$ is context association that characterizes cp_i for k-th CP. The Eq. (1) relates the application requirements with resources available resulting in Eq. (9) resulting in Eq. (15).

$$QoC_{(i|j)} = 1 - \frac{P\left(cp_j|cp_i\right) * P(cp_i)}{P\left(cp_j\right)} \tag{9}$$

The resulting correctines, Eq. (10), is obtained from resulting QoC.

$$Correctiness = QoC_{(X|Y)} \tag{10}$$

$$QoC_{(X|Y)} = 1 - \frac{P(X|Y) * P(X)}{P(Y)} \tag{11}$$

$$QoC_{(X|Y)} = 1 - \frac{\frac{X}{Y} * \frac{X}{Z}}{P(Y)} \tag{12}$$

$$QoC_{(X|Y)} = 1 - \frac{\frac{X^2}{Y * Z}}{\frac{Y}{\frac{Z}{N}}} \tag{13}$$

$$QoC_{(X|Y)} = 1 - \frac{X^{2*}}{Y^2}\frac{1}{N} \tag{14}$$

Thus, the correctiness parameter for X amount of computation requested is defined by Eq. (15).

$$Correctiness = 1 - \left(\frac{1}{N}\right) * \left(\frac{X}{Y}\right)^2 \tag{15}$$

The resulting QoC is given by Eq. (16), for all CP_j potential associated with CP_i, and from the number of all n possible CPs, where m is the number of parameters considered QoC resource CP_i. The expression $QoC\left(cp_i\right)_k$ is the k-th conditional element that characterizes the *job* J_i.

$$QoC(J_i) = \frac{1}{m}\left(\sum_{k=1}^{m} QoC(J_i)_k\right) \tag{16}$$

From the QoC, obtained in our model, it is possible to quantitatively predict the user's experience. The relationship between QoE and QoC proposed in our work is presented in the following section.

4. Predicting the quality of experience: a quantitative approach

In our model, QoE quantitatively expresses the prediction of user's satisfaction from QoC-utility function not depending on subjective feelings. The reason is that quantitative metrics are not only more meaningful, but also provide an improved magnitude reference of the measured parameters. From this magnitude reference is possible to benefit the user, and the resource provider environment. In our work, the correctness and runtime are proposed as quantitative metrics for prediction.

In a fog environment, or any distributed system is acceptable that computing resources are provided on demand. The matching between available resources and application requirements is expressed by utility function. The utility function is expressed by the Eq. (17). Utility $U(p)_r$ assigned to a particular resource r, represents a metric proportion of resources adequacy for application, also given by p (*probability of correctness*). Correctness is the main metric applied to measuring the QoC to meet applications demands. The utility is given by Eq. (17).

$$U_r = correctiness_r \tag{17}$$

The QoE directly depends on application runtime rt, and the response time t. Response time is given t by sum both execution time and waiting time. So, QoE is expressed in Eq. (18).

$$QoE = \frac{rt}{t}\sum_{r=0}^{R} U_r^{\alpha} \tag{18}$$

The α value quantifies the resource importance, r, to application, and his value ranging between 0 and 1. The α parameter is related with application category, thus, is determinant for utility function. The α parameter is expressed by Eq. (19).

$$\alpha = 1 - U(p) \tag{19}$$

In this section, the metric used to predict users QoE was discussed. Next section covers the obtained results from experiments conducted.

5. Experiment and results

In order to test the efficiency of proposed model, experiments were conducted using the *SimGrid* [11] simulator. This simulator was chosen due their scientific community importance. The mentioned simulator are widely used in academic works in the area of distributed systems for determining a flexible test platform, which can provide hundreds of resources to be used in several experiments. The experiments were carried out aiming to verify the proposed model behaviour considering different workloads, numbers of users and context parameters.

The experimental evaluations, section 5.2, were conducted aiming at performance and QoE. The experiment, aims to insert an exhaustive workload in the tasks submission, and discusses the QoE and the performance of the environment. The QoC was obtained in experiments from the equations proposed in the 3.1 section, using as values for variables, the computational capacity for processing and network, number of processes, workload, and communication latency. The value of Eq. (1), CP_i corresponds to the value of the resource i requested by each workload, and the value to CP_j corresponds to the capacity of the requested resource available on a given node j.

