Journal of Information Systems Engineering and Management
2020, 5(3), em0121
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

MODESTUM
OPEN ACCESS

Scheduling in Cloud and Fog Architecture: Identification of Limitations and Suggestion of Improvement Perspectives

Celestino Barros [1]*, Vítor Rocio [2], André Sousa [3], Hugo Paredes [4]

[1] Faculty of Science and Technology of University of Cabo Verde, Praia, CAPE VERDE
[2] INESC TEC and Open University of Portugal, Lisbon, PORTUGAL
[3] Critical TechWorks, Porto, PORTUGAL
[4] INESCT TEC and University of Trás-os-Montes and Alto Douro, Vila Real, PORTUGAL
***Corresponding Author:** celestino.barros@docente.unicv.edu.cv

Citation: Barros, C., Rocio, V., Sousa, A. and Paredes, H. (2020). Scheduling in Cloud and Fog Architecture: Identification of Limitations and Suggestion of Improvement Perspectives. *Journal of Information Systems Engineering and Management, 5*(3), em0121. https://doi.org/10.29333/jisem/8429

ARTICLE INFO

Published: 30 Jul. 2020

ABSTRACT

Application execution required in cloud and fog architectures are generally heterogeneous in terms of device and application contexts. Scaling these requirements on these architectures is an optimization problem with multiple restrictions. Despite countless efforts, task scheduling in these architectures continue to present some enticing challenges that can lead us to the question how tasks are routed between different physical devices, fog nodes and cloud. In fog, due to its density and heterogeneity of devices, the scheduling is very complex and in the literature, there are still few studies that have been conducted. However, scheduling in the cloud has been widely studied. Nonetheless, many surveys address this issue from the perspective of service providers or optimize application quality of service (QoS) levels. Also, they ignore contextual information at the level of the device and end users and their user experiences.

In this paper, we conducted a systematic review of the literature on the main task by: scheduling algorithms in the existing cloud and fog architecture; studying and discussing their limitations, and we explored and suggested some perspectives for improvement.

Keywords: task scheduling, scheduling algorithm, scheduling cloud computing, scheduling fog computing, cloud-fog computing

INTRODUCTION

The current computing techniques that use the cloud are becoming unsustainable, as billions of devices, mainly driven by the rapid growth of the Internet of Things (IoT), are connected to the Internet and the data obtained by the sensors and the applications have increased exponentially. Many of these devices enable the addition of a single device application-execution functionality, communication, entertainment, games, among others. In addition, one of its main features is its ability to identify and share different types of information at the level of the user, device, application and network. On the other hand, they have several limitations such as: reduced processing capacity, scarcity of resources, reduced battery life, low connectivity, among others. These limitations impose on the analysts and developers to adopt services that enhance the ability of these applications running on these devices through the use of hosted services in the cloud (Fernando, Loke, and Rahayu, 2013). However, despite of the advantages, in some situations it is not beneficial to use the cloud architecture. It is centralized and therefore the processing is performed in data centers concentrated, to optimize energy costs and communications. Different techniques that minimize the passage in the cloud through local processing in peripheral elements and allow to solve constraints such as low latency; mobility; location; among others, it has been proposed. One technique involves the use of fog architecture that allows reduced response time, latency and power consumption compared with the cloud (Musumba and Nyongesa, 2013).

In this article, we reviewed the literature on the main task scheduling algorithms in cloud and fog architecture, we study and discuss its limitations, explore and suggest some prospects for improvement. The design of this article follows the criteria of bibliographic research, with a qualitative approach. It is divided into six sections, first being the introduction, second the contextualization. In the third, the methodology used for the literature review is described. In the fourth, scheduling algorithms in the fog-cloud architecture are presented. Based on their characteristics were categorized into: basic, priority-based and QoS-oriented and context-aware. In the fifth section, is made the discussion and limitations are identified, analyzed and suggested prospects for improvement. Finally, in the sixth session, we concluded the article.

BACKGROUND

The task scheduling refers to the allocation of resources needed to complete a task execution and it is intended that the requests are implemented taking into account the defined constraints (Musumba and Nyongesa, 2013). The task scheduling, assumes as an essential process to improve the reliability and flexibility of the systems. It requires advanced algorithms able to choose the most appropriate resource available to perform the task. The systems deal with priority requests, priority tasks and/or tasks with strict requirements of QoS. To ensure the proper functioning of these systems and the execution of tasks within the set time limits, the analysis needs to be handled with perfection. An efficient task-scheduling algorithm must ensure efficient simultaneous processing of tasks independent of their workflow. In fog architecture, there is a demand for more sophisticated task scheduling algorithms because the data needs to flow between client devices, fog nodes and cloud servers (Swaroop, 2019). A scheduling algorithm should take two important decisions can be based on some default values or through dynamic data obtained during the execution of the task (Mahmud et al., 2016): determine the tasks that can be performed in parallel and define where to execute those parallel tasks. In fog there are two main stages involved: Resource provisioning phase: this one detects, classifies and provides the resources necessary for the execution of the task and the Task mapping phase: phase in which a suitable server/processor is identified and the task is mapped to that server/processor (Mahmud et al., 2016).

Difficulties in Scheduling Algorithm Design

The design of a scheduling algorithm must observe some restrictions such as cost of tasks, dependencies between tasks and location (Swaroop, 2019). Regarding the cost of tasks, we should get answers to the following questions: what information do we have available on the cost of the tasks used? All tasks have the same cost; when can this cost be known during the execution? Regarding the dependencies between tasks, we should ask: How many dependent and independent tasks exist? What are the parent and child processes? When this relationship to the algorithm should be provided? Regarding the location, we must consider the following questions: where will the dependent and independent tasks be performed in order to reduce the total cost? How to minimize the cost of communication? How can we have information on the reporting requirements?

Decisions Escalations

The scheduling decisions can be static or dynamic (Swaroop, 2019). In static scheduling, the decision on the scheduling is made at the compiling time and so we use some static analysis methods that lets room for setting the size of the task. The acquisition of information at the compiling time in many cases is not easy, and many times it is incomplete. In the next step, a static mapping is done in the parallel architecture of the search tree, a directed graph is created for the tasks flow, where nodes represent the tasks and links represent dependencies, the communication order is then calculated for the tasks. In dynamic scheduling, which is also known as a practice of sharing adaptive tasks (Swaroop, 2019), uses information about the job status at a particular time during the task execution to make decisions; it's the best approach, because it allows several problems to have solution, and that can be represented by a search tree. However, these problems are computationally demanding, require parallelization strategy and dynamic load balancing.

METHODOLOGY

In order to achieve greater scientific rigor in this survey, we try to ensure the research process based on the prospects of systematic literature review, that identifies evaluates and interprets all relevant research available for a given subject area, one specific question, or a phenomenon of interest (Kitchenham, 2004). Its implementation must consider the following stages: preparation of the review; definition of methodologies for the implementation and extraction of the systematic review and procedures for the creation of the review reports. Due to its rigor, and possibility of iterations, the results obtained by systemic revision are more reliable in comparison to primary review of the literature (Kitchenham, 2004). Systemic review by this article took into account the application of the following methodology:

1. a database of scientific proposals using the following parameters was created: full articles available online, published in the 2015 to 2020 period, with the following keywords: *task scheduling or scheduling algorithm and scheduling cloud or fog scheduling computing and cloud-fog computing,* using the databases Science Direct, 371 articles, IEEE Explore, 108 articles, Google Scholar, 278 articles, ACM, 302 articles and SCOPUS, 200 articles. As shown in **Figure 1**, totaling 1.259 articles.

