Journal of Ubiquitous Systems & Pervasive Networks
Volume 13, No. 1 (2020) pp. 11-19

QoS-Aware Placement of Tasks on a Fog Cluster in an Edge Computing Environment

Elarbi Badidi *

Department of Computer Science and Software Engineering,
College of Information Technology, UAE University, AL-AIN, PO Box. 15551, UAE

Abstract

The advances made in the sensing and communications technologies over the last few years have made the deployment of IoT solutions possible on a massive scale. The wide deployment of IoT sensors and devices has resulted in the development of smart services that were not possible before. These services typically rely on cloud services for processing IoT data streams, given that edge devices have limited computing and storage capabilities. However, time-sensitive IoT applications and services do not tolerate the high latency they can encounter when sending IoT data streams to the cloud. Fog computing-based solutions for these services are increasingly becoming attractive because of the low latency they can guarantee. With increasing deployments of fog nodes and fog clusters, we propose an architecture for the placement of IoT applications tasks on a cluster of fog nodes in the vicinity of the application's data sources. The Fog Broker component can implement various scheduling policies to help IoT applications meet their quality-of-service (QoS) requirements. Our simulations show that it is possible to maintain low application latency and distribute the load between the fog nodes of the cluster by using simple scheduling strategies.

Keywords: *Fog computing, cloud computing, quality-of-service, latency, scheduling*

1. Introduction

In recent years, the Internet of Things (IoT) has proliferated at an unprecedented rate. IoT devices and sensors are deployed in many areas such as factories, manufacturing plants, smart cities, healthcare facilities, transportation systems, and more.

We expect all of the objects around us to be embedded with sensors and IoT devices. These devices generate large amounts of data while continuously monitoring machinery and equipment in manufacturing plants, road traffic conditions and availability of parking spaces in cities, weather conditions, consumption of energy and water in buildings and facilities, safety in public spaces, the structural integrity of bridges, and much more. The massive amount of data collected must be analyzed to help and speed up decision-making processes, improve services, and reduce costs.

More precious than ever, data is now considered the main product of the digital economy. Organizations and businesses looking to gain a competitive advantage are investing heavily in IoT solutions to capture and store massive amounts of data that can help them obtain insights from collected data. These insights would help them optimize processes, develop new products, and detect anomalies before they occur. The challenge lies in their ability to use data streams that are still in motion and to react quickly to critical events before even storing the data in the cloud for further processing and analysis.

In addition to the growing deployment of IoT, new technologies in advanced analytics and artificial intelligence promise to take IoT-based monitoring to another sophisticated level. Leveraging IoT-based data streams means increased visibility into the events taking place across edge devices and continuous development of smart services. Since edge devices generally do not have the computing and storage capabilities for local processing of the generated data, the traditional approach is to transfer data to the cloud, which has abundant processing and storage resources. This approach has proven costly in terms of latency, network bandwidth, storage, and energy consumption. To solve these problems, a new paradigm called "fog computing" was first proposed by Cisco systems [13]. This paradigm is not a replacement for cloud computing but a complementary computing model. It promotes the idea of moving specific IT and storage capacities to the edge of the network and near the data sources. An essential benefit of fog computing is the reduction in the amount of data that has to be transmitted and stored on the cloud. We discuss other benefits of fog computing in the background section.

A growing number of studies have investigated the issues of resource allocation and service scheduling in fog computing environments [26][32][27][15][2][5]. Most of these investigations studied the scheduling of tasks for execution on a single fog node or edge device, or cloud. The low latency requirement of time-sensitive IoT applications requires finding

* Corresponding author. Tel.: +971 3 7135552
Fax: +971 3 713 6901; E-mail: ebadidi@uaeu.ac.ae

DOI: 10.5383/JUSPN.13.01.002

suitable fog services that provide the service with the required QoS. In many scenarios, several fog nodes may be deployed near IoT devices, such as in a smart road intersection. Therefore, it is essential to choose appropriate fog nodes for these applications based on the current state of their available resources to meet the QoS requirements of IoT applications.

In this work, we address this issue by proposing a broker-based architecture that uses a fog broker component to schedule the different tasks (of time-sensitive and non-time-sensitive applications) on the available fog nodes. The Fog Broker's components receive requests from various applications and collect up-to-date information on the available resources of the fog nodes of the cluster in order to come up with a schedule for the different tasks. It can implement various scheduling algorithms. As a proof of concept, we consider a scenario of a fog cluster with five fog nodes, and we use the CloudAnalyst simulation tool and three scheduling strategies implemented by CloudAnalyst to assess the performance of the task placement on the fog cluster resources [4].

The main contributions of this paper are summarized as follows:

1) the design of a fog broker-based architecture for the placement of IoT applications' tasks in a fog cluster.
2) A review of some existing works on the issues of resource provisioning and task scheduling in fog computing. Our work differs from existing works, described in the related work section, in that it investigates QoS-aware tasks placement in a fog cluster using a scheduler-workers model. Each fog node of the cluster has several virtual machines for processing tasks and uses a load balancing policy.
3) The proposed model's performance was assessed using simulation with three scheduling strategies and three load balancing policies. The results show that the best results in terms of latency, average fog node processing time, and balancing the fog nodes load are obtained using the Throttled load balancing policy irrespective of the scheduling policy. To the best of our knowledge, none of the existing works assessed the three scheduling policies' performance together with the load balancing policies in a fog computing environment.

The remainder of the paper is organized as follows: Section 2 presents background information on edge and fog computing. Section 3 describes related work on the resource provisioning and scheduling of applications on the edge, fog, and cloud. Section 4 describes our proposed architecture for tasks' placement on a fog cluster. Section 5 details the task scheduling and placement problem. Section 6 describes the results of the simulations we did using the CloudAnalyst simulation tool. Finally, Section 7 concludes the paper and highlights future work.