The mentioned values were extracted from the simulation in the SimGrid environment. Programming codes were inserted into the simulation in order to collect data to simulate the *Context-provider*. The QoE for each workload was calculated according to the metric discussed in the 4 section. The QoE was obtained through the resulting QoC considering the parameter *Correctness*, and from the execution and waiting times.

The objective of this experiment according is to analyze the performance of proposed model and perform QoE estimates, when the model is subjected to thousands of *jobs* of different characteristics considering a well-defined interconnection grid, in the SimGrid environment. The SimGrid simulator provides a complete simulation environment to simulate point-to-point interconnection between computers in a grid. The SimGrid simulator has better support for managing *links* when compared to others simulators. In SimGrid it is possible to manage the allocation parameters addressed (Cpu speed, number of processes, transmission rate, latency) peer-to-peer. In general, SimGrid has support for simulating data transmission over the network. In this experiment, files in mrc format were used as input to specify the set of tasks to be simulated, and as output were generated *trace* files in mrc format containing execution data and QoE and QoC estimates. In this experiment, the execution times of the *jobs* and the QoE estimates obtained by user were analyzed. In addition, the central tendency and dispersion of the data were analyzed, as well as the *outliners* generated by the proposed model.

5.1 Infrastructure and workloads

The environment that configures the simulated infrastructure determines a heterogeneous test platform, aiming to measure the performance of our model. For this, 10 grid resources were configured, each feature is determined by a computational grid with the following specifications in **Table 1**:

Resource	Machines available	Mips per process
0	200	50
1	200	50
Baud Rate 1000000: Network between resources 0 and 1		
2	200	150
3	200	150
4	100	150
Baud Rate 500000:Network between resources 2, 3, and 4		
5	100	300
6	100	300
7	100	300
Baud Rate 200000: Network between resources 5, 6, and 7		
8	50	800
9	50	800
Baud Rate 100000: Network between resources 8 and 9		

Table 1.
Simulated resources description.

The environment configuration used aims to represent a heterogeneous set of resources. This type of environment is similar to those found in fog computing environments. The jobs used in this experiment aim to establish a diversified workload, in order to explore the features of the model proposed in this work. Therefore, a number of 20 different types of jobs were used, which are described in **Table 2**.

The environment used as a platform for carrying out the tests was the platform. xml file available along with the simulator. SimGrid is an ideal testing platform for resource allocation policies involving networks, as it has simulation classes that implement point-to-point communication mechanisms. The connection between the various nodes of a computational grid can be specified in SimGrid. The proposed *trace* file records the times obtained with the execution of *jobs*, together with the estimated QoC and QoE. The context-broker receives the file *mrc* as input, which must contain the description of tasks by users, which are:

- Category: Category of the application, cpu or network oriented, sequential or parallel;

- Process_amount_req: Number of processes required by the user;

Type of task	Process required	MIPS required	Type
1	200	200000	CPU/Network bound
2	200	200000	CPU/Network bound
3	200	1000	Network bound
4	200	100	Network bound
5	100	500	Network bound
6	100	500	Network bound
7	100	500	Network bound
8	100	100000	CPU bound
9	50	800	CPU/Network bound
10	50	100000	CPU bound
11	50	100000	CPU bound
12	50	50000	CPU bound
13	20	50000	CPU bound
14	20	100	Network bound
15	20	500	Network bound
16	10	50000	CPU bound
17	10	50000	CPU bound
18	10	50000	CPU bound
19	5	5	Network bound
20	5	200	CPU/Network bound

Table 2.
Workloads description.

- Computation_amount_req: Amount of computation in FLOPS estimated by the user;
- Communication_amount_req: Amount of communication in bits estimated by the user;
- Execution_time: Estimated execution time by the user.

The workloads were synthesized by generating random values for the mentioned parameters above, in order to simulate the unpredictable behavior of a real distributed system. A workload was generated with 3000 tasks randomly distributed among 20 fictitious users represented by numbers (IDs) from 1 to 20. The tasks (or *jobs*) were synthesized using the random function of the GCC library. The system clock time was used to calculate and generate the random numbers.

5.2 Results

The absolute values obtained by running the workloads using the algorithm *Round-Robin* and proposed model context-based strategy are shown in **Tables 3** and **4**, respectively.