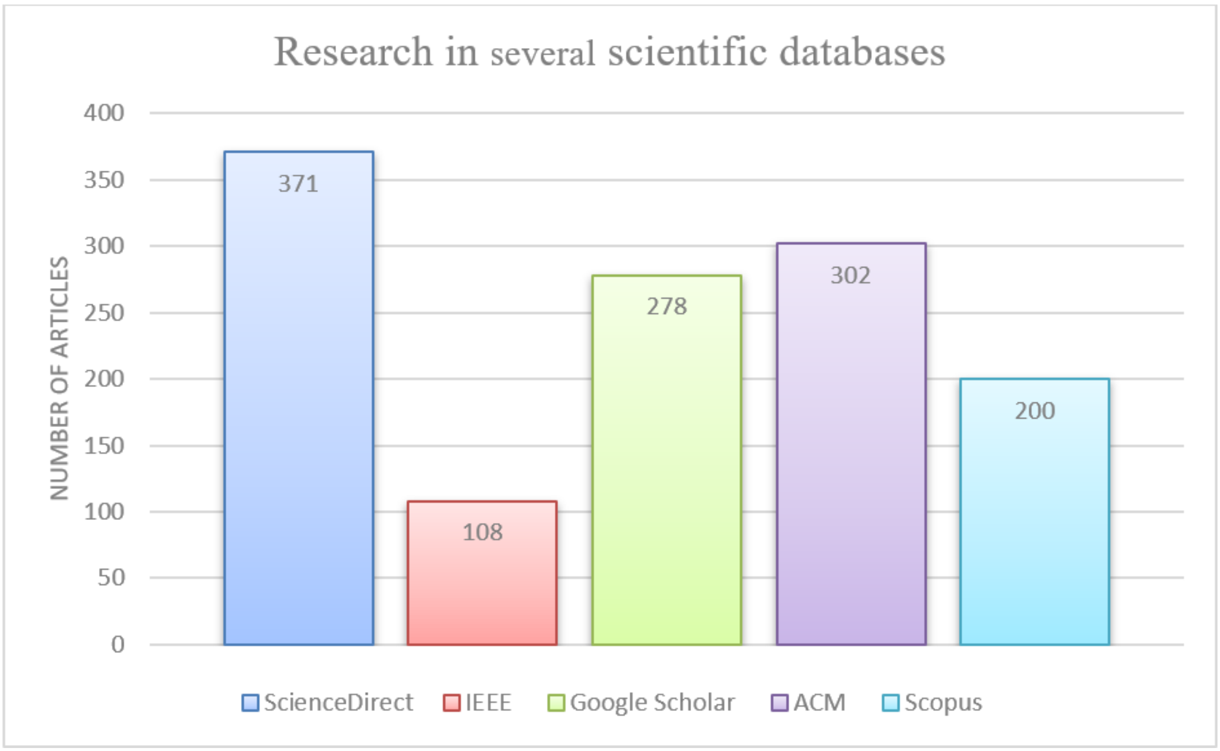

Figure 1. Identified articles

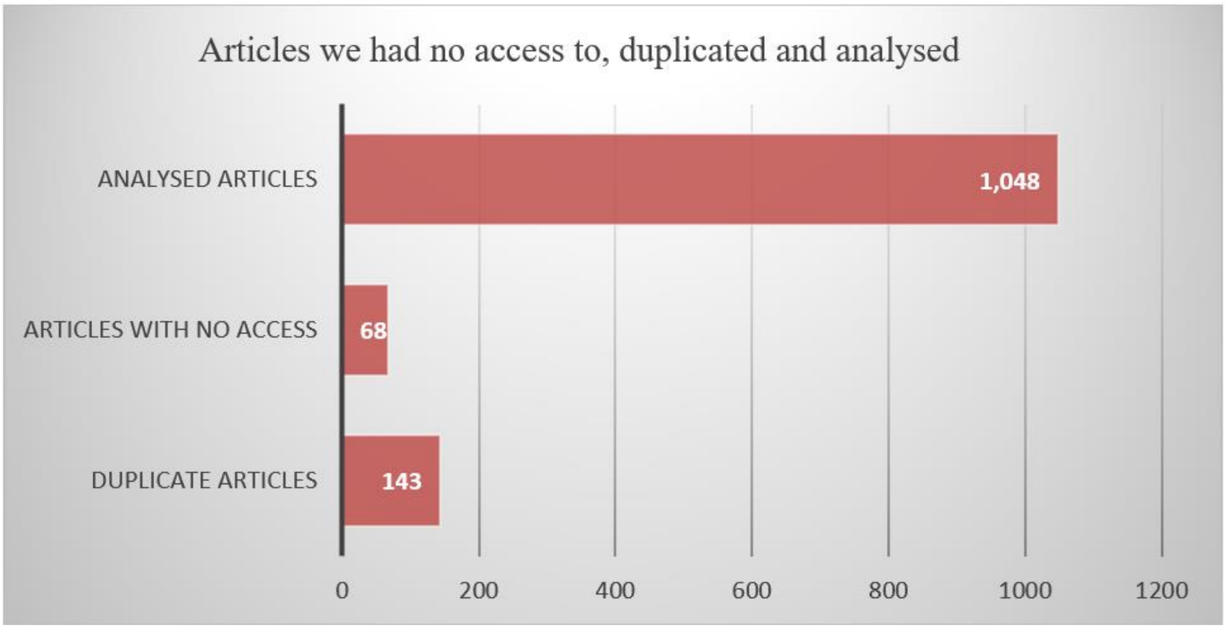

Figure 2. Filtering the identified articles

Results were imported into the Publish or Perish software (Harzing, 2020), a free application that provides simple metrics on the articles, namely, the level of total citations, adjusted to the year, automatic detection of duplicates, among others.

2. After cleaning the duplicated items 143, a list was compiled of 1.116 articles, of which 68 were removed by not having access to the document. **Figure 2** shows the numbers of articles that we did not access, duplicate and analyzed.
3. Based on the list of 1,048 articles, we proceed to the exclusion of proposals not directly relevant to the requirements, “not to be a task scheduling proposal, or task scheduling algorithm” and “not to be related to cloud or fog computing”. This exclusion was performed by analyzing the titles and abstracts of articles and, when in doubt, for analysis of the article, having removed the type of articles: Survey (53), theoretical reviews or position papers (78), not be a proposal of tasks scheduling, or task scheduling algorithm (184), not related to cloud or fog computing (450). After this analysis, we obtained a list of 285 papers, as shown in **Figure 3**.

