INTERNATIONAL JOURNAL OF SOCIAL SCIENCE, INNOVATION AND EDUCATIONAL TECHNOLOGIES (ONLINE) - ISSN: 2717-7130

Vol:4 Issue: 16 pp: 264-275

JEL Codes: C18, C88

DUYUL, E., YAVUZ, M. O., DUYULER, E., BAYSAL, Y. (2023). "**Fuzzy Control Based Resource Planning for Fog Computing**" *International Journal of Social Science, Innovation and Educational Technologies (Online)*", Vol: 4 Issue: 16 pp: 264-275

Keywords: *fog computing, resource scheduling, task scheduling, fuzzy control*

Article Type Research Article

Fuzzy Control Based Resource Planning for Fog Computing

Arrived Date	Accepted Date	Published Date
15.09.2023	26.10.2023	31.10.2023

Emrah DUYUL[1] Muzaffer Orkun YAVUZ[2] Erbay DUYULER[3] Yusuf BAYSAL[4]

ABSTRACT

Fog Computing is adopted, which enhances the exchange of confidential information to distributed systems without interruption. Fog is identical to cloud computing; however, the only difference is that it is accessible to the data owners to manage and reply to the user in less time. Furthermore, it is useful for broadcasting applications in real-time, sensor networks, the Internet-of-Things demanding the maximum network performance, and the Internet's trustworthiness. One of the critical features of fog computing is resource distribution. In this paper, researchers introduced a lightweight of resources management for fog computing. Initially, the author normalizes the characteristics of resource management and normalize the data. Next, fog computing recommends a dynamic method for scheduling resources based on the fuzzy logic controller theory. Outcomes demonstrate that the proposed method enhances cloud computing's resources management efficiency and quality of service.

INTRODUCTION

A significant number of domains are used for cloud computing. However, with the invention of the Internet of Things, several more problems occurred in cloud computing. The delay-sensitive cloud computing and location-conscious applications with many distinct forms of IoT systems, on the one hand, cannot cooperate fully, and cloud computing applications, on the other hand, cannot be developed frequently because the design costs are too high.

The IoT devices increase significantly in sufficient quantity for information processing volume in the next few decades to be passed into data centers for data processing. (i.e., CISCO forecasts the annual global data center IP traffic of 50 billion connected devices to 15.3 ZB by the year 2020 (Cisco, 2015). If IoT always controls the considerable number of devices and data using the recent cloud computing

[1] emrahduyul@hotmail.com, Bakü / AZERBAIJAN

[2] orkun2019@hotmail.com, Bakü / AZERBAIJAN

[3] erbayduyuler01@gmail.com, Samsun /TÜRKİYE

[4] yusuf.baysal001@gmail.com, Muğla/ TÜRKİYE

http://dx.doi.org/10.54603/iss.194

approach, it caused delays and congestion in the computer system. Fog computing was recommended in 2012 by CISCO on the basic principle of such challenges Bonomi et al. (2012).

Industry 4.0 has digital prototype parts and components; many devices, like a mobile robot, or a chain drive, could also be used in the whole manufacturing process to start manufacturing each phase. In each of fog devices, the sensors are used to collect information on the status of the real-time basis and afterwards identify false alarms and exceptions. There is insufficient management bandwidth for existing automation systems; sometimes, there is even an absence of computational resources. To achieve supply chain response time, computationally intensive activities, for comparison purposes, in real-time state evaluation and fault detection are always provided to data centers. Network nodes perform together in the increased automation of the manufacturing process. Because one of them continues to fail, the whole production process is disrupted. For illustration, a mobile robot problem can occur in defective products or even prevent the manufacturing process's development without instant response.

Each device could perhaps access its sensor data to a high-performance cloud service to minimize the impact of hardware errors. On a server, the applications investigate information in real-time and instantaneously decide things that are suitable for a backup repository or alerting check and maintenance professionals. If fog computing could instantly forecast failure (i.e., the sensing data posted by the automaton is studied in virtual situations) and the respective maintenance people are made aware promptly, a derailment may be avoided during the manufacturing process.