The smaller standard deviation represents a smaller dispersion in the resulting runtimes. The users' QoEs are shown in the graphs in **Figures 1** and **2**. The graphs present the average QoE values for each user's identifier represented in the ID_users axis. The number of users for the graphs in **Figures 1–4**, is fixed and equal to 20, with the QoE and standard deviation resulting from each user as the workload on the system increases.

The surface generated by the data referring to the average QoE of users when using the model proposed in this work. **Figure 1**, shows a global trend of QoE values around 90% of user satisfaction. This behavior is reinforced by the graph in **Figure 3**. It is

Number of tasks	Average	Standard deviation
100	31028.43	63988.94
200	35732.46	68043.18
300	35076.89	68064.07
400	36377.30	68674.47
500	37436.61	71239.62
600	38791.38	72928.67
700	38329.13	71744.08
800	38329.13	70564.11
900	37032.33	70093.15
1000	36626.69	69546.43
1500	37604.74	70032.94
2000	36255.30	69071.16
2500	36158.03	68628.92
3000	35713.19	67863.07

Table 3.
Average runtime of Round-Robin.

Number of tasks	Average	Standard deviation
100	31979.29	27697.15
200	32031.66	27756.22
300	30226.01	27179.46
400	29601.41	27145.35
500	30676.16	27611.36
600	30342.75	27571.99
700	30068.66	27620.44
800	30065.64	27616.27
900	30136.01	27558.11
1000	29872.54	27565.27
1500	30086.24	27400.14
2000	30270.34	27406.40

Number of tasks	Average	Standard deviation
2500	30507.55	27418.29
3000	30400.25	27429.05

Table 4.
Average proposed model runtime.

important to note that the values are shown in the range between 0 and 1 to represent a QoE from 0 to 100%.

The graph in **Figure 2** shows several QoE spikes revealing an imbalance in terms of user satisfaction with the Round-Robin and Dijkstra policy. Another situation observed in **Figure 2** is the instability of the Round-Robin and Dijkstra policy for QoE. According to the graph in **Figure 2**, users tend to be more dissatisfied when the task set increases. According to **Figure 2**, for a workload with a size of more than 1500 tasks, the users' QoE starts to decrease more sharply.

The graph in **Figure 3**, represents the estimated QoE standard deviation. The graph shows that as the number of tasks submitted to the system increases, the standard deviation becomes more linear. The graph in **Figure 3** reveals a more uniform surface for workloads with a quantity above 1000 tasks. This is because users' QoE tend to decrease with variability in resource states (totally free, fully occupied), especially when there are few tasks. For large numbers of tasks, resource variability occurs more "smoothly" and distributed over time.

The fairness generated by adequate load balancing, which respects the users' needs, is a determining factor in the overall QoE experience. This situation is shown in the graphs of **Figure 1**. The graph presents the average QoE values for each user's identifier represented on the Users_ID axis. The number of user's for the graphs in **Figures 1** and **3** is fixed and equal to 20, with the QoE and the resulting standard deviation of each user being shown as the workload on the system increases. The surface generated by the data referring to the average user's QoE, **Figure 1**, shows a

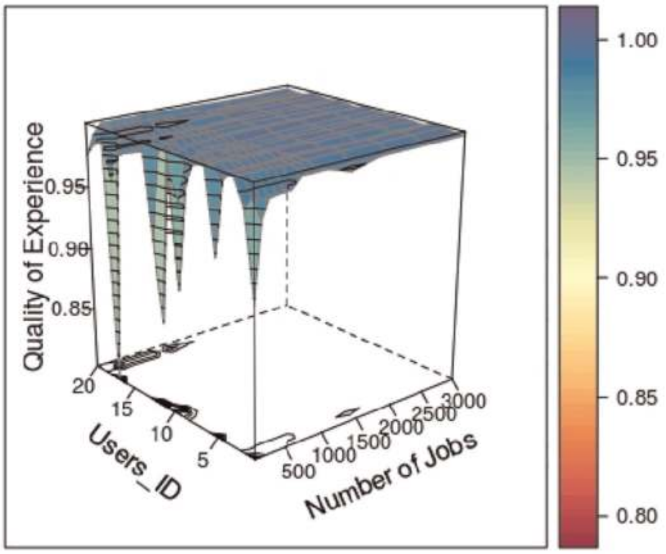

Figure 1.
QoE per user from proposed model.