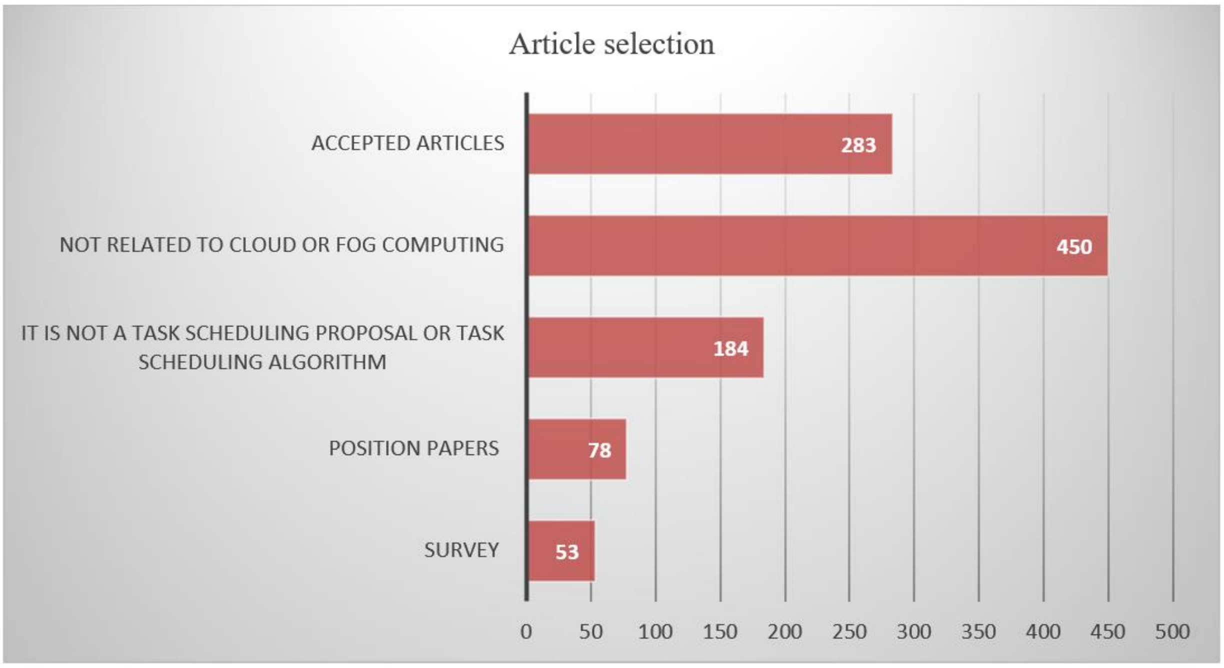

Figure 3. Article selection

4. Selecting a reduced set of proposals, from the articles 283, based on the following rules:
    a. Ordering by number of citations in the literature adjusted annually (calculated by the Publish or Perish software) - yielded a set of proposals that have already been analyzed and validated by other authors, and remove the bias caused by the publication time;
    b. Relevance to the theme: yielded a set of proposals that meet the needs, in particular to be inserted into the theme of task schedules algorithms in the cloud or fog architecture or be task scheduling proposals in cloud or fog architecture - were excluded 195 items not relevant and six by not having access.

Based on this selection method, a sample of 30 proposals was obtained and will be presented and analyzed in next section.

SCHEDULING ALGORITHMS

The selected proposals have been categorized according to their characteristics into: Basic, based on priority and QoS-oriented and context-aware.

Basic Scheduling Algorithms

Both in cloud and fog architecture we found some scheduling algorithms that due to its characteristics were characterized as basic.

The authors in Deng et al. (2016), examined the relationship between the delay in transmission and energy consumption in the cloud. They also advanced that the fog architecture enables mobility, geographic distribution, location and mobility of the user supported by the fog devices. The geographical distribution can provide context information from the network, so it can be used by the application in the fog.

In Li et al. (2017), it is proposed the Energy-Aware Dynamic Task Scheduling aimed at minimizing the energy consumption of cyber physical systems, with the focus on smartphones, which due to their limitations, have the need to make many and frequent interactions with the cloud, in order to perform tasks such as: request resources, send requests and receive responses. These tasks are simple, but due to its often consumption of a lot of energy, to achieve the perfect performance, they proposed to solve two problems. The first is to keep maintaining the performance, reduce to a minimum the cost of energy. The second is to make highly reliable tasks. That is, allow tasks to be completed with high probability success.

The authors in Lawanyashri et al. (2017) presented the Fruity Fly Algorithm where the proposed approach is focused on the issue of load balancing, but also emphasizes the power consumption reduction in data centers. The proposed method was able to optimize the use of resources and reduced the cost and power consumption. One based on game model in which the player maximizes his reward in order to minimize the response time. That is, there is a minimization of the response time at the end of the task unloaders and maximizing battery life in tasks receivers is proposed in (Tiwary et al., 2018).

In Yang et al. (2018), the Delay Balanced Energy Task Scheduling (DEBTS) is presented as a multi-layer analytical framework to formulate and study the balance between the delay of the service and energy consumption in task scheduling in fog networks.

The Deadline-Aware Task Scheduling Mechanism is proposed in Fun et al. (2017), through it, service providers exploit the collaboration between the nodes of the fog itself and the resources of rented cloud to perform efficiently the tasks of users in large geographical scale. The main objective is to maximize the profits of fog service provider, through the satisfaction of time set for the execution of tasks.

The authors in Shinde et al. (2018) proposed an optimization method called BAT + BAR to perform the task scheduling and allocation paid distributed computing resources according to the use. The proposed structure classifies tasks and allocates according to the size restriction and bandwidth in a virtual machine. They compared both algorithms in terms of performance metrics and ascertained improvement.

The Energy-Aware Multi-Job Scheduling Model is presented in Wang, Wang, and Cui (2016), it is based on MapReduce and aims to improve energy efficiency through server CPU setting. A survey of the cloud characteristics and the way it provides several resources to customers through various service models is presented in (Prasan and Kumar, 2018). They proposed a scheduling algorithm designed to stagger the tasks related to health, managed and stored in the data center.

The authors in Sarkar, Chatterjee, and Misra (2015), evaluated the applicability of fog architecture in order to meet the demands of latency sensitive applications in the context of IoT and concluded that the goal of fog, which aims to provide answers to requests in real time with low latency, favors its use in the context of IoT. The adequacy of the fog in the context of requests in real time and performance in relation to the history of distributed computing has been evaluated and reached a low power consumption, low rates of carbon dioxide gas, latency and cost.

In Bitam, Zeadally, and Mellouk (2018), is presented the Life Bees Algorithm (BLA) that aims to find the balance between task execution time in the CPU and its allocation in memory. The proposed algorithm has achieved the objective of minimizing the execution time compared to the algorithms Particle Swarm Optimization and Genetic Algorithm. A hybrid approach based on fuzzy theory and genetic algorithm, called FUGE is presented in Shojafar et al. (2015), it is intended to perform a great load balancing considering the time and cost of implementation. The proposed algorithm has achieved the objective of minimizing the execution time compared to the algorithms Particle Swarm Optimization and Genetic Algorithm.