Automation, networking, and storage resources are presented by Fog to the sensor that requires cloud services and can promptly solve problems. Different devices import sensor data to neighbouring fog nodes, sensing data from these fog nodes, and adapting to research findings. However, current fog nodes have restricted data analysis, network, and storage resources. A few other components were included in implementation in the fog environment. The FD components can perform a wide range of computations, filter data to eliminate inappropriate objects, and provide data with a label (Satyanarayanan, 2009). The FDs provide the resources for programming interfaces as Micro Data Centers (MDCs). The MDC will enhance resource productivity to optimize the distribution of computing components (CCs) to process modules (Gupta and Ghrera, 2016). The resource schedule in fog computing identifies resources in a given exchange to components. Because a fog design has more than enough resource management and components, the process is a critical, challenging issue. A percentage of QoS parameters Rodriguez and Buyya (2014) is the core of the allocation of resources.

This article provides a dynamic process of resources management predicated on the idea of a fuzzy logic controller. This research suggests the following findings as observes:

(1) A resource management system fog computing.

(2) A resource requirement method is implemented an essential role in the long and continuous available resources.
(3) In the resource development process, Fuzzy logic control methodology is introduced to determine the resource scheduling. The theory classifies users' requirements and resources in several resource clusters and user-friendly manufacturing technology, significantly improving the QoS for cloud computing reliability.
The rest of the paper is organized. Section 2 deals with the conventional practice of resources management and related research. The proposed model based on fuzzy Control is shown in Section 3. Section 4 provides an integrated dynamic strategy for time management. In section 5, experiments are performed, and the results are investigated. Eventually, the complete paper ends, and the possible improvements are described.

Related Work

Bonomi et al. (2012) which have introduced the fog compute economic model, have highlighted that fog computing is a virtual server service that delivers computation, networking, and storage providers between terminal devices and classical cloud computing network infrastructure at functional requirement specifies, commonly but not primarily. The investigation is Fog Computing, and various researches decided to focus on layout, architectural design, and management of resources. Based on the previous work Bonomi et al. (2011), Vaquero et al. (2014) have described fog computing as a sort of situation in which interconnected, and decentralized (Wi-Fi/perhaps autonomous) devices can communicate and possibly collaborate, during which a network performs scalability activities without the involving a third party. Activities can accept network layer functionalities and unique application services allowed to run in a virtualized setting, and benefits are provided to clients leasing parts of their devices to support these service providers. Several studies Bonomi et al. (2011), Li et al. (2018) have suggested applications to fog computing to comply with any of these classifications, which include home automation, smart cars, smart grid, wireless sensor networks. Moreover Mukherjee et al. (2017), Wang et al. (2015) illustrated real problems and limitations, including the availability of virtual technology, security and privacy problems, and access control.

Traditional programming methods are easy to use. Each provider offers a few heterogeneous VMs and global data-sharing cloud storage in designing multi-level, time-consisting empirical workflows Malawski et al. (2015). An initial overall time in which costs and cost-effective objectives are configured is a schedule changes technique. In Durillo and Prodan (2014) compared a business network as a cloud storage service. The Pareto Front has been used as a tool to help decide the trade between practical approaches. The planning costs were reduced by half, but the cost increases by 5%. Because of the best value point VM, this technique has a significant disadvantage. In Zhu et al. (2016) proposed network issues based real-time user experience scheduler. Basic principles in their algorithm are an opposite transformation, the boost of resource management, and the significant decrease in resource management use, as appeared differently concerning other significant techniques. In Poola et al. (2014) the fault-tolerant task routine is assumed by buying a strategic plan for spot VMs and the maximum time for on-demand cases. Fuzzy systems address things with a system of restrictions. The classical methodologies also include the first, best, and the worst fit. The cloud, fog, and edge providers Li et al. (2019) with heuristic algorithms. Pham and Huh (2016) the activities configured by genetic algorithms, which includes the cost of effort and engagement of processes. The outcomes emphasize the significance of the methodologies is better performance and reduction. In Sun et al. (2016) a cluster-based coalition game for fog-filled access networks is suggested. With the distributed user scheduling algorithm, it optimized the network performance. A dynamic algorithm Chen and Wang (2017) with queue length and throughput parameters is another method for a vehicular cloud system. The Markov server system has designed and represented activities on the stochastic Petri network and OpenStack.