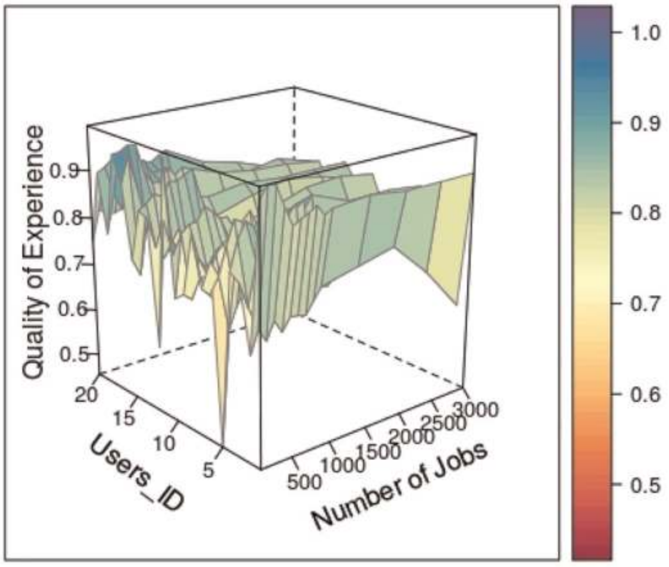

Figure 2.
QoE per user Round Robin and Dijkstra.

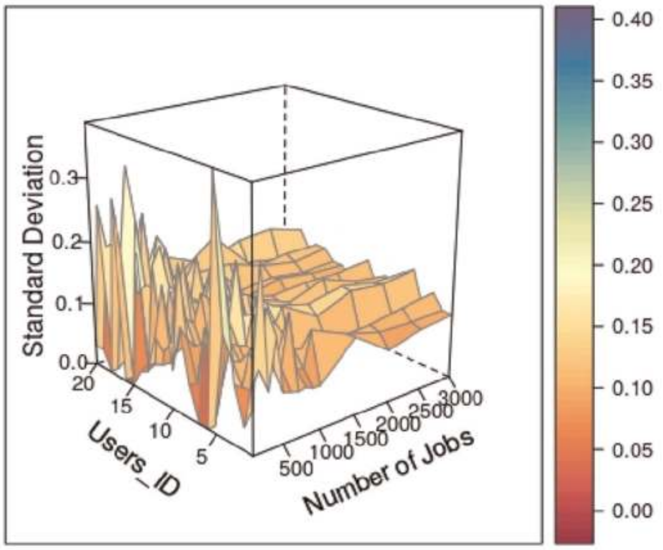

Figure 3.
Standard deviation for QoE per user from proposed model.

global trend of QoE values around 90% of user satisfaction. This behavior is reinforced in the graph of **Figure 3**. It is important to note that the values are shown between 0 and 1 to represent a QoE between 0 and 100%.

The graph in **Figure 4** displays the standard deviation of the estimated QoE using Round-Robin policy with Dijkstra. The graph reveals a greater variation in users' QoE when compared to our proposal. The standard deviation surface of the graph in **Figure 4**, presents irregular characteristics for the QoE of all users. For task sets of sizes 100, 200, and 300, the standard deviations are low, however, the values for QoE for these task sets have lower QoE for all users.

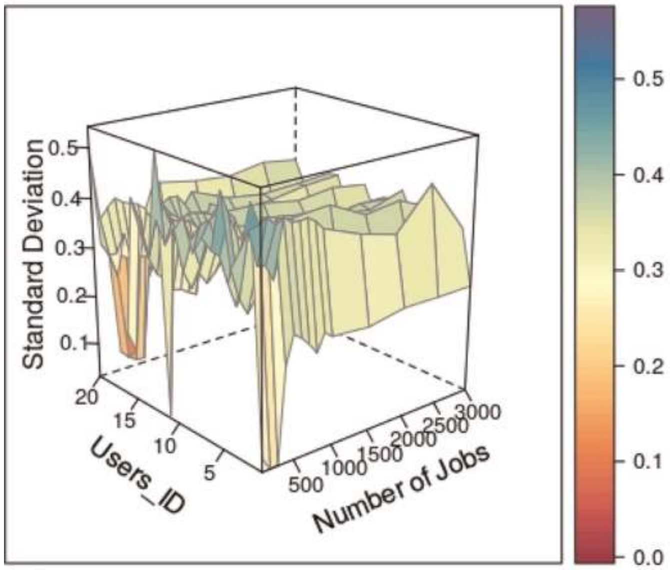

Figure 4.
Standard deviation for QoE per user from Round Robin and Dijkstra.