The Hybrid Fog and Cloud-Aware Heuristic (Hybrid-EDF) has been proposed and described in Stavrinides, Karatza (2019), this approach uses spaces in the scheduling of virtual machines in cloud and fog, to scale computationally demanding tasks with low communication requirements in the cloud and intensive communication tasks with low computational requirements in fog. During scheduling, the proposed approach takes into account the cost of communications arising from the transfer of data from the sensors and devices of the fog layer. The performance of the heuristic proposal was evaluated and compared with the Baseline Cloud-Unaware Strategy algorithm, Fog-EDF.

The Cooperative Model-Based Sensing for Smart-Tasks (CMST) was proposed in Li et al. (2018), it aims to promote the quality of data collection in urban and suburban areas for feeding platforms through the fog architecture. It used a cooperative scheduling algorithm focused on improving the rewards associated data collections in suburban regions. The rewards for each user with a smart sensor are distributed according to the density of the region. In addition, for each task there is a relationship of cooperation between the users, who cooperate to achieve the volume of data that the platform requires.

Priority-based and QoS-Oriented Scheduling Algorithms

Research on priority-based and QoS-oriented task scheduling in the fog infrastructure is recent. In Skarlat et al. (2017), a scheduling approach aimed at putting QoS sensitive applications in virtualized resources of fog has been set. They consider the deadline satisfaction the implementation of applications such as QoS metrics. The proposed policy goes through a colony-based orchestration between nodes of fog and reconciles the application resource requirements with the available system resources. When colonies require additional resources, connect to cloud via a middleware.

The authors in Gill, Garraghan, and Buyya (2019), put forward that one of the main challenges of fog architecture is the resource management and that, although the standard fog architecture exists there, they focus only on the management of an important subset of parameters that cover the system response time, network bandwidth, power consumption and latency and so far, no fog resource manager considered all these parameters simultaneously for the taking decisions and they proposed a new resource management technique called ROUTER, which takes advantage of Particle Swarm Optimization to improve both these parameters: system response time, network bandwidth, power consumption and latency and enable better decision-making.

Instead, in Aazam et al. (2016), they proposed a resource estimate policy fog based on QoE, MEdia fog entitled Resource Estimation (MeFoRE). The proposed policy considers the user’s service waiver history (dropout rate) and QoE (NPS) while prioritizing service requests and estimates the resources of fog, in order to maximize the use of resources and QoS. Violations of service-level agreement (SLA) is monitored by low NPS user-provided values. The increase in the amount of resources based on the degree of SLA violation and user confidence can be recovered. Service Level Agreement (SLA) violations are monitored by low NPS values provided by users.

The issue of load balancing in fog architecture, in order to improve the quality of user experience were analyzed in (Oueis, Strinati, and Barbarossa, 2015). The authors consider that all requests for different users whose computing needs to be discharged are performed on local resources clusters. In this proposal, a reduced complexity of task scheduling algorithm for fog where resources are allocated in order to serve small cells (ie, fog node) was introduced based on some specific escalations rules. The main rule is to define the allocation of local computing resources in each small cell to serve internal users. Each small cell needs the user’s discharge rate according to a specific parameter, such as: time of arrival; latency restriction; among others. This classification also defines the scheduling policy (for example, First In First Out (FIFO), Earliest Deadline First (EDF)) to be adopted for the allocation of local resources. Thus, they can be given different priorities for different user requests, depending on the sort parameter.

In Cardellini et al. (2015), the authors evaluated the distributed scheduler service quality QoS data flow for recognition processing (DSP) in fog environment. They innovated by introducing new components such as monitor worker; QoS monitor and a native adaptive scheduler. The worker monitor, this architecture perform the function of obtaining the data input and output rate for each performer is defined as a computer component that implements a set of tasks in the fog node. This rate of input and output data is stored in a local database for further use by the adaptive stepper. The QoS monitor, estimates the QoS parameters (eg, network latency) and it is responsible for obtaining and using intra-node and intra-us information availability. This information is sent to the adaptive distributed scheduler, which implements the system scheduling policy. The adaptive scheduler performs a single iteration cycle periodically, in order to verify that the candidate task will be executed (called mobile performer). If the performer is effectively allocated, the adaptive scheduler performs the corresponding actions. For this, the scheduler determines a node that will run the executor candidate only if that node improve application performance in terms of runtime resources. If the performer is effectively allocated, the adaptive scheduler performs the corresponding actions. For this, the scheduler determines a node that will run the executor candidate only if that node improve application performance in terms of runtime resources. If the performer is effectively allocated, the adaptive scheduler performs the corresponding actions. For this, the scheduler determines a node that will run the executor candidate only if that node improve application performance in terms of runtime resources.

The authors in Intharawijitr, Iida, and Koga (2016), defined a mathematical model of a fog network to assist a computer system (eg based infrastructure in fog 5G) to achieve maximum efficiency, ensuring optimal scheduling tasks. For task scheduling in fog, three policies were considered in this study: the random policy in a fog node is selected randomly from a uniform distribution to perform a task; The lower latency policy in a fog node provides the task with the lowest latency based on the current state of the system and ultimately the capacity of policy available, which selects the fog node, with maximum excess funds between candidate nodes. The simulation results showed that the lowest latency policy provides significantly better performance due to the availability of resources.

The authors concluded that all policies could be used to find the fog node best suited for a task. However, the use of a particular policy may not be the ideal solution for the entire system.

In Bittencourt et al. (2017), the authors analyzed the task scheduling problem in fog and focused on how the mobility of users influence the performance of applications. Evaluated as scheduling algorithms: Concurrent, First Come First Served and Delay-Priority, can be used to improve the performance of the task based on the characteristics of applications. Unlike fog, the scheduling tasks based on priority and QoS oriented, it has been extensively studied in the cloud.

Zhou et al. (2015) proposed a task scheduling algorithm based on QoS, guided by priority and completion time for the Mobile Cloud Server that based on the task attributes such as: User privilege; expectation; Task volume and suspended time in the queue, priority is calculated, then, the tasks are scheduled based on the minimum completion time provide better performance and load balancing.

The Offline Multi-Level Scheduling was proposed in Seddik and Hanzálek (2017), and aims to solve the problem of criticality level of each task, based on their priority, they consider the total time of task processing and investigated in view of the ultimate goal for the weighted number of the executed task as fulfilled the requirement. The proposed method achieved tasks that are more critical.

Context-aware Scheduling Algorithms

Research on context-aware scheduling algorithms in fog architecture is very recent. In the cloud architecture, we find some proposals.

In Shi et al. (2016), an adaptive probabilistic scheduler is presented, which optimizes the task power consumption with time restrictions and in Zhou et al. (2017), the Context-Aware Offloading Framework is proposed a decision algorithm discharge with recognition of the context that takes into account the context of changes such as: network tiers and heterogeneous mobile cloud resources to provide code discharge decisions. Also, they have provided a general model of cost estimate for the mobile cloud infrastructure resources to estimate the cost of performing the task, including run time, energy consumption.