2. Background

2.1. Edge and Fog Computing

Edge computing brings data storage and computing closer to the edge devices that generate data rather than relying on remote cloud servers that are far away. It allows businesses to save money by reducing the amount of data that needs to be transferred and processed on the cloud [20]. Many firms that have adopted cloud-based solutions for many of their applications have realized that bandwidth costs are higher than expected. Real-time processing of generated data would allow time-sensitive IoT applications to avoid latency issues that can affect application performance. For example, to monitor road traffic conditions multiple Internet-connected video cameras deployed at different locations in the road send their live footage for processing. Data generated by a single video camera can be transmitted easily over a network. However, many problems arise when live recordings from multiple video cameras are transmitted simultaneously. High latency can affect the monitoring application response time, and the bandwidth costs can be high. With an edge computing-based solution, the application could run locally on an edge server or gateway for fast processing and response. Other applications such as autonomous cars, virtual and augmented reality, smart healthcare systems have similar requirements in term of latency. Figure 1 depicts the edge-fog-cloud three tiers architecture for managing and processing IoT data.

Some solutions, such as micro-data centers and cloudlets [19], [20], and fog-based solutions have been developed and deployed on the edge of the network to address the ineffectiveness of cloud computing in data processing for time-sensitive IoT applications. Bonomi et al. from Cisco Systems defined Fog computing as: "*Fog Computing is a highly virtualized platform that provides compute, storage, and networking services between end devices and traditional Cloud Computing Data Centers, typically, but not exclusively located at the edge of the network.*" [13].

The earlier literature treated fog computing and edge computing interchangeably as they both deal with processing IoT data close to data sources before transferring it eventually to the cloud. With edge computing, data is processed in edge devices and transmitted, using the communication capabilities of edge gateways, to a fog node or cloud server. With fog computing, the data is processed in a fog node located near the local network. Additional fog nodes can be deployed when more computing power is needed to improve scalability and provide the elasticity required by several business applications.

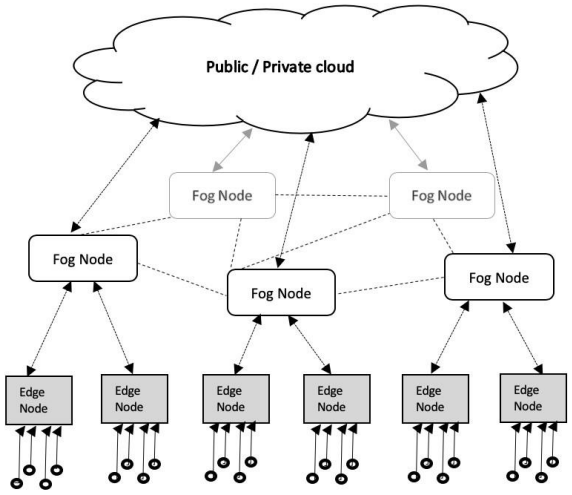

Fig. 1. Edge, fog and cloud layers.

2.2. 5G and Fog Computing

The 5G technology promises to boost the cellular network not only to interconnect people but also to interconnect and control machines, things, and devices. The main features of 5G networks are enhanced mobile broadband, massive machine-type communications, and ultra-reliable and low-latency communications, which are expected to foster a wide range of high-performing applications and services [33]. 5G will provide new

performance levels and efficiency that will allow new user experiences and new industries to connect. The 5G will offer multi-Gbps data rates, ultra-low latency, huge capacity, and more consistent user experience.

The significant value that 5G brings to fog computing is the large number of simultaneous devices supported by 5G cells compared to 4G. For instance, using 5G in smart city scenarios means that the smart city's components can be connected autonomously, sharing data, aggregating it, and exploiting it in real-time. Santos et al. [34] proposed a fog computing-based framework to manage and orchestrate smart city applications in 5G wireless networks. The framework aims to provide low latency, high energy efficiency, and high mobility to overcome IoT scenarios' strict requirements. It allows for autonomous capabilities for resource provisioning in 5G-enabled smart cities. Van Lingen et al. [35] discussed what they call "The Unavoidable Convergence of NFV, 5G, and Fog" and described an architecture that addresses some of the central challenges behind the convergence. The proposed solution was applied to a smart city use case.

2.3. Fog Nodes

The main components of fog computing infrastructures are fog nodes, which can be physical or virtual components connected with smart IoT end devices and access networks to provide data storage and computing resources to these devices. Physical fog nodes include gateways, routers, switches, and servers. Virtual nodes can be in the form of virtual machines and cloudlets. For instance, Cisco fog nodes include wireless access points, switches, routers, and the Cisco Unified Computing System (UCS) servers. These servers typically include computing hardware, virtualization support, switching fabric, and management software. The user can develop and deploy IoT applications on the cloud or the fog [6]. Fog nodes can operate in a stand-alone mode or be part of a fog cluster with several fog nodes to provide their services to the users. They can be deployed anywhere with a network connection. Several works described the design and implementation of fog nodes [31] [1][28].

2.4. Fog computing to support smart services

Fog computing promises to play an important role in the implementation of smart applications and services. For instance, smart cities can use fog computing for transportation applications, by obtaining data on current road and intersection conditions, and security applications by using video surveillance. Data management and processing operations such as data collection, aggregation, and rich and advanced analyzes that involve machine learning and event processing can be performed in one or more fog nodes at the edge of the network. The main benefits for organizations that use fog computing are bandwidth optimization, traffic reduction, reduced latency, and improved privacy and security.

- *Bandwidth optimization and Reduced traffic*. Organizations can reduce network traffic and optimize bandwidth usage if they are equipped to process and analyze the large amount of data generated by IoT devices locally. In doing so, only aggregated and compiled data is sent regularly to a cloud server for further analysis. Fog analysis can be supported by machine learning and artificial intelligence algorithms. Many modern devices already support artificial intelligence algorithms. For example, edge accelerators, industrial robots and delivery drones with computer vision capabilities.
- *Reduced latency*. Fog computing-based solutions store and process IoT data streams locally, allowing applications to get faster responses and reduce network traffic. This is vital for many scenarios, such as intelligent transportation systems, logistics and supply chain management, and emergency response.
- *Enhanced privacy and security*. Fog computing reduces data propagation by analyzing sensitive data in fog nodes instead of a remote data center outside the control of the organization. This can greatly improve the privacy of the data.