The proposed model presented better results and stability for QoE and standard deviation when compared to the Round-Robin and Dijkstra based strategy, which does not consider the context to provide resource allocation.

Figure 5 shows the number of tasks that were submitted by each user. The graphs in **Figures 6** and 7 shows the data dispersion in relation to the proposed model QoE. The graphs of **Figures 6** and 7, reveal that although the model is efficient in a global perspective, it is subject to undesirable *outliners* generated by unsatisfactory QoE. The box plot in **Figure 6** confirms the trend of user satisfaction between 80 and 100%, but reveals the cases in which these values did not occur. The same happens for the graph of **Figure** 7. The graph of **Figure** 7 shows the values of the QoEs obtained by each user. According to the graph, it is possible to observe that values close to 100% and close to 0% are not concentrated in specific users, but distributed among all users. Thus, it is possible to carry out a qualitative analysis, concluding that the model established (*fairness*) to resource allocation.

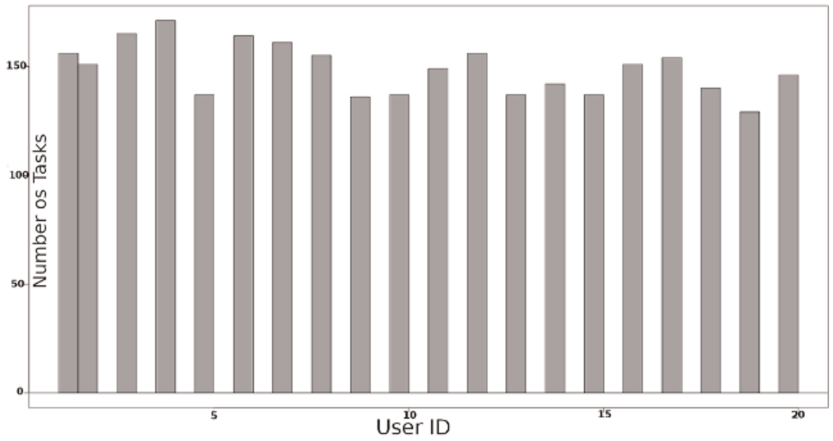

Figure 5.
Number of tasks submitted per user.

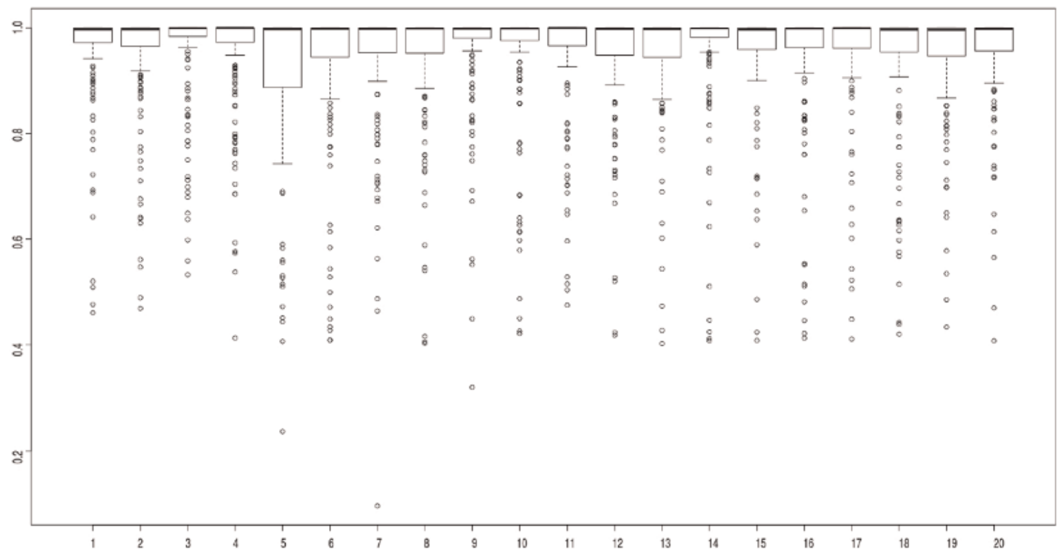

Figure 6.
Box Chart for Users QoE.