In Mahmud et al. (2016), an application scheduling policy with context recognition is presented in Mobile Cloud Computing (MCC) that allows the improvement of the user QoE. The algorithm runs on a Cloudlet and prioritizes the requests of users based on the level of the applicant’s device battery and signal / noise ratio of the network. Also care about improving the QoE of the user. The Highest Priority Job First algorithm (HPJF) was used to sort, preemption and scheduling tasks.

The Multi-Capacity-Aware Resource Scheduling is proposed in Sheikhalishahi et al. (2015), to solve the multi-resources scheduling problem. The scheduling plan allows any task to be chosen on the basis of resource requirements similar to those of the available system and a specific final objective that aimed at dealing with the use of resources in a context of several resources.

The Context Aware Cloud System was proposed in Ghouma and Jaseemuddin (2015), aimed at scaling and resource allocation with recognition of the context. Monitors and uses the user’s device context information to make intelligent resource allocation decisions in the cloud and scheduling tasks. The mobile user context information (device connection quality and battery level), are used in the definition of services to the flows of application tasks that are sent and processed in the cloud. The proposed model aims to rationalize the use of cloud environment resources and provides significant improvement in service quality.

DISCUSSION

Some authors such as (cf. Sheikhalishahi et al. (2015), Lawanyashri et al. (2017), Tiwary et al. (2018) and Shinde et al. (2018)) analyze the scheduling under the perspective of service providers and ignore the contextual issues of end-users.

The authors in Deng et al. (2016), Li et al. (2017), Lawanyashri et al. (2017) and Yang et al. (2018), proposed approaches that emphasize the reduction of energy consumption from data centers and note a great concern of the authors to energy efficiency.

In Zhou et al. (2015), an algorithm is proposed where the attributes of tasks such as user privileges; size of the task; expectation and suspended time in the queue are used to calculate the priority. The scheduling algorithm proposed in Stavrinides, Karatza (2019), takes into account the cost of communication due to data transfer from sensors and fog-layer devices during the scheduling process.

The authors in Aazam et al. (2016), and Skarlat et al. (2017), proposed algorithms that aim to maximize the use of resources and QoS. These scaling algorithms do not exploit the connectivity levels of devices to prioritize the execution of tasks. The proposal in Intharawijitr, Iida, and Koga (2016), try to ensure the best use of the width of available bandwidth.

In Sheikhalishahi et al. (2015), Deng et al. (2016) and Lawanyashri et al. (2017), there is an evident concern of the authors with the energy efficiency of resources. The algorithm proposed in Shojafar et al. (2015) and Fun et al. (2017), aims to meet the deadline of tasks and maximize profits of the service provider of fog aim to perform an optimal load balancing considering the time and cost of execution of tasks.

In these last five proposals, there is an evident concern of the authors and to focus primarily defend the interests of service providers, rather than consider the contexts of end users and their use of experiences.

The authors in Ghouma and Jaseemuddin (2015), Zhou et al. (2017), and Li et al. (2018), explore the levels of connectivity of the network and leased virtual machine resources to provide codes discharge decisions. However, they ignore important parameters of context as QoS requirements, traffic on the network, network bandwidth, and others who can influence extensively the actual decisions taken.

The algorithm proposed in Mahmud et al. (2016), runs on a Cloudlet and prioritization of requests are made considering the parameters of contexts battery level and device network. The parameters contexts explored by the authors, in our perspective are important and are intended to remedy shortcomings of requests from mobile devices and increase the users' QoE.

The algorithm proposed in Zhu et al. (2015), is used in tasks grouped in order to reduce the expected run time. The proposed in Oueis, Strinati, and Barbarossa (2015), despite providing latency gain and low energy consumption, as those proposed in Cardellini et al. (2015) and Zhu et al. (2015), performance degrades when used in large-scale fog infrastructure.

In Gill, Garraghan, and Buyya (2019), a new resource management technique for environments cloud and fog-aware QoS is proposed that, contrary to standard fog scheduler takes advantage of the Particle Swarm Optimization to simultaneously optimize various context parameters.

In Aazam et al. (2016), it is proposed an algorithm that aims to estimate the resources of fog based on QoE. Instead, Skarlat et al. (2017), the approach aims to scale QoS-sensitive applications in virtualized fog resources. They consider the satisfaction of the set time limit for the implementation of applications such as QoS metric, focused on response time optimization of the components of the fog and did not consider the time of service or the user experience in the event of battery life be reduced and low connectivity network.

Contrary to Zhu et al. (2015), Oueis, Strinati, and Barbarossa (2015), Mahmud et al. (2016) and Aazam et al. (2016), the issues related to improving end-user QoE were considered. It proposed an approach to scaling QoS sensitive applications in virtualized resources of the fog. They considered meeting the deadline for the implementation of applications such as QoS metrics, focused on optimizing the response time of the fog components, and did not consider the service time or user experience in case of reduced battery life and low connectivity network. They still advance that, despite many efforts, there are still several challenges related to task scheduling in cloud and fog architecture.

Many proposals such as those described in: Sheikhalishahi et al. (2015), Lawanyashri et al. (2017), Tiwary et al. (2018) and Shinde et al. (2018), address the problem of optimizing the use under the perspective of service providers and ignore contextual issues of end-users and their use of experiences. Others, such as those defined in: Cardellini et al. (2015), Aazam et al. (2016), Skarlat et al. (2017), and Gill, Garraghan, and Buyya (2019), are mainly intended to optimize QoS levels of application and some focus only on the task scheduling in cloud and fog architecture. Still others worry about the energy efficiency.

Despite many efforts, there are still many challenges, in the task scheduling in cloud and fog architecture. Different schedulers have their own shortcomings, in the following, as defined in Barros et al. (2020), we highlight some limitations of schedulers presented in Section "Scheduling Algorithms":

Analyze policies from the perspective of services, most schedulers, analyze policies only in the service perspective. The optimization of costs for users and providers, as well as improving the user experience quality are not taken into account;

Lack of end-user context, the scheduling techniques with existing context-aware, end-user connections are analyzed in a limited scope. Signal strength associated with a request for example, are not taken into consideration. As a result, any device may be disconnected before or while obtaining a request response; the level of the device's battery end-users is also ignored. To ensure that a request has always answers in due time, a limit to the battery level should be preserved;

Poor scheduling tasks, basic tasks schedulers such as defined in: Li et al. (2017), Lawanyashri et al. (2017), Yang et al. (2018), Wang, Wang, and Cui (2016), and Li et al. (2018), are schedulers that privilege energy efficiency and do not consider the other problems as the recognition context, quality of service and experience of users;

Inadequate prioritization of tasks, some schedulers based on priority have been proposed. However, some do not provide how the priority is set and others do not clearly explain the methodology used to prioritize tasks;

Increasing the average waiting time, usually as the requests increase, the average waiting time also tends to increase proportionally. In schedulers analyzed, no compensation for this issue was proposed;

Quality of experience subtle, though there are some scheduling algorithms that are based on QoS for prioritization of tasks, do not focus on maximizing the quality of user experience.

Supervision to maintain service quality: schedulers analyzed do not supervise the quality of service. This is the maximum time allowed to obtain answers are not considered in some proposals and others are considered improperly.