With the increasing power and resources of fog nodes, organizations can acquire, aggregate, and process IoT data streams to meet the stringent quality of service of time sensitive IoT applications. However, a right balance between cloud and edge/fog computing is required since the cloud has more storage and processing resources than fog nodes.

3. Related Work

Since IoT devices typically have limited compute and storage capacity, computationally demanding real-time IoT applications are typically offloaded to fog nodes with significant compute and storage resources. The QoS they provide to applications depends very much on how fog resources are managed and provisioned. Resource provisioning has been widely investigated in distributed environments in general and in cloud computing in particular [29][22]. Resource provisioning in fog computing shares with resource provisioning in cloud computing similar concepts, challenges, and research problems. As fog landscapes are generally more volatile than cloud environments, the existing approaches to resource provisioning cannot be applied to fog computing without adaptation. Therefore, more contextual information about fog landscapes and edge devices should be used to adapt effectively to the dynamic change in IoT environments. A growing number of studies have recently studied the problem of resource provisioning in fog nodes.

Skarlat et al. [26] proposed a conceptual framework for fog resources provisioning that aims to distribute task requests and data between fog nodes. The framework includes a hierarchy of fog colonies, each with a fog orchestration control node that controls several fog cells (fog nodes). The fog orchestration control node's role is to monitor resource utilization within its fog colony and create a provisioning plan for task requests. The authors formalized the resource provisioning problem as an optimization problem capable of considering existing resources in the fog landscape. The authors stated that the results of their resource provisioning model show a reduction in delays of up to 39% compared to a reference approach, which results in round trip times and shorter time intervals.

Yao et al. [32] considered that a fog node that relies on cloud technologies could rent and free virtual machines in an on-demand manner. The QoS (i.e., task completion time) offered to mobile IoT devices with limited power may be degraded due to varying conditions of the wireless channel. The authors investigated how to jointly optimize fog resources provisioning (i.e., decisions on the number of virtual machines to rent) and the power control to help maintain the wireless transmission rate and therefore enhance the QoS. They formulated the joint optimization problem– which aims to minimize the cost of the system (VM rentals) while ensuring QoS requirements– as a

non-linear mixed integer programming problem, and proposed an approximation algorithm to solve the problem.

The authors of [16] proposed a QoS-aware network resource management framework for containers in fog nodes. Fog nodes are increasingly using containerization to achieve low overhead for resource-limited fog devices such as WiFi access points and set-top boxes. However, the control of network bandwidth for outbound traffic is a weakness of containers, representing a challenge in using containers in fog nodes. The failure to achieve desirable network bandwidth control by existing container solution make bandwidth-sensitive applications suffer unacceptable network performance. The proposed framework aims to limit the rate of outbound traffic in fog nodes. It supports three scheduling policies, proportional share scheduling, minimum bandwidth reservation, and maximum bandwidth limitation, which can be applied to containers simultaneously.

In computing, scheduling represents the process of allocating computing resources to an application and mapping its tasks to those resources to meet certain quality of service (QoS) and resource conservation objectives. The general problem of mapping tasks to distributed resources belongs to the class of NP-hard problems [8]. No known algorithm has ever found the optimal solution in polynomial time for this mapping problem. As the exhaustive search solutions are not practical due to the extremely high cost of generating schedules, many algorithms based on heuristics and metaheuristics have been proposed to schedule the tasks of applications in heterogeneous distributed systems environments. The scheduling of applications on the edge, fog, and cloud servers consists of placing the logical components of the application on the resources available on those three layers for execution, deciding on their interactions in this compute and network hierarchy to meet their QoS requirements. As edge resources and fog layers can be mobile, applications can also impose requirements on the mobility of processes and data. Besides, changes in data generation rates, network behavior, or battery energy levels may require reactive strategies. The application must be scheduled and coordinated to meet a variety of quality of service objectives, such as latency, energy, and monetary constraints.

Over the last two decades, there has been much work on scheduling approaches for clouds and clusters. However, given that edge and fog computing are emerging paradigms, there are a few studies concerning application distribution and scheduling on the edge and fog resources individually and in clouds. The existing literature has proposed the conceptual underpinnings of edge and fog computing [13][10][24]. Others discussed the benefits and challenges of coordinating the edge, fog, and cloud layers in a hierarchical model [30][25] but did not examine in detail their impact on applications and their schedule. There is a growing number of research works, which proposed scheduling approaches to assign application tasks to edge, fog, or cloud to meet the application QoS requirements [27] [15] [2] [5].

Bayer et al. [27] studied how to provide a consistent environment for combining Fog and Cloud Computing using container virtualization. The architecture focused on the development of new deployment algorithms for the distribution of services between work nodes. The paper described how to integrate centralized and decentralized measurement probes to collect node conditions, such as available network connection, performance, and storage capabilities, as well as security and privacy controls. Haruna et al. [15] investigated the problem of user mobility and allocation of resources in the edge/fog and cloud data centers. They proposed an algorithm that works with the Seamless Handover Scheme for mobile IPV6 and suggested scheduling policies to mitigate the user mobility challenge to reduce application latency.

Singh A. et al. [2] studied the issue of load balancing in a fog computing environment while taking into account the jobs' security constraints on different fog nodes. They proposed three heuristic algorithms: minimum distance (MD), minimum load (ML), and minimum hop distance and load (MHDL). Also, they proposed an Integer Linear Programming-based algorithm to solve mapping and scheduling problems. They compared the performance of their proposed algorithms to the cloud only algorithm and to another heuristic algorithm called fog-cloud-placement (FCP). The authors stated that the results of the simulation they performed revealed that the MHDL heuristic works better than other scheduling policies in the fog computing environment while meeting application privacy requirements.