In Zahaf et al. (2017) a probabilistic phase is introduced to improve power utilization by linear real-time work in a heterogeneous environment. In this investigation, dynamical programming is used to choose the frequency and to assign threads. The research schedules challenge in a cloud fog model was analyzed by Pham et al. (2016). A hash function was recommended to balance the optimum time of completion and cloud services' resource limitations. Zeng et al. (2016) viewed an efficient approach represented by fog development tools and tested different tasks connectivity problems. These significant issues were designed as a non-linear web development multi-integer challenge to achieve better knowledge and experience, and the study proposed a cost-effective multi-objective evolutionary business model for processes. In fog computing devices based on Priced Timed Petri Nets (PTPN), Ni et al. (2017) recommended allocating resources strategic plans.

Furthermore, this PTPN task approach was established depending on the characteristics of fog resource management. The algorithm proposed can be significantly greater than static application methods, along with a particular fog computing computer system, some of the above data analysis resource allocation mechanisms. Moreover, a few researchers have incorporated optimization algorithms for resource management. In Luo et al. (2012) investigators have been using fuzzy cluster analysis to improve resource management in complex hardware/software. The system has been evaluated in a designed physical model and can reliably plan task resources. In Rattanatamrong and Fortes (2014) resource management's objective function for real-time ensemble systems was suggested as a faulty reasoning feedback scheduler. It presented all the meta-analyses in virtual environments while significantly reducing the system of available resources and complexities. Masmoudi and Hait (2013) suggested a technique that would ensure uncertainty at all phases of developing the modelling techniques and finding solutions protocol by analyzing fuzzy modeling work of the challenge control triggered by the fuzzy business model. The process is being used for the servicing of civil-military planes and was successful.

The abovementioned surveys effectively improve resource management from distinct viewpoints through alternative techniques. Clients enhance the accuracy of the execution of tasks. Standard QoS requirements such as cost, Bandwidth, and completion times are encountered, but there is no need to consider some extended QoS specifications like system friendliness and user interface. But since these enhanced QoS requirements play a significant role for a client in choosing if several participants are typically enough to implement cloud computing correctly in maintaining with the challenging behaviour. They often develop an erroneous demand based on the complexity of the job. The way to address such fluid requirements is a crucial illustration of cloud computing friendliness. The research aim of this section was about how to focus on improving system friendliness. The paper also presents the flowing logic and feedback control system into the cloud computing strategic plan for planning resources and conceptual frameworks the user model and resource allocation method.

The Proposed Approach

System Model

Fog Computing Design has a typical example of cloud, edge, and sensor nodes. Sensors are situated at the architectural design base in the reduced geographical area and then forward collected data via gates to the highest management. Productive bars at the initial stages of the fog production process describe or reconfigure the environment. Throughout this fog network process, data is sensed and distributed, and fog devices are transmitted, a request is categorized into many components, and resources are allocated for their implementation. Applications are being used to collect and analyze MDC information and evaluate and methodology data sets.