While there are several schedulers for cloud-fog architecture that allow solve many problems, some aspects can be explored in order to achieve improvements over existing strategies. Then we suggest, as defined in Barros et al. (2020), some improvements to the analyzed schedulers:

Awareness of the context in task scheduling, several studies ensure that informed decisions from end user context can be effective with regard to the improvement of service and optimize costs, from the point of view of users and the service providers;

Prioritization of tasks with context recognition: should be introduced scheduling models of prioritized applications from the context of recognition, where each request articulates information from the device and the contexts of the application (eg, battery level, network signal strength, application QoS, among others). Considering these contexts associated with a request, the scheduler must do the scheduling in fog node in order to obtain better performance.

Energy restriction of preservation: given that many devices have less processing power and reduced battery life, energy restriction should be taken into consideration during the execution of tasks in architecture cloud and fog. The level of the end user device battery associated with a request should be limited to a battery threshold level, so that the requesting device is preserved until the end of the run.

Conservation of the network signal strength: signal strength associated with a request to ensure minimum signal strength to allow a request for resources, the end user device, in order to allow the user to request features and get answers in timely.

Safeguarding the quality of service: given that the tasks transferred to the fog are differentiated in terms of device context, the end-user and application requirements, including their QoS. Therefore, when scheduling tasks, it is necessary to take into account the time required for obtaining answer must be confined within that time limit.

Reducing the average waiting time: compensation for the average waiting time should be introduced by scheduling tasks for the increased waiting time in relation to the task arrival rate is decreased.

Maximizing the quality of experience, The QoE of using a service or product is defined through the analysis of the user's performance in objective and subjective psychological dimensions. The user gain or loss is one of the objective measures of QoE. The scheduling algorithm should also concentrate on maximizing the quality of the user experience and the quality of service.

A summary of the studied scheduling algorithms, elucidating their advantages and inconveniences is provided in **Table 1**.

Table 1. Summary of the analyzed scheduling algorithms

Authors	Algorithms	Advantages	Inconveniences
Shinde et al. (2018)	BAT+ BAR	Facilitates resource rental in the *cloud*.	Scheduling and resource allocation are very complex
Wang, Wang, and Cui (2016)	Energy-Aware Multi Job Scheduling	Improves server-energy efficiency in data allocation	Has limitations in task scheduling and optimization
Bitam, Zeadally, and Mellouk (2018)	Bees Life Algorithm (BLA)	Maximizes task execution time	There is a lack of studies regarding its use in environments such as *fog*
Aazam et al. (2016)	MEdia Fog Resource Estimation (MeFoRE)	Estimates *fog* resources based on user's QoE, allowing for the optimization of the use of resources and QoS; Enables decentralized management while prioritizing tasks during scheduling.	Does not respect user expectations when accessing the service or processing time; Processing speed is, also, not satisfactory
Woo, Jung, and Kim (2017)	Genetic Algorithm	Allows the reduction of task execution time	Has little improvisation, which makes it difficult to use in *fog* architecture
Sarkar, Chatterjee, and Misra (2015)	Fog Computing Paradigm	Improves the performance of latency-sensitive applications	Enables large data traffic
Deng et al. (2016)	Fog –Cloud Architecture	Improves mobility, geographic distribution and location.	Response latency can lead to service removal
Lawanyashri et al. (2017)	Fruity Fly Algorithm	Improves availability and scalability; Emphasizes energy consumption reduction, minimizing costs	Resource allocation is not done in a balanced fashion
Tiwary et al. (2018)	Non-Cooperative Game Model	Improves the response time of requests.	Executes tasks with limited sizes
Sheikhalishahi et al. (2015)	Multi-Capacity Aware Resource Scheduling	Improves resource use and strives for energy efficiency	Uses less CPU, making task processing very complex
Seddik and Hanzálek (2017)	Offline Multi-Level Scheduling	Improves resource use	Defining the processing time for a task is complex
Prasan and Kumar (2018)	CPU Intensive Scheduling Data-Intensive Scheduling	Improves server and network infrastructure	Can cause unavailability of resources and have a negative impact at the level of QoS
Wang, Wang, and Cui (2016)	Energy-aware multi job scheduling	Improves server-energy efficiency in resource allocation	Has unsatisfactory performance in large task scheduling and poses optimization problem
Oueis, Strinati, and Barbarossa (2015)	Reduced Complexity Task Scheduling Algorithm for Fog	Provides a high degree of user satisfaction in terms of latency gains and low energy consumption	Suffer with the high complexity of *fog* infrastructure*;* Algorithms used achieve good results only in low-density computing infrastructures
Skarlat et al. (2017)	QoS-Aware Application Placement on Virtualized Fog Resources	Complies with expectations in terms of resource balance and processing time and satisfies the status of instances regarding resource availability of and processing speed; Enables decentralized management and prioritizes tasks during scheduling	Does not respect expectations in terms of access to the service; Does not meet status of instances for proximity or response rate
Zhou et al (2015)	QoE-Driven Video Cache Allocation Scheme for Mobile Cloud Server	Complies with expectations in terms of access to the service, satisfies the status of instances regarding the proximity or response rate and resource availability; Enables decentralized management and prioritizes tasks during scheduling	Does not respect expectations in terms of resource balance and processing time; Does not meet the status of instances regarding processing speed
Cardellini et al. (2015)	Quality of Service Distributed Scheduler with QoS Awareness	In small environments, it improves application performance and enhances the system with runtime adaptation features	When in complex fog topologies with many operators it causes instability, decreasing application availability levels
Mahmud et al. (2016)	QoE and Context-Aware Scheduling (QCASH)	Complies with expectations in terms of access to the service and processing time; Prioritizes tasks by exploring contexts such as battery level and signal quality when scheduling and optimizes user QoE	Management is centralized, making it difficult to implement in densely distributed environments such as fog
Shi et al. (2016),	Adaptive Probabilistic Scheduler	Allows the reduction of average energy consumption in a successful task and maintains a high task execution rate. It has a high adaptability in both fixed and mobile networks.	Can cause imbalance in the use of resources.
Zhu et al. (2015)	Priority-Based Two-Phase Min–Min Scheduling Policy (PTMM)	Can be applied to grouped tasks in order to minimize the expected execution time;	Poor performance if requests are complex, which can result in poor QoE of users
Oueis, Strinati, and Barbarossa (2015)	Algorithm Designed for Cluster Formation and Load Balancing for Fog Computing	Allows customization of metrics; scheduling rules and clustering objectives can be defined according to specific applications and network requirements; Enables reduced complexity and multi-parameter optimization; Allows significant latency reduction compared to other algorithms and has low average energy consumption	It proved to be an effective solution to be adopted for energy-efficient small cell clustering, with performance deteriorating for large-scale fog architecture