Rahabri D. et al. [9] proposed a Knapsack-based scheduling algorithm (GKS) to allocate the appropriate resources to modules in a fog network. The algorithm was simulated using iFogSim [14]. The authors stated that the results obtained in terms of the execution cost, energy consumption, and sensor lifetime show that their proposed algorithm outperforms the first-come-first-served, concurrent, and delay-priority algorithms.

4. Architecture for task placement on a fog cluster

Figure 2 depicts our proposed architecture for QoS-aware task placement on a fog cluster. The main components of the architecture are Edge Gateways, the Fog Broker, the cluster of Fog nodes, and applications.

4.1. Edge gateways

IoT gateways are fundamental components in IoT deployments. They have become essential architectural components that improve the performance of IoT solutions. Many solution suppliers have opted for ready-to-use devices that are flexible enough to meet the different demands of individual IoT deployments. An IoT gateway is typically responsible for aggregating the data received from IoT devices and sensors, translating the sensors' protocols to guarantee their interoperability, and preprocessing the data before transferring it to other tiers for more advanced processing. Unlike traditional network gateways, which habitually perform protocol translations, today's IoT gateways are full-fledged computer systems. They are capable of performing advanced tasks such as protocol and data bridge between devices that use different communication protocols and data formats, protection against malware, and storage and analytics [12]. For example, the Dell Edge Gateway 5000 family can aggregate data, perform analysis locally, and deliver only meaningful information to the next level.

4.2. Fog broker

The Fog Broker extends the capabilities of the IoT gateway in order to deal with a cluster of fog nodes. It also decouples the applications from dealing directly with the fog nodes. Given that these applications do not have the capabilities to monitor the available resources of fog nodes, they delegate to the fog broker the selection of appropriate fog nodes capable of meeting their requirements in terms of latency, response time and throughput. The components of the Fog Broker include the Fog Resources Monitor (FRM), the Task Scheduler (TS), the Identity and Access Manager (IAM), the Fog Services Registration Manager, the Queue of Applications Inquiries (QAI), and the Queue of Periodic Tasks (QPT). Periodic tasks can come with different frequencies and deadlines, according to the current system

configuration and requirements of the environment (for example, the current number of vehicles at the road intersection and the current status of the nearby roads). IAM is responsible for managing applications profiles, including their preferences in terms of personalized services and required QoS. FRM is in charge of collecting up-to-date information about the available resources of each fog node in the fog cluster. TS collects all the tasks to be processed from both the QAI and QPT queues. This component is responsible for the formulation and configuration of the task allocation process at the fog nodes. It mainly implements a placement algorithm to achieve a sub-optimal mapping that minimizes the latency of application tasks. The Task Scheduler output is a scheduled mapping between the tasks to be performed and the fog nodes. This component is also responsible for assigning tasks to the fog nodes assigned to them. Metaheuristic-based approaches can often be used to reach suboptimal solutions faster than placement algorithms that seek to reach optimal mappings [7] [17][21].

4.3. Fog cluster

Several heavy field applications such as computer vision, object recognition, image processing, and anomaly detection, could benefit from fog computing because they do not tolerate high latency and high-cost network bandwidth for data transfer to the cloud. Their respective tasks must be distributed and executed on a cluster of fog nodes to meet their strict latency and bandwidth requirements. As shown in Figure 2, the fog broker's scheduler is responsible for assigning tasks to the fog nodes in the cluster for processing. Also, fog nodes can support executing tasks in virtual machines or containers using well-known orchestration frameworks such as Kubernetes, Apache Mesos-Marathon, and Docker Swarm [3]. Fog nodes can also implement several load balancing policies to distribute the load between the virtual machines of the node.

4.4. Fog node

As we mentioned earlier, fog nodes are the main components of fog computing infrastructures, which provide data storage and computing resources to edge devices. The typical components of a fog node might include an orchestration framework, a distributed messaging system, a data processing engine, data storage, and a load monitoring component. In recent years, several implementations of the publish-subscribe message broker have been developed. However, the most popular are ActiveMQ, RabbitMQ, and Kafka. Apache Kafka is renowned as the most advanced open-source distributed messaging system that can handle data streams efficiently and in a scalable manner [18]. Kafka is perfect for real-time scenarios such as telemetry from sensors, social analytics, clickstream analysis, and network monitoring. Kafka integrates well with many data processing engines such as Apache Storm, and Apache Flink. In our previous work, we proposed a fog node architecture for real-time processing of urban IoT data streams [11].