The data received is processed or sent to the cloud in Figure 1. The fog devices and unique products demonstrate this. An MDC that detects, sensors and stores the data received from the sensors is a fog device. The fog device features encompass a million instructions per second (MIPS), RAM, throughput, frequency band down, route level number, MIPS level, authority in the busy scenario, and stagnant energy. Including anchors as {$Anchor_1$, $Anchor_2$,…,$Anchor_n$}, each fog device. The features of an anchor include RAM, Bandwidth, storage, and Computer Components (CC).

$$FB_{Low} \leq \sum_{i=1}^{N} AB_i \leq FB_{High}$$

where,

FB - Bandwidth of Fog Computing

AB - Bandwidth of Anchors

FB_{Low} – Lower Bound of Bandwidth

FB_{High}- Upper bound of Bandwidth

AB_i - i^{th} Anchor Bandwidth

N - Sum of Anchors

The overall Bandwidth of all anchors in individually fog device is between FB_{Low} and FB_{High}. CC of anchors is assigned and performed with fog nodes to software modules. MIPS is the primary component in a CC. The significance sets for all fog node at the beginning of the procedure. The combined granted MIPS of all CCs is verified after a CC is assigned to a control system application.

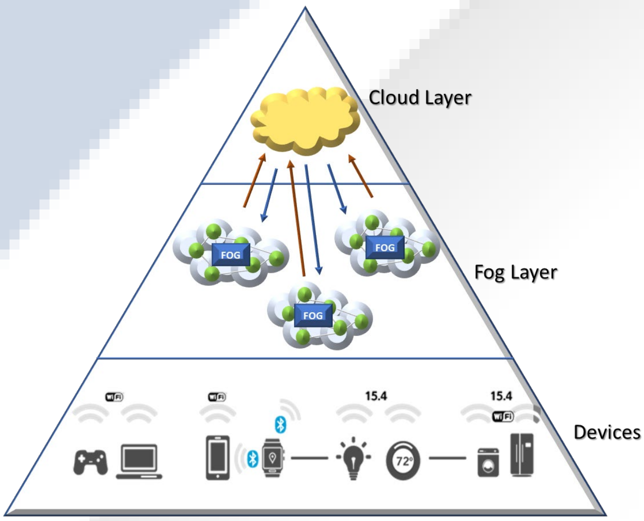
Thus, we have $CGM = \sum_{i=1}^{N} \sum_{j=1}^{M} CEM_{i,j}$, where CGM is the combined granted MIPS of a fog device that is <= to MIPS of that fog device ($CGM \leq FDMIPS$), and CEM is the CC's MIPS.

N - *Number* of Anchors

M - Number of CCs in an Anchor

$CEM_{i,j}$ - MIPS of j^{th} CC in i^{th} Anchor

The application module is a type of VM. The MIPS Properties of the configuration also provide MIPS, size, Bandwidth, and percentage of the CCs properties to cooperate. In each fog device, the simulation time is significantly higher than the CCs. In fact, $\sum_{i=1}^{C} Module_i > \sum_{j=1}^{K} FD_j$, where C is the overall number of elements, and K is full of fog devices.



Model of User Entity

The client is the entity that rents cloud storage by threshold and problem to solve confirmation. The user settings can be shown as $Users = \{U_1, U_2 ... U_m\}, U_i$ The following six tuples can be highlighted as a service attribute that owned the cloud

$$U_i =< U_{ID}, U_{Name}, USL, U_{Jobs}, U_{Pos}, U_{Sec} >$$

U_{ID} - Cloud Services User Entity Tag for CSP

U_{Name} - Friendly classification of the User Entity

USL - Level of User Service

SL - Service Level

$SL = \{Tier\ I, Tier\ II, Tier\ III\}$ Criteria for the user's billing tasks.