Table 1 (continued). Summary of the analyzed scheduling algorithms

Authors	Algorithms	Advantages	Inconveniences
Li et al. (2017)	Energy-Aware Dynamic Task Scheduling	Reduces the energy consumption of cyber-physical systems, compared to the critical path scheduling method and the parallelism-based scheduling method	This is still an understudied approach. It was tested only on Android mobile devices. There is a need to run more tests and combine more Mobile Edge Computing techniques.
Li et al. (2018)	Cooperative-Based Model for Smart-Sensing Tasks (CMST)	Extensive experiments have shown that CMST improves data coverage and quality and reduces costs.	Shows good performance only in specific cases such as data collection aiming at feeding platforms.
Shojafar et al. (2015)	FUzzy GEnetic Task Scheduling (FUGE)	Results of the experiments show the efficiency of this approach in terms of time, execution cost, and average degree of imbalance.	It does not consider energy consumption and the migration of virtual machines; it is not energy efficient and consistent; When scheduling, it only considers the execution time of a task (makespan)
Yang et al. (2018)	Delay Energy Balanced Task Scheduling (DEBTS)	Minimizes energy consumption and reduces the average service delay and the delay of data delivery on the network (jitter) when scheduling tasks; Enables better performance than traditional algorithms like *Random scheduling* and *Least busy scheduling*.	It has been tested only in simple fog architecture; so there is the need to be tested in heterogeneous and complex fog architectures.
Gill, Garraghan, and Buyya (2019)	ROUTER	Has allowed a reduction in the use of network bandwidth, response time and latency, and energy consumption.	Tested on a small-scale Smart Home; Does not meet parameters such as scalability, cost, reliability and availability.
Fun et al. (2017)	Deadline-Aware Task Scheduling Mechanism	Improves system performance when compared to algorithms such as *Min-Min* and *First Come First Served.*	In large and densely distributed environments such as fog, performance decays and there is the need to consider distributed and lightweight algorithms for optimization.
Stavrinides, Karatza (2019)	Hybrid Fog and Cloud-Aware Heuristic (Hybrid-EDF)	Provides, on average, a decrease in delivery failures because it uses cloud resources, especially when it is made of on-demand multi-tenant virtual machines.	Presents a high cost when the cloud layer consists of virtual machines on reserved and dedicated hosts.
Zhou et al. (2017)	Context-Aware Offloading Framework	Provides download decisions based on the current context of the devices, allowing the reduction of execution time cost and energy consumption.	To enhance efficiency and reliability, there is the need to expand the structure so that cloud resources have the ability to intercommunicate, based on changes in context
Ghouma and Jaseemuddin (2015)	Context Aware Cloud System	Reduces task execution cost and, at the same time, fulfills the defined execution length.	The high latency associated with the cloud makes it impossible to use it in applications that require low latencies.

CONCLUSIONS

In this article, we reviewed the literature on the task scheduling algorithms in cloud-fog architecture, we studied and discussed their limitations and we explored, suggested some perspectives for improvement.

The main conclusions were that the fog architecture, due to its density and heterogeneity of devices, the analysis is very complex and in the literature, there are few studies. In contrast, the analysis in the cloud has been extensively studied.

With regard to the task scheduling algorithms in the cloud and fog, most analyzes the policies from the perspective of service providers. Others are applied in tasks grouped to minimize the time of expected completion and some maximizes application QoS. Still others explore contexts such as the battery level, rented virtual machines and presence applications. However, in these latters, the connection of the devices end users' applications is analyzed in a limited scope.

Many of the task scheduling algorithms in cloud-fog infrastructure do not consider the prioritization of tasks based on the context information.

Although in Ghouma and Jaseemuddin (2015), Mahmud et al. (2016), Li et al. (2017), and Zhou et al. (2017), the context parameters such as levels of network connectivity device, battery level, renting virtual machines are considered. However, many authors have ignored other important context parameters such as quality of service requirements, bandwidth, among others, that can influence extensively the taken of effective decisions. In other formulation for setting the priorities of the process table is covered in a practical way. Others still ignore how the heterogeneity of different contexts is treated, many dismiss the improvement of user experience quality, and policies are analyzed only in the service perspective.

Current schedulers allow solving many problems. however, some aspects such as the consideration of context awareness in task scheduling; prioritizing tasks with context recognition; consideration of energy restriction; preservation of the network signal strength; preservation of the quality of service; reducing the average waiting time and maximizing the quality of the experience can be explored and improved.

ACKNOWLEDGEMENT

The authors would like to express their sincere thanks to Calouste Gulbenkian Foundation for its funding of this research through the PhD scholarship under the reference No. 234242, 2019 - Postgraduate Scholarships for students from PALOP and Timor-Leste.

REFERENCES

Aazam, M., Hilaire, M. St., Lung, Ch. and Lambadaris, I. (2016). MeFoRE: Resource Estimation QoE based at Fog to Enhance QoS in IoT. In*: Proc. of the 23rd International Conference on Telecommunications*, ICT '16, IEEE, pp. 1-5, https://doi.org/10.1109/ICT.2016.7500362

Barros, C., Rocio, V., Sousa, A. and Paredes, H. (2020). Survey on Job Scheduling in Cloud-Fog Architecture. *2020 15th Iberian Conference on Information Systems and Technologies (CISTI), Sevilla, Spain*, pp. 1-7. https://doi.org/10.23919/CISTI49556.2020.9141156

Bittencourt, L. F., Diaz-Montes, J., Buyya, R., Rana, O. F. and Parashar, M. (2017). Mobility-Aware Computing in Fog Application Scheduling. *IEEE Cloud Computing, 4*(2), 26-35, https://doi.org/10.1109/MCC.2017.27

Cardellini, V., Grassi, V., Presti, F. L. and Nardelli, M. (2015). On QoS-Aware Scheduling of Data Stream Applications over Fog Computing Infrastructures. *IEEE Symposium on Computers and Communication (ISCC)*, pp. 271-276, https://doi.org/10.1109/ISCC.2015.7405527

Deng, R., Luan, T. H., Lu, R., Liang, H. and Lai, C. (2016). Optimal Allocation Workload in Fog-Cloud Computing Towards Balanced Delay and Power Consumption. *IEEE Internet Things J., X*(X), 1171-1181, https://doi.org/10.1109/JIOT.2016.2565516

Fan, J., Wei, X., Wang, T., Lan, T. and Subramaniam, S. (2017). Deadline-aware task scheduling in a Tiered IoT Infrastructure. GLOBECOM 2017 - 2017 IEEE Global Communications Conference, Singapore, pp. 1-7, https://doi.org/10.1109/GLOCOM.2017.8255037

Fernando, N., Loke, S. W. and Rahayu, W. (2013). Mobile cloud computing: The survey. *Future Generation Computer Systems, 29*(1), 84-106, https://doi.org/10.1016/j.future.2012.05.023

Ghouma, H. and Jaseemuddin, M. (2015). Context aware resource allocation and scheduling for mobile cloud. *2015 IEEE 4th International Conference on Cloud Networking (CloudNet), Niagara Falls, ON*, pp. 67-70, https://doi.org/10.1109/CloudNet.2015.7335282

Gill, S. S., Garraghan, P. and Buyya, R. (2019). ROUTER: Fog enabled cloud based intelligent resource management approach for smart home IoT devices. *Journal of Systems and Software, 154*, 125-138, https://doi.org/10.1016/j.jss.2019.04.058

Harzing, A. W. (2020). *Publish or perish*. Available at: https://harzing.com/resources/publish-or-perish (Accessed: 06 July 2020).