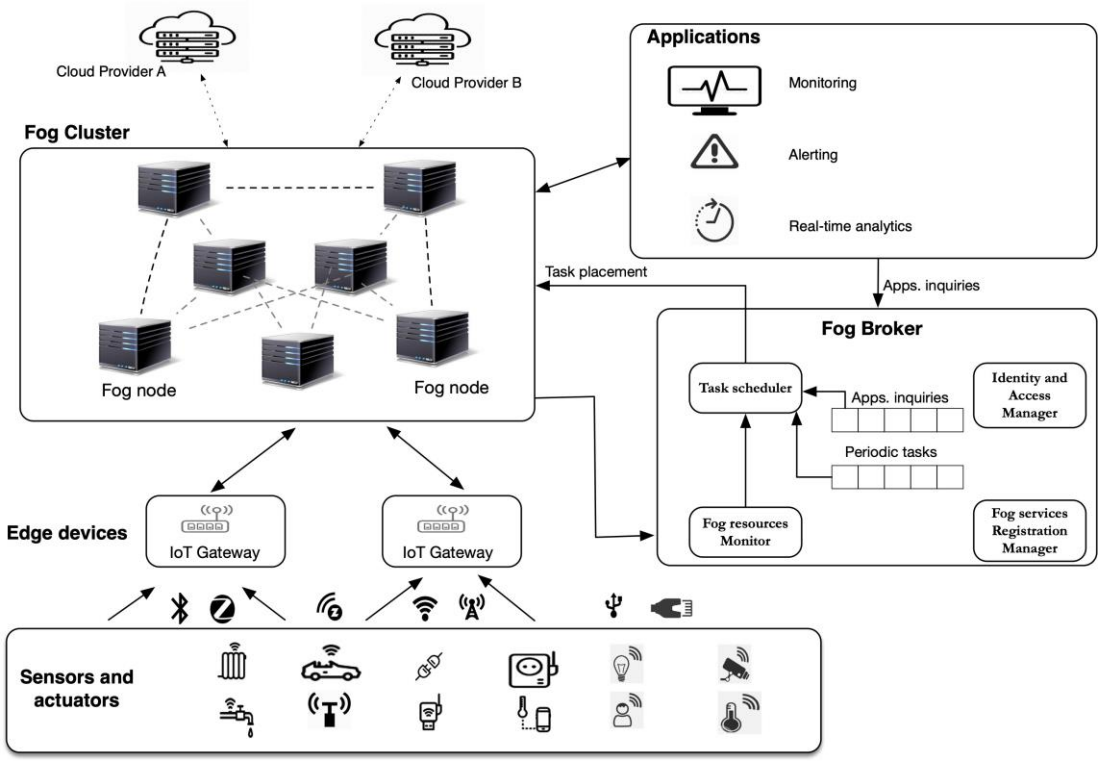

Fig. 2. Architecture for QoS-based task placement on the fog cluster

5. Task scheduling and placement

In this work, scheduling is the process that maps and manages the execution of application tasks on fog computing resources (fog nodes). It aims to allocate appropriate resources to tasks so that tasks' execution can be completed to satisfy the performance goal specified by the applications (see Figure 3).
In the context of fog environments, scheduling decisions need to be made as quickly as possible. The rationale is that, as mentioned earlier, fog computing is intended to improve application latency, and many applications may compete for the appropriate resources, and time slots desired by an application

task can be occupied by another application at any moment. Our objective here is to minimize the latency of tasks and ensure that time-sensitive applications meet their deadlines. In some scenarios, fog brokers associated with different IoT gateways might compete for getting the resources of the same cluster of fog nodes. This is the case when IoT gateways are deployed in the same area and are configured to interact with the same cluster of fog nodes.

The Task scheduler orchestrates all work across the fog nodes cluster. It tracks all active fog nodes (workers) and applications and manages the workers to perform the tasks requested by the applications. It tracks the current state of the entire cluster, determining which tasks execute on which workers in what order. It updates the state of tasks in response to stimuli from workers and applications. After the update of any new information, it ensures that the entire system is on track to complete the desired tasks. Workers provide two functions: i) process tasks as directed by the scheduler, and ii) store and serve computed results to other workers or applications. Each worker has a queue of tasks that it has to execute as requested by the scheduler. It stores the results of these tasks locally and serves them to other workers of the cluster or the scheduler. If a worker is asked to execute a task for which it does not have all of the necessary data, then it may reach out to its peer workers in the cluster to get the necessary dependencies. The scheduler can execute several scheduling policies that can range from static approaches, such as random and round-robin, to dynamic policies that make use of the current status of each worker. Dynamic policies very often aim to make the optimal or near-optimal placement of tasks to meet the QoS requirements of applications [23].

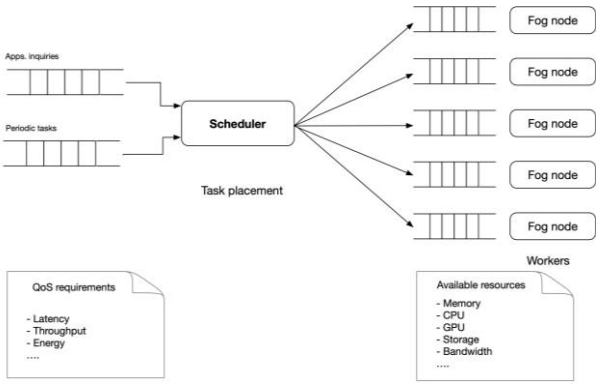

Fig. 3. Tasks scheduling in a scheduler-workers model

6. Experimentation

To assess the performance of application tasks' placement in a fog environment, we use CloudAnalyst [4]. CloudAnalyst is a simulation tool that allows simulating cloud environments by providing the ability to define data centers and the hosts on each data center, and specify three scheduling policies (closest data center, optimize response time, reconfigure dynamically with load). Each host can support several virtual machines. The tests were simulated on MacBook Pro, core i5 processor, 3.40 GHz with 8 GB of memory. As in our context, fog nodes are located in the same environment close to IoT devices; we are considering the same locality as a data center with several fog nodes (hosts). Table 1 shows the deployment scenario of fog nodes, and Table 2 shows the number of virtual machines that each fog node supports.

Table 1. Deployment scenario of fog nodes

Fog Node	Memory (Mb)	Storage (Mb)	Available BW	Nbr. proc.	Proc. speed	VM Policy
FN1, FN3, FN5	8192	1048576	1024	4	10000	time-shared
FN2, FN4	16384	1048576	1024	8	10000	time-shared

Table 2. Deployment scenario of virtual images in the fog nodes

Fog Node	OS	VMM	Nbr. VMs	Image Size (Mb)	Memory (Mb)	BW
FN1, FN3, FN5	linux	Xen	5	10000	512	1000
FN2, FN4	linux	Xen	10	20000	1024	1000

CloudAnalyst provides support for simulating three policies for load balancing across VMs in a single fog node. These policies are Round-Robin, Equally Spread Current Execution Load, and Throttled. Also, it provides three broker scheduling policies: Optimize Response Time (ORT), Closest Fog Node (CFN), and Reconfigure Dynamically with Load (RDL). With the above configuration and setting, we simulate the execution of the application requests for 24 hours using these three scheduling policies, respectively. The number of simultaneous requests a fog node supports is set to 10, and the load balancing policy of the VMs in a fog node is the Equally Spread Current Execution Load policy, Round Robin, and Throttled respectively. Each execution, corresponding to one of the scheduling policies, provides the following results:

1) Average, minimal, and maximal overall cluster response time.
2) Average, minimal, and maximal overall fog node processing time.
3) Average, minimal, and maximal request servicing time of each fog node.