U_{Jobs} - Collection of the task request and a collection of the job. Identify the user's location

U_{Sec} - Level of security of user data has been mentioned

Model of Task

Also, objectives are identified to characterize the task model: Parallel function by several jobs to interact regularly. Each activity controls the virtualization resources. To verify that all activities in the schedule are performed, preventing the work component and reducing the energy cost. The tasks introduced are being exposed $Tasks = \{Task_1, Task_2 \dots Task_n\}$, $task_t$ can be exposed by the subsequent tuple:

$$< T_{ID}, T_{Type}, T_{Len}, T_{Res}, T_{ART}, T_{AWT}, T_{Tasks} >$$

T_{ID} - Unique Value of $Task_t$

T_{Type} – Classification of j#obs

$$T_{Type} = \{Small \mid T_{Len} \in [0, Max - 1]\},\ T_{Type} = \{Large \mid t \in [Max]\}$$

$T_{Len} = |t_{tasks}|$ Job Length

T_{Tasks} – Set of Tasks in lively, $T_{jobs} = \{Job_1, Job_2, \dots Job_k\}$

T_{Res} - Minimum Set of Resources

To study process cost and presentation, the limitation t_{art} Symbolized job's Average Response Time that is the time from the job into the job scheduling line to execution. Displays job's Average Wait Time, that is, the time since the job into the scheduling line to activate the line.

Model of Resource Management

Computational and storage managements are managed and configurable (e.g., networks, servers, storage, etc.).

Model of Resource Management demonstrations $Resource = \{R_1, R_2, \dots rR_p\}$, resource total number $P = |Resource|$, $_k$ is the following tuples:

$$R_k = < R_{ID}, R_{Supply}, R_{Type}, R_{Comp}, R_{Mem}, R_{Stor}, R_{IO}, R_{Net}, R_{Pos} >$$

R_{ID} —Unique Resource ID

R_{Supply} - Resource's Provider

R_{Type} – Type of Resource

R_{Comp}-The processing Volume of Resources

R_{Mem} - Availability of Memory Management

R_{Stor} - Storage Size of resources

R_{IO} - IO Size

R_{Net} - Bandwidth

R_{Pos} — Information Geographical Position

Resource Scheduling Algorithm Design

The resource index in the development process is significantly decreased after a sensible division of fog resource management. The specifications of user groups can be divided into the following classifications. The primary users are linked with resource management in the class after searching for the proper resource classification. To complete the resource schedule, this article utilizes minimal weight interaction to reach it.

$$Rank = Total\ (Result\ (Req_i - Rset_i\ ) * \text{Weight}_i\ )/\text{Total}\ (\text{Weight}_i\ )$$

Where,

Req_i represents the attribute of the user's needs,

Res_i represents the attributes of the resources, and

Weight_i represents the weights of the attributes

Varying resource requirements for multiple users. They can be divided into computation requirements, bandwidth needs, and memory size for different task desires. Each phase involves, like resource management, three attributes, each takes a unique weight. The element needed by the system and the resource attribute together are approximated in the above methodology, and the user is assigned the most significant score in the resource routine.

The resource sequence-dependent setup pseudo-code for Fuzzy Control (FRSFC) Method Fog resource management resource allocation is as shown in:

Algorithm 1: FRSFC

Input: Resource Set $\{Rset1, Rset2, \ldots, Rsetm\}$, Job Set $\{Job1, Job2, \ldots, Jobn\}$, n,φ, M

Output: Ranking, Similarity Result

1: Process the Original Data

2: Perform the Resource Clustering on Resource Set

3: RConRS $(M, L) \rightarrow$ *User* and Resource Clusters

4: *Res (s)*=size(R)

5: U = RConRS(m)

6: **Do Until Test < 1**

7: $Test \leftarrow Test + 1$

8: $x^J \leftarrow x$

9: Process xn

10: **For** $l = 1 \rightarrow l$ **Do**

11: **For** $j = 1 \rightarrow m$ **Do**

12: Space $(k, i) = Sp$ (Info $(j, :)$

13: **End**

14: **End**

15: Measure x^J

If Result $(x^{J} - x, \text{data}) < \varphi$ **Then**

Break

End If

$x \leftarrow x^{J}$

End While

Obtain Resource Categories

Measure Rank $(Req_i, Rset_i, \omega_i)$

$Rank = Total\ (Result(req_i - Rset_i) * \text{Weight}_i) / \text{Total}\ (\text{Weight}_i)$