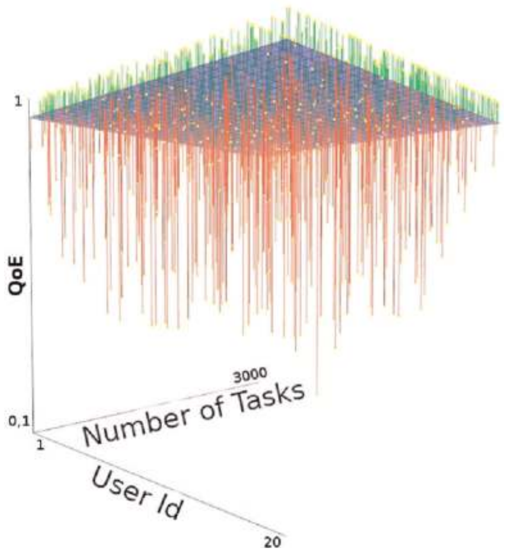

Figure 7.
Scatterplot for (QoE x amount of tasks) per user.

The objective of this experiment was to analyze the relationship between performance and QoE established by proposed model, when the model is submitted to thousands of *jobs* of different characteristics considering a well-defined interconnection grid, in the SimGrid environment. From the results obtained with experiment, it was concluded that the model proposed in this paper has better performance when compared to the *Round-Robin* algorithm with the routing based on the Dijikstra algorithm. For the workloads submitted to execution, QoE estimates were performed. The proposed model obtained average results between 80 and 100% of satisfaction for each user, and a standard deviation adherent to a flat surface for workloads with a large amount of tasks. In this way, the proposed model showed stability in terms of the QoE obtained. The results obtained also show that the proposed model established fair behavior, *fairness*, in resource allocation.

6. Related works

Messina [12] proposes an agent-based model to provide System Level Agreement negotiation. The authors use an ontology to establish the resources needed for each submission. Ontologies are used to provide knowledge to the environment. Therefore, it is possible to establish a more adjusted allocation of resources according to the application's input parameters. Ontologies generate knowledge about the semantic rules to perform the correspondence between the requested resource and the available resource. The work proposed in [12] uses an ontology to provide knowledge, however this approach does not handle situations not foreseen by the ontology well. The context-based strategy of the model proposed in this article provides the best configuration for running applications based on the updated state of the system. The approach proposed in this article performs the allocation of resources considering, in addition to the performance of the application, the performance of the system, which does not happen in the work proposed by [12].

Das uses a resource scheduling policy based on [13] artificial intelligence. However, Das uses the Teaching-Learning Based Optimization learning algorithm as a basis for his proposal. The justification given by the authors for the adoption of the learning algorithm in the context of resource allocation in computational grids is that Teaching-Learning Based Optimization is considered a light and efficient algorithm to find the global solution to optimization problems. Another work based on artificial intelligence is presented in [14]. The authors use an approach centered on estimating values for various data transmission parameters, such as latency and use of *links*. The approach used in [14] is applied only to provide service guarantees, based on QoS prediction through fuzzy logic. Parameters, or formulations, for controlling overall performance are not specified.

In [15] a dynamic resource allocation method, is proposed for load balancing in fog environments. For this, the method has a scheduler capable of performing dynamic migration of services to achieve load balancing for computing systems in fog. Negative aspects related to migration, QoS degradation, and QoE, are not considered in the [19] work. In the works [16, 17] scheduling strategies with real-time constraints are discussed. In [16], aspects of QoE and QoS are addressed but not in depth in order to propose directives to measure and improve these attributes. In [17], the authors develop a work in order to investigate how utility is affected by performance parameters in environments focused on fog aimed at healthcare applications. To evaluate the use of a fog data center, the resources of the iFogSim tool were used. In the work [18], a new resource allocation algorithm based on stable correspondence is proposed, in order to benefit users and providers in the fog environment. However, the authors do not clearly show how the aspects involving user satisfaction and the performance of the environment are treated. **Table 5** shows that our proposal establishes a model that meets QoE and makespan together. This tradeoff is not achieved in the works analyzed so far being our main contribution.