Figure 4 depicts the overall cluster average response time for each scheduling and load balancing policy. Figure 5 depicts the overall fog node average processing time for each scheduling and load balancing policy. Finally, Figure 6 depicts the average

request servicing time of each fog node and for each scheduling and load balancing policy.

The results show that the average response time of the cluster of fog nodes is around 109 ms, which is acceptable for many applications. The minimum response time obtained is 42.42 ms. The best average response time is obtained using the RDL and CFN scheduling policies with the Throttled load balancing policy. These two scheduling policies did not perform well with the Round-Robin load balancing policy. The ORT policy, however, gave better results with Round-Robin.

The average processing time of all fog nodes depends on the load balancing policy used by the fog nodes. The best results were obtained with the Throttled load balancing policy with an average processing time of 0.6 ms for the three scheduling policies.

The average request servicing time is less than 2ms on all nodes of the cluster and for all the three scheduling policies and the three load balancing policies. Also, the throttled load balancing policy allows to have uniform average request servicing time of 0.6 ms and uniform load across the fog nodes of the cluster with the three scheduling policies.

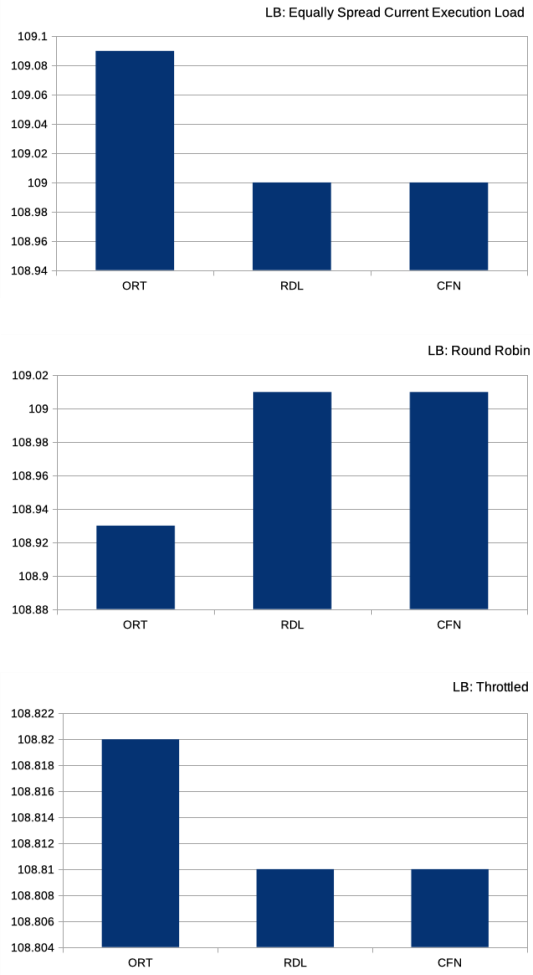

Fig. 4. Cluster Average Response Time

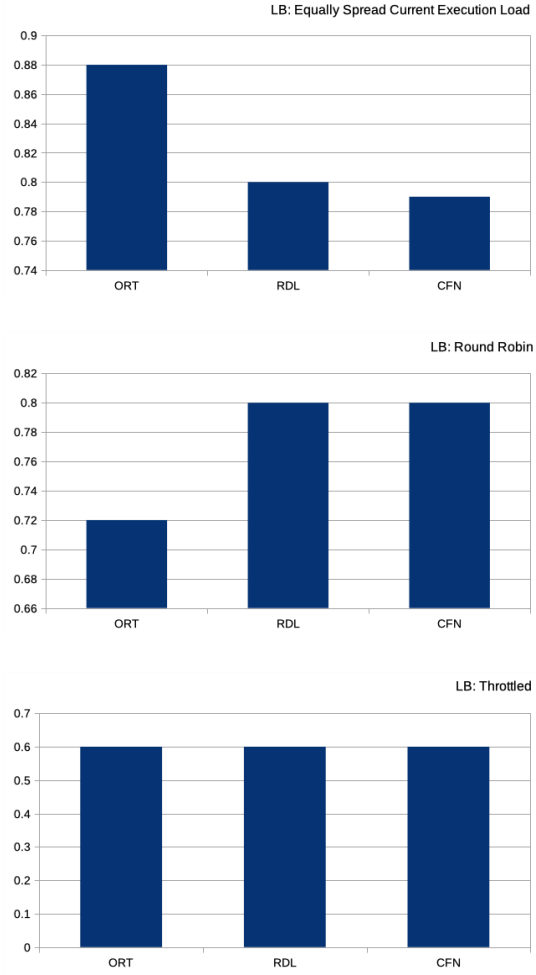

Fig. 5. Fog Node Average Processing Time

7. Conclusion

Organizations that deploy IoT solutions, to improve their services and products, need to process IoT data streams in near real-time and use data analytics to make sense of collected data. Over the last few years, collected data are typically conveyed to cloud servers for processing, storage, and analysis. This scenario is feasible in many use cases. However, time-sensitive IoT applications, such as object detection and processing videos recorded by CCTV cameras, cannot endure sending data streams to the cloud for processing due to their higher network bandwidth requirements and the high latency experience.