Return Similarity Measure S

Experimental results from Fog Resource Scheduling using Fuzzy Control (FRSFC)

Researchers compare the resource distribution level of various methodologies with the reliability of FRSFC's allocation of resource management. Authors assign goals for different lengths of activity to various virtual servers here. Figure 2 illustrates the velocity with which the CPU is being executed on many edge machines. The figure represents the FRSFC schedule to accommodate the available resources to correlate connected devices, based on the activities' length. If the task length is longer, the virtual machine's implementation speed boosts.

The phenomenon is unique from other methodologies because when ALGM 3 and ALGM 2 could adapt faster as FRSFC to the tasks' duration. In order to have access, the ALGM 1 could perhaps react to the specific features. When five virtual machines exist, the following figure 3 shows the work time using new objectives. The period to perform the job is the maximum when ALGM 1 is accepted. The possible explanation for this is that activities are distributed orderly to virtual servers and trigger the lack of supply and wholly inadequate resource management use. ALGM 2 enhances the efficiency of the task and reduces the execution time compared to ALGM 1. Nevertheless, processes are not identified, resulting in simplistic activities and resources being improperly used following best practice virtual machines.

When ALGM 3 is decided to adopt, the processing system's resources are incorporated in step with the length and the aggregate processing time problem significantly reduces between virtual machines. The execution time will be reduced when FRSFC is decided to adopt. The reason behind it is that FRSFC adopts automatically manages the resource availability between virtual machines; however, according to demand, it significantly decreases the computational cost and dramatically reduces the timeframe between computers.