In [19] is proposed a pricing policy based on the QoE. This QoE is expressed from result of the allocation and try to optimize resource allocation from statistical information of the computational requests. This mentioned strategy is implementable in real-time brokers according to the authors. optimal dynamic allocation rule based on

Reference	Addresses a QoE model	Makespan	How implements resource allocation policy
[12]	Yes. SLA model	No	Knowledge model from Ontologies
[13]	No	Yes	Teaching-Learning Based Optimization
[14]	No. QoS only	No	Fuzzy logic to provide QoS
[15]	No	Yes	Dynamic Load Balancing
[16]	Yes. Aimed to Real-Time tasks	No	Neural Networks and Fuzzy Logic
[17]	No	Yes. Based on utility	Algorithm based on stable correspondence
[18]	In a non-detailed indirect way	Yes	Cost Load-Balancing Strategy
[19]	Yes	No	Statistical and real-time approach
[20]	No	Yes	Energy-Aware Load-Balancing Strategy
Our Model	Yes	Yes	QoE/QoC Tradeoff Model

Table 5.
Comparison between proposals.

the The developed solution is statistically optimal, dynamic, and implementable in real-time. The proposal in [20] is based on an energy and collaborative model in load balancing. The proposal in [19] is based on a statistical and dynamic model aiming users QoE. In [20] is proposed an algorithm to both: meeting the application latency requirements and providing energy efficiency in the heterogeneous edge tier. To the algorithm proposed in [20] implements a collaboration strategy at the edge of the network aiming at heterogeneous environment characteristics. According [20] is possible to reduces the waiting time for meeting requests. The proposal in [20] is based on an energy and collaborative model in load balancing. **Table 5** summarizes related work and shows a comparison with our proposal in terms of QoE, environment performance (makespan), and technical approaches to implementation.

7. Conclusions

The scheduling techniques to resource allocation in distributed systems do not generally meets jointly the user satisfactin and throughput. The QoE emerges as a differentiated paradigm to fill this gap. The QoE approach is particularly important for resource allocation, since the resource allocation adjusted to users needs must to consider contextual parameters. The use of QoC to make the resource allocation allows an efficient management, resulting in a performance gain achieved by a strategic load distribution, improving the level of users QoE. Aiming to act in this mentioned scenario was proposed, and evaluated through experiments, QoC-based approach to provide resource allocation in fog computing. The proposed model performs management decisions based on a QoC policy. The QoC is used also to predict the user's QoE. Our model quantifies resource feasibility considering an application demand. An experiment was conducted to analyze the performance of the proposed model. From the results obtained it was concluded, that our model shows a lower standard deviation, when compared to well-known strategies. From the use of the proposed QoC-based policy, was obtained average QoE scores between 80 and 100% considering each user. The resulting QoE values adhere to a flat surface for workloads with large amounts of tasks. Thus, our model shows a stable behavior considering obtained QoEs. Results show that the proposed model established fair behavior (fairness) in resource allocation. Our QoC-based policy stands out, especially when there are excessive workloads and a lack of resources. Among the works mentioned in related works, there are approaches to allocate resources considering the performance of the environment and user satisfaction. Although the related proposals aim to meet user demands, the authors do not provide metrics for effective measurement of user satisfaction. In addition to all that has been mentioned, the works found in the literature on resource allocation are generally applied to specific environments, which do not consider orchestrating different paradigms of distributed systems. Therefore, the main contributions of this work are a solution to the existing gap between user satisfaction and environmental performance (makespan) for distributed systems. Although this work aims to meet the needs of the system and the users, it does not guarantee the QoE individually for the users, it only proposes to improve the average satisfaction. Another limitation of the work is that although the model has been tried in specialized simulators, this model has not been implemented in a physically robust fog environment.

Abbreviations

GB	Gigabytes
GBps	Gigabytes per second
QoS	Quality of Service
QoC	Quality of Context
QoE	Quality of Experience
IoT	Internet of Things
MoS	Mean Opinion Score
DCP	Dependent Context Parameters
CCP	Conditional Context Parameters
AVR	Average
ENV	Environment
MIPS	Million Instructions Per Second
HPC	High Performance Computing

Author details

André Luiz Tinassi D'Amato*† and Wellington Oliveira de Andrade†
Universidade Tecnologica Federal do Paraná, Apucarana, Brasil

*Address all correspondence to: andredamato@utfpr.edu.br

† These authors contributed equally.