In this paper, we have described an architecture for tasks' placement in a fog computing environment. The processing of data streams at the fog layer reduces network traffic and latency for time-sensitive applications. The main component of the architecture is the fog broker, responsible for scheduling the execution of application tasks on suitable fog nodes using various scheduling policies. We carried out several simulations using the CloudAnalyst simulation tool and three scheduling policies and three load balancing policies of VMs in fog nodes. The results show that the best results in terms of latency, average fog node processing time, and balancing the fog nodes load are obtained using the Throttled load balancing policy irrespective of the scheduling policy.

As future work, we intend to investigate: i) the effect of several parameters (number of fog nodes and number of virtual machines) on the applications' latency; ii) Workflow scheduling in a fog cluster as many applications are executed as workflows of several tasks; ii) The security challenges at fog nodes and how

they impact task placement; and iv) Fog AI and deep learning models in fog brokers and fog nodes and how they could assist in task placement decisions.

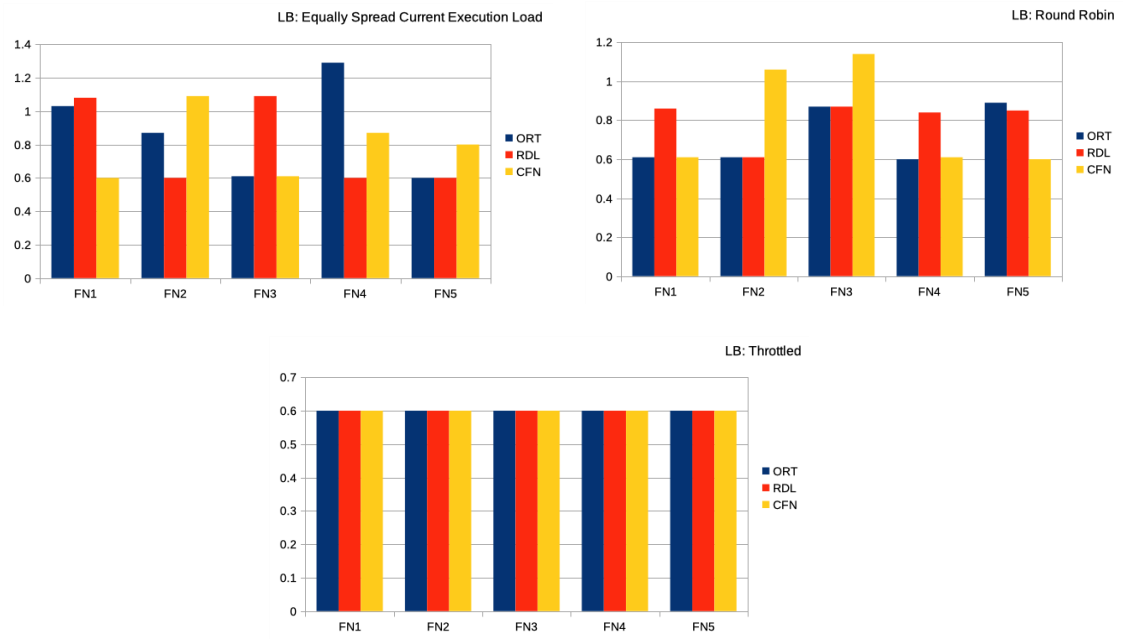

Fig. 6. Average Request Servicing Time of the fog nodes

Acknowledgements

This work is supported by the UAEU Program for Advanced Research Grant N. G00003443.

References

[1]. Anawar MR, Wang S, Zia MA, Jadoon AK, Akram U, and Raza S. Fog Computing - An Overview of Big IoT Data Analytics. Wireless Communications and Mobile Computing, 2018; 1:1–22. https://doi.org/10.1155/2018/7157192

[2]. Anil S and Nitin A. Load balancing aware scheduling algorithms for fog networks. Software: Practice and Experience, 2019; 3(1):1–19.

[3]. Arif A and Guillaume P. Docker Container Deployment in Fog Computing Infrastructures. International Journal of Cloud Computing, 2018; 1–20.

[4]. Bhathiya W, Rodrigo C, and Buyya R. CloudAnalyst: A CloudSim-based tool for modelling and analysis of large scale cloud computing environments. In The 24th IEEE International Conference on Advanced Information Networking and Applications, 2010; 446–452.

[5]. Cheng B, Solmaz G, Cirillo F, Kovacs E, Terasawa K, and Kitazawa A. FogFlow: Easy Programming of IoT Services Over Cloud and Edges for Smart Cities. IEEE Internet of Things Journal, 2018; 5(2):696–707. https://doi.org/10.1109/JIOT.2017.2747214

[6]. Cisco.com. Cisco fog computing solutions: Unleash the power of the internet of things. https://www.cisco.com/c/dam/en us/solutions/trends/iot/docs/computingsolutions.pdf, 2015.

[7]. Consoli S and Darby-Dowman K. Combinatorial optimization and metaheuristics. Technical report, School of information systems, computing and mathematics, Brunel University, 2006.

[8]. Ullman JD. NP-complete scheduling problems. Journal of Comp. and Sys. Sciences, 1975; 10(3):384–393. https://doi.org/10.1016/S0022-0000(75)80008-0

[9]. Dadmehr R and Mohsen N. Low-latency and energy efficient scheduling in fog-based IoT applications. Turkish Journal of Electrical Engineering & Computer Sciences, 2019; 27(2):1406–1427. https://doi.org/10.3906/elk-1810-47

[10]. Dastjerdi AV, Gupta H, Calheiros RN, Ghosh SK, and Buyya R. Fog computing: Principles, architectures, and applications. In Internet of Things, Elsevier, 2016; 61–75. https://doi.org/10.1016/B978-0-12-805395-9.00004-6

[11]. Badidi E. A Fog Node Architecture for Real-Time Processing of Urban IoT Data Streams. In Advances in Intelligent Systems and Computing, Springer International Publishing, 2019; 330–341. https://doi.org/10.1007/978-3-030-19807-7_32

[12]. Fatemeh J, Olivia SJ, Timothy L, and Frank S. Cognitive IoT Gateways. In the SIGCOMM Posters and Demos, New York, USA, ACM Press, 2017; 121–123.