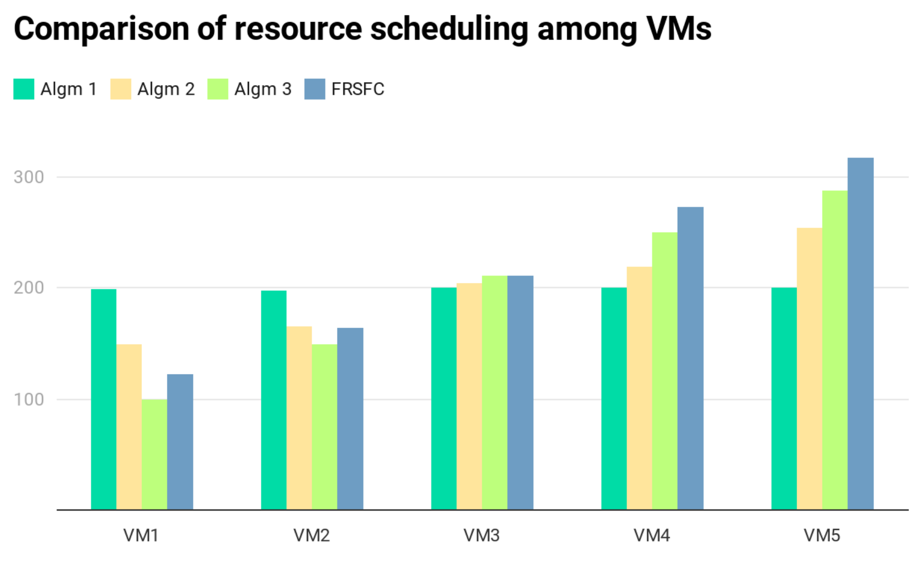


Figure 2. Running CPU speed on respective virtual servers

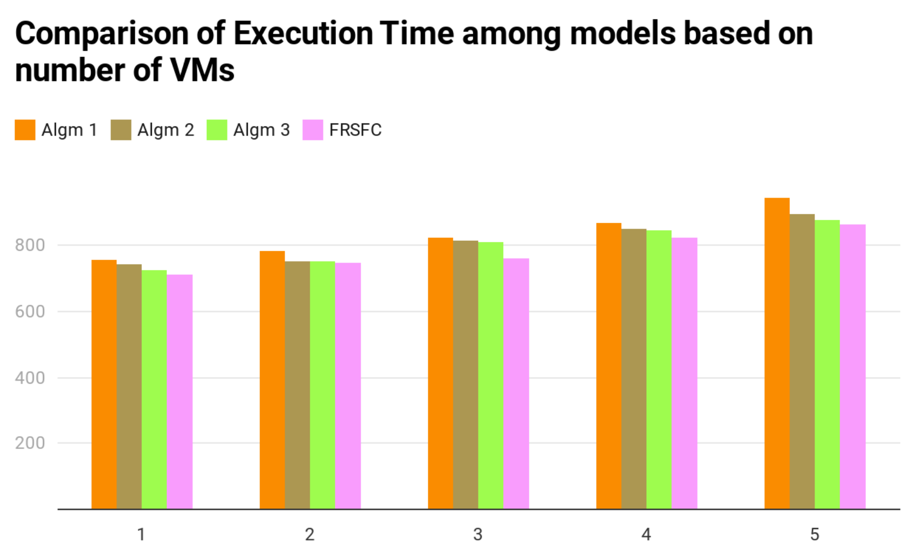


Figure 3. Task time using practical scenarios if five VMs are configured

This article identified the performance of resources management in fog computing. Initially, researchers clustered the fog resources, which mostly reduces the array of the analysis phase. The FRSFC technique is also recommended to find out the scheduling of resource management. Ultimately, from the experimental investigation, a higher accuracy rate than that of the other algorithms was seen in this manuscript. Moreover, the FRSFC suggested technique may faster match user requests and significantly improve data quality with developed management classifications. Authors will understand dynamic resource transformation in future work and introduce a suitable timetable strategy to improve resource use and resolve user issues.

Acknowledgment: The authors have not received financial support from the University or any other institution/organization. The authors are grateful to the journal's anonymous reviewers for their extremely helpful suggestions to improve the quality of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

Bonomi, F., Milito, P., Natarajan, Zhu, J. (2011). "Fog computing: A platform for internet of things and analytics," Big Data and Internet of Things: A Roadmap for Smart Environments, pp. 169–186. https://cse.buffalo.edu/faculty/tkosar/cse710_spring19/bonomi-bdiot14.pdf

Bonomi, F., Milito, R., Zhu, J., Addepalli, S. (2012). "Fog computing and its role in the internet of things," *Mcc Workshop on Mobile Cloud Computing ACM*, pp. 13–16. https://conferences.sigcomm.org/sigcomm/2012/paper/mcc/p13.pdf

Chen, X, Wang, L. (2017). Exploring Fog computing-based adaptive vehicular data scheduling policies through a compositional formal method—pepa. IEEE Communications Letters, *21*(4): 745-748. doi: 10.1109/LCOMM.2016.2647595

Cisco (2015). "Cisco global cloud index: Forecast and methodology, 2015–2020," Cisco, [A1]

Durillo, JJ., Prodan, R. (2014). Multi-objective workflow scheduling in amazon ec2. *Cluster Computing,* 17(2): 169-189. doi: 10.1007/s10586-013-0325-0

Gupta, P., Ghrera, SP. (2016). Trust and deadline aware scheduling algorithm for cloud infrastructure using ant colony optimization. In: *International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH)*, Noida, India, pp. 187-191.

Li, C., Bai, J., Tang, J. (2019). Joint optimization of data placement and scheduling for improving user experience in edge computing. *Journal of Parallel and Distributed Computing,* 125: 93-105. doi: 10.1016/j.jpdc.2018.11.006

Luo, M., Zhang, K., Yao, L. & Zhou, X.(2012). "Research on resources scheduling technology based on fuzzy clustering analysis," in *Proceedings of the 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD '12)*, pp. 152–155, May 2012.