References

[1] Cheol-Ho H, Blesson V. Resource management in Fog/Edge computing: A survey on architectures, infrastructure, and algorithms. ACM Computing Surveys. 2019;**52**:1-37

[2] Jouret G et al. Cisco Delivers Vision of Fog Computing to Accelerate Value from Billions of Connected Devices. 2014. Available from: wsroom.cisco.com [Accessed: July 1, 2022]

[3] Shekhar S, Chhokra A, Sun H, Gokhale A, Dubey A, Koutsoukos X, et al. URMILA: Dynamically trading-off fog and edge resources for performance and mobility aware IoT services. Journal of Systems Architecture. 2020;**10**: 101-710

[4] Junaid S, Markus F, Denis C. Quality of Experience from user and network. Annals of Telecommunications. 2010;**65**: 47-57

[5] Fiedler M, Hossfeld T, Tran-Gia P. A generic quantitative relationship between quality of experience and quality of service. Network IEEE. 2010; **24**:36-41

[6] Katrien DM, Istvan K, Wout J, Tom D, Lieven DM, Luc M, et al. Proposed framework for evaluating quality of experience in a mobile, testbed-oriented living lab setting. Mobile Networks and Applications. 2010;**15**:378-391

[7] De Koning TCM, Veldhoven P, Knoche H, Kooij RE. Of MOS and men: Bridging the gap between objective and subjective quality measurements in mobile TV. 2007

[8] Marc S, James P, Philip K. Practical issues in subjective video quality evaluation: Human factors vs. psychophysical image quality evaluation. In: Proceedings of the 1st International Conference on Designing Interactive User Experiences for TV and Video. 2008

[9] Michael K, Iris H. Challenges in modelling and using quality of context (Qoc). In: Proceedings of the Second International Conference on Mobility Aware Technologies and Applications. Springer-Verlag; 2005

[10] Brgulja N, Kusber R, David K, Baumgarten M. Measuring the probability of correctness of contextual information in context aware systems. In: Dependable, Autonomic and Secure Computing, 2009. DASC '09. Eighth IEEE International Conference.

[11] Quinson M. SimGrid: A generic framework for large-scale distributed experiments. In: IEEE Ninth International Conference on Peer-to-Peer Computing. 2009

[12] Messina F, Pappalardo G, Santoro C, Rosaci D, Sarne GML. An agent based negotiation protocol for cloud service level agreements. In: WETICE Conference. 2014. pp. 161-166

[13] Das D, Pradhan R, Tripathy CR. Optimization of resource allocation in computational grids. Journal of Grid Computing and Applications. 2015;**6**:1-18

[14] Kolomvatsos K, Anagnostopoulos C, Marnerides AK, Ni Q, Hadjiefthymiades S, Pezaros DP. Uncertainty-driven ensemble forecasting of QoS in software defined networks. In: 2017 IEEE Symposium on Computers and Communication. 2017. pp. 908-913

[15] Xu X, Shucun F, Cai Q, Tian W, Liu W, Dou W-C, et al. Dynamic

resource allocation for load balancing in Fog environment. In: Wireless Communications and Mobile Computing. 2018

[16] Talaat FM, Ali SH, Saleh AI, Ali HA. Effective Load Balancing Strategy (ELBS) for Real-Time Fog computing environment using fuzzy and probabilistic neural networks. Journal of Network and Systems Management. 2019;**1**:1-47

[17] Khattak HA, Arshad H, Islam S, Ahmed G, Jabbar S, Sharif AM, et al. Utilization and load balancing in fog servers for health applications. EURASIP Journal on Wireless Communications and Networking. 2019;**91**:1-12

[18] Battula SK, Garg SK, Naha RK, Thulasiraman P, Thulasiram RK. A micro-level compensation-based cost model for resource allocation in a fog environment. Sensors MDPI. 2019;**19** (13):1-21

[19] Farooq MJ, Zhu Q. QoE Based Revenue Maximizing Dynamic Resource Allocation and Pricing for Fog-Enabled Mission-Critical IoT Applications. IEEE Transactions on mobile computing. 2021;**20**:3395-3408

[20] Xavier Tiago CS, Delicato FC, Pires Paulo F, Amorim Claudio L, Wei L, Albert Z. Managing heterogeneous and time-sensitive IoT applications through collaborative and energy-Aware resource allocation. ACM Transactions on Internet of Things. 2022;**3**:1-28