[13]. Flavio B, Rodolfo M, Jiang Z, and Sateesh A. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC '12, New York, NY, USA, ACM, 2012; 13–16.

[14]. Gupta H, Dastjerdi AV, Ghosh SK, and Buyya R. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. Software: Prac. and Exp., 2017; 47(9):1275–1296. https://doi.org/10.1002/spe.2509

[15]. Haruna AM, Francisca OO, and Ezendu A. User mobility and resource scheduling and management in fog computing to support IoT devices. In the Seventh

International Conference on Innovative Computing Technology (INTECH), 2017; 191–196.

[16]. Hong CH, Lee K, Kang M, and Yoo C. QCon: QoS-aware network resource management for fog computing. Sensors (Switzerland), 2018; 18(10):3444. https://doi.org/10.3390/s18103444

[17]. Tarokh MJ, Yazdani M, Sharifi M, and Mokhtarian MN. Hybrid Meta-heuristic Algorithm for Task Assignment Problem. Journal of Optimization in Industrial Engineering, 2011; 4(7):45–55.

[18]. Kreps J, Narkhede N, and Rao J. Kafka: A distributed messaging system for log processing. In the Sixth International Workshop on Networking Meets Databases Workshop, Athens, 2011; 1–7.

[19]. Satyanarayanan M, Bahl P, Caceres RR, and Davies N. The case for vm-based cloudlets in mobile computing. Pervasive Computing, IEEE, 2009; 8(4):14–23. https://doi.org/10.1109/MPRV.2009.82

[20]. Satyanarayanan M, Simoens P, Xiao Y, Pillai P, Chen Z, Ha K, Hu W, and Amos B. Edge Analytics in the Internet of Things. IEEE Pervasive Computing, 2015; 14(2):24–31. https://doi.org/10.1109/MPRV.2015.32

[21]. Mala K and Sarbjeet S. A review of metaheuristic scheduling techniques in cloud computing. Egyptian Informatics Journal, 2015; 16(3):275–295. https://doi.org/10.1016/j.eij.2015.07.001

[22]. Manvi SS and Shyam GK. Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey. Journal of Network and Computer Applications, 2014;41:424–440. https://doi.org/10.1016/j.jnca.2013.10.004

[23]. Kumar M, Sharma SC, Goel A, and Singh SP. A comprehensive survey for scheduling techniques in cloud computing. Journal of Network and Computer Applications, 2019; 143:1–33. https://doi.org/10.1016/j.jnca.2019.06.006

[24]. Chiang M and Zhang T. Fog and IoT: An Overview of Research Opportunities. IEEE Internet of Things Journal, 2016; 3(6):854–864. https://doi.org/10.1109/JIOT.2016.2584538

[25]. Varshney P and Simmhan Y. Demystifying Fog Computing: Characterizing Architectures, Applications and Abstractions. In The 2017 IEEE 1st International Conference on Fog and Edge Computing, 2017. https://doi.org/10.1109/ICFEC.2017.20

[26]. Skarlat O, Schulte S, Borkowski M, and Leitner P. Resource Provisioning for IoT Services in the Fog. 2016 IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA), 2016; 32–39. https://doi.org/10.1109/SOCA.2016.10

[27]. Bayer T, Moedel L, and Reich C. A Fog-Cloud Computing Infrastructure for Condition Monitoring and Distributing Industry 4.0 Services. In CLOSER, SCITEPRESS Science and Technology Publications, 2019;233–240. https://doi.org/10.5220/0007584802330240

[28]. Tordera EM, Masip-Bruin X, Garcia-Alminana J, Jukan A, Ren GJ, Zhu J, and Farre J. What is a Fog Node? A Tutorial on Current Concepts towards a Common Definition. https://arxiv.org/abs/1611.09193, 11 2016.

[29]. Wu L, Garg SK, Versteeg S, and Buyya R. SLA-Based Resource Provisioning for Hosted Software-as-a-Service Applications in Cloud Computing Environments. IEEE Transactions on Services Computing, 2014; 7(3):465–485. https://doi.org/10.1109/TSC.2013.49

[30]. Masip-Bruin X, Marin-Tordera E, Tashakor G, Jukan A, and Ren GJ. Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems. IEEE Wireless Communications, 2016; 23(5):120–128. https://doi.org/10.1109/MWC.2016.7721750

[31]. Yannuzzi M, Lingen FV, Jain A, Parellada OL, Flores MM, Carrera D, Perez JL, Montero D, Chacin P, Corsaro A, and Olive A. A new era for cities with fog computing. IEEE Internet Computing, 2017; 21(2):54–67. https://doi.org/10.1109/MIC.2017.25

[32]. Yao J and Ansari N. QoS-Aware Fog Resource Provisioning and Mobile Device Power Control in IoT Networks. IEEE Transactions on Network and Service Management, 2019; 16(1):167–175. https://doi.org/10.1109/TNSM.2018.2888481

[33]. Vannithamby R and Talwar S. Towards 5G: Applications, requirements and candidate technologies. Wiley, 1st Edition, 2017; ISBN-10: 9781118979839. https://doi.org/10.1002/9781118979846

[34]. Santos J, Wauters T, Volckaert B, and De Turck F. Fog computing: Enabling the management and orchestration of smart city applications in 5G networks. Entropy, 2018; 20(1): p. 4. https://doi.org/10.3390/e20010004

[35]. Van Lingen F, Yannuzzi M, Jain A, Irons-Mclean R, Lluch O, Carrera D, Perez JL, Gutierrez A, Montero D, Marti J, Maso R, and Rodriguez AJP. The Unavoidable Convergence of NFV, 5G, and Fog: A Model-Driven Approach to Bridge Cloud and Edge. IEEE Communications Magazine, 2017; 55(8): 28–35. https://doi.org/10.1109/MCOM.2017.1600907