Malawski, M., Figiela, K., Bubak, M., Deelman, E., Nabrzyski, J. (2015). Scheduling multi-level deadline-constrained scientific workflows on clouds based on cost optimization. Scientific Programming 2015, 1-13. doi: 10.1155/2015/680271

Masmoudi, M. & Hait, M. (2013). "Project scheduling under uncertainty using fuzzy modelling and solving techniques," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 1, pp. 135–149.

Mukherjee, M., Matam, R., Shu, L., Maglaras, L., Ferrag, M. A., Choudhury, N., Kumar, V. (2017). "Security and privacy in fog computing: Challenges," IEEE Access, vol. 5, no. 19293-19304. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8026115

Li, Z., Chang, B., Wang, S., Liu, A., Zeng, F., Luo, G. (2018). "Dynamic compressive wide-band spectrum sensing based on channel energy reconstruction in cognitive internet of things," IEEE Transactions on Industrial Informatics, *(14)*6, 2598–2607.

Ni, L., Zhang, J., Jiang, C., Yan, C., Kan, Y. (2017). Resource allocation strategy in fog computing based on priced timed petri nets. IEEE Internet Things J., 4, 1216–1228.

Pham, XQ, Huh, EN. (2016). Towards task scheduling in a cloud-fog computing system. In: 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), Kanazawa, Japan: IEEE, pp. 1-4.

Poola, D., Ramamohanarao, K., Buyya, R. (2014). Fault-tolerant workflow scheduling using spot instances on clouds. *Procedia Computer Science*, 29, 523-533. doi: 0.1016/j.procs.2014.05.047.

Rattanatamrong, P. & Fortes, J. A. B. (2014). "Fuzzy scheduling of real-time ensemble systems," in *Proceedings of the International Conference on High Performance Computing and Simulation (HPCS '14)*, pp. 146–153, IEEE, Bologna, Italy, July 2014.

Rodriguez, MA., Buyya, R. (2014). Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. *IEEE Transactions on Cloud Computing*, 2(2): 222-235. doi: 10.1109/TCC.2014.2314655.

Satyanarayanan, M., Bahl, V., Caceres, R., Davies, N. (2009). The case for VM-based cloudlets in mobile computing. *IEEE pervasive Computing*, 8(4): 14-23. doi: 10.1109/MPRV.2009.82

Sun, Y., Dang, T., Zhou, J. (2016). User scheduling and cluster formation in fog computing-based radio access networks. *IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB),* Nanjing, China, pp. 1-4.

Vaquero, L. M. & Rodero-Merino, L. (2014). "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM Sigcomm Computer Communication Review, (44)*5, pp. 27–32.

Wang, Y., Uehara, T., Sasaki, R. (2015). "Fog computing: Issues and challenges in security and forensics." Taichung, Taiwan: IEEE Computer Software and Applications Conference.

Zahaf, HE., Benyamina, AEH., Olejnik, R., Lipari, G. (2017). Energy-efficient scheduling for moldable real-time tasks on heterogeneous computing platforms. *Journal of Systems Architecture*, 74: 46-60. doi: 10.1016/j.sysarc.2017.01.002

Zeng, D., Gu, L., Guo, S., Cheng, Z., Yu, S. (2016). Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system. IEEE Trans. Comput., 65, 3702–3712.

Zhu, X., Wang, J., Guo, H., Zhu, D., Yang, LT, Liu, L. (2016). Fault-tolerant scheduling for real-time scientific workflows with elastic resource provisioning in virtualized clouds. IEEE Transactions on Parallel and Distributed Systems, 27(12), 3501-3517. doi: 10.1109/TPDS.2016.2543731. https://ieeexplore.ieee.org/ielaam/71/7740988/7435325-aam.pdf?tag=1