Intharawijitr, K., Iida, K. and Koga, H. (2016). Analysis of Fog Model considering Computing and Communication Latency in 5G Cellular Networks. IEEE International Conference on Pervasive Computing and Communication Workshops (Workshops Percom), pp. 1-4. https://doi.org/10.1109/PERCOMW.2016.7457059

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews. *Join Tecnical Report, Keele University RT / SE-0401*. Available at: http://www.it.hiof.no/~haraldh/misc/2016-08-22-smat/Kitchenham-Systematic-Review-2004.pdf (Accessed: 6 July 2020).

Lawanyashri, M., Balusamy, B. and Subha, S. (2017). Energy-Aware fruitfly hybrid optimization for load balancing in cloud environments is EHR applications. *Informatics Med. Unlocked, 8*(March), 42-50, https://doi.org/10.1016/j.imu.2017.02.005

Li, Q., Novak, E., Yi, S. and Hao, Z. (2017). Challenges and Software Architecture for Fog Computing. *IEEE Internet Computing, 21*(2), 44-53. https://doi.org/10.1109/MIC.2017.26

Li, T., Liu, Y., Gao, A. L. and Liu, A. (2017). A for Cooperative - based Smart Sensing Tasks in Fog-Computing. *IEEE, Access, 5*, 21296-21311. https://doi.org/10.1109/ACCESS.2017.2756826

Mahmud, M. R., Afrin, M., Razzaque, M. A., Hassan, M. M., Alelaiwi, A. and Alrubaian, M. (2016). Maximizing Quality of Experience through Context-Aware Mobile Application Scheduling in Cloudlet Infrastructure. *Software: Practice and Experience, 46*(11), 1525-1545. https://doi.org/10.1002/spe.2392

Musumba, G. W. and Nyongesa, H. O. (2013). Context awareness in mobile computing: a review. *International Journal of Machine Learning and Applications, 2*(1), 1-5, https://doi.org/10.4102/ijmla.v2i1.5

Oueis, J., Strinati, E. C. and Barbarossa, S. (2015). The Fog Balancing: Load Cell Distribution for Small Cloud Computing. *IEEE 81st Vehicular Technology Conference (VTC Spring)*, Glasgow, pp. 1-6. https://doi.org/10.1109 / VTCSpring.2015.7146129

Sahoo, P. K. and Dehury, C. K. (2018). Efficient data and CPU-intensive job scheduling algorithms for healthcare cloud. *Computters and Electrical Engineering, 68*(March), 119-139. https://doi.org/10.1016/j.compeleceng.2018.04.001

Salim, B., Sherali, Z. and Abdelhamid, M. (2018). Fog computing job scheduling optimization based on bees swarm. *Enterprise Information Systems, 12*(4), 373-397. https://doi.org/10.1080/17517575.2017.1304579

Sarkar, S., Chatterjee, S. and Misra, S. (2018). Assessment of the Suitability of Fog Computing in the Context of Internet of Things. *IEEE Transactions on Cloud Computing, 6*(1), 46-59. https://doi.org/10.1109/TCC.2015.2485206

Seddik, Y. and Hanzálek, Z. (2017). Match-up scheduling of mixedcriticality jobs: ‘Maximizing the probability of execution jobs. *European Journal of Operational Research, 262*(1), 46-59. https://doi.org/10.1016/j.ejor.2017.03.054

Sheikhalishahi, M., Grandinetti, L., Guerriero, F., Wallace, R. M. and Vazquez-Poletti, J. L. (2015). Multi-dimensional job scheduling. *Future Generation Computer Systems, 54*, 123-131. https://doi.org/10.1016/j.future.2015.03.014

Shi, T., Yang, M., Li, X., Law, Q. and Jiang, Y. (2016). An energy-efficient scheduling scheme for time-constrained tasks in the local mobile clouds. *Pervasive and Mobile Computing, 27*, 90-105. https://doi.org/10.1016/j.pmcj.2015.07.005

Shinde, S. K. and Gawali, M. B. (2018). Task scheduling and resource allocation in the cloud using heuristic approach. *Journal Cloud Computing, 7*, 4. https://doi.org/10.1186/s13677-018-0105-8

Shojafar, M., Javanmardi, S. and Abolfazli, S. (2015). FUGE: The joint meta-heuristic approach to cloud job scheduling algorithm using fuzzy theory and the genetic method. *Cluster Computing, 18*, 829-844. https://doi.org/10.1007/s10586-014-0420-x

Skarlat, O., Nardelli, M., Schulte, S. and Dustdar, S. (2017). Towards QoS-aware Service Placement Fog. In: *Procedure of the First IEEE International Conference on Fog and Edge Computing, ICFEC ‘17*, IEEE. https://doi.org/10.1109/ICFEC.2017.12

Stavrinides, G. L. and Karatza, H. D. (2019). A hybrid approach to real-time scheduling IoT workflows in fog and cloud environments. *Multimedia Tools and Applications, 78*, 24639-24655. https://doi.org/10.1007/s11042-018-7051-9

Swaroop, P. (2019). Cost Based Job Scheduling In Fog Computing (PhD thesis), DTU, India, Available at: http://dspace.dtu.ac.in:8080/jspui/handle/repository/16722 (Accessed: 6 July 2020).

Tiwary, M., Puthal, D., Sahoo, K. S., Sahoo, B. and Yang, L. T. (2018). Response time for optimization cloudlets in Mobile Computing Edge. *Journal of Parallel and Distributed Computing, 119*, 81-91. https://doi.org/10.1016/j.jpdc.2018.04.004

Wang, X., Wang, Y. and Cui, Y. (2016). An energy-aware bi-level optimization model for multi-job scheduling problems under cloud. *Soft Comput., 20*(1), 303-317. https://doi.org/10.1007/s00500-014-1506-3

Yang, Y., Zhao, S., Zhang, W., Chen, Y., Luo, X. and Wang, J. (2018). DEBTS: Delay Balanced Energy Task Scheduling in Homogeneous Fog Networks. *IEEE Internet of Things Journal, 5*(3), 2094-2106. https://doi.org/10.1109/JIOT.2018.2823000

Zhou, B., Dastjerdi, A. V., Calheiros, R. N., Srirama, S. N. and Buyya, R. (2017). mcloud: The Context-Aware Offloading Framework for Heterogeneous Mobile Cloud. *IEEE Transactions on Services Computing, 10*(5), 797-810. https://doi.org/10.1109/TSC.2015.2511002

Zhou, X., Sun, M., Wang, Y. and Wu, X. (2015). The New QoE-driven Video Cache Allocation Scheme for Mobile Cloud Server. In: *Procedure of the 11th Conference on International Heterogeneous Networking for Quality, Reliability, Security and Robustness, QSHINE ‘15*, IEEE, pp. 122-126.

Zhu, C., Li, X., Leung, V., Hu, X. and Yang, T. L. (2015). Towards Integration of Wireless Sensor Networks and Cloud Computing. *IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, IEEE, Singapore, pp. 62-69. https://doi.org/10.1109/CloudCom.2015.27