Simulating Smart Campus Applications in Edge and Fog Computing

Denis Contini*, Lucas Fernando Souza de Castro†,
Edmundo Madeira‡, Sandro Rigo§ and Luiz Fernando Bittencourt¶
Institute of Computing, University of Campinas - UNICAMP
Campinas - Brazil
Email: *denis.contini@students.ic.unicamp.br,
{†lucas.castro, ‡edmundo, §sandro, ¶bit}@ic.unicamp.br

***Abstract*—Due to the rapid increase of IoT applications and their use in many different areas, large amounts of data have been generated to be processed and stored. In this scenario, some applications are sensitive to high latency and response times. In order to fulfil these requirements, Edge and Fog Computing appear with the objective of bringing processing and storage devices closer to applications and management mechanisms. In this context, due to limitations related to high cost, scalability and planning, several mechanisms and algorithms need to be simulated before being implemented in the real world. This paper presents a comparison between two simulation tools and their main characteristics (EdgeCloudSim and iFogSim) using a smart campus scenario deployed at the University of Campinas, where the sensors collect data from water meters and smart energy marker watches, in addition to smart public transportation and battery disposal bins. Our evaluation shows that the information processing in edge and fog can efficiently serve the applications, however, each simulation tool has its specificities, and should be used according to the researcher's objectives and needs.**

***Index Terms*—fog computing; edge computing; internet of things; simulations.**

I. Introduction

Recently there has been a huge increase in the number of IoT (Internet of Things) applications in different areas and sectors. As a result of this increase, the amount of data sent and received has also increased considerably.

Although the processing and storage of data in the cloud are abundant, applications that are sensitive to latency and low response time are hindered due to the long way to reach cloud servers. To deal with these critical factors, the concepts of Fog [1] and Edge [2] Computing arise.

Although they present themselves as different concepts and specific characteristics, fog and edge computing have a common target: bring the processing and storage devices closer to the devices allocated in the network access layers. To help with this task, some works recommend the use of orchestrating devices, so that the tasks are better distributed among the processing devices [13] [14].

Even with the many benefits and advantages that can be achieved by allocating processing and storage devices close to the edge, capacity planning is also necessary like cloud applications. Besides, each application has its specificities and characteristics, that is, the proposed solution for one scenario may not work for others.

In order to minimize problems and study possible configurations, the scenarios, as well as applications, services, and resources allocated in the cloud, must be simulated and tested considering many technical factors.

Given the challenges, restrictions, and difficulties encountered when simulating applications in edge and fog, this paper aims to present the main characteristics, metrics and differences of two simulation tools: EdgeCloudSim [3] and iFogSim [4]. For this, a scenario based on an intelligent campus application from the University of Campinas (UNICAMP) was used, where the data collected by sensors are directed to processing devices allocated in edge and fog. The simulated applications included in the scenario were as follows: i) smart energy; ii) smart water; iii) smart bus; iv) smart battery; and v) AR - Augmented Reality.

In the implemented scenario, applications (i), (ii), (iv) and (v) transmit data over a wireless network (IEEE 802.11n). Application (iii) sends data through a second-generation mobile network (2G - GPRS). The bandwidth rates used in each test, according to the established transmission technology, are described in detail in sections III-A and III-B. In all the elaborated cases, the data transmitted by the sensors are treated as text strings of fixed size defined by the simulation tools.

After running the simulation rounds, it was possible to notice that the devices with less computational power when compared to those available in cloud data centers, allocated in edge and fog, fulfilled the requirements of the tasks, indicating, also resulting in reduced energy consumption by processing devices when compared to large servers available in the cloud.

The rest of this paper is organized as follows. Section 2 introduces the basic concepts of Edge and Fog Computing and the main characteristics of the simulation tools used (iFogSim and EdgeCloudSim). Section 3 presents the case study utilized as a basis for simulations in the EdgeCloudSim and iFogSim, as well as the results achieved. Finally, Section 4 shows the conclusion.

II. Basic concepts

This section shows the concepts and definitions related to Edge and Fog Computing. Moreover, we present the iFogSim and EdgeCloudSim as the simulation tools used in this work and their main characteristics and limitations.

A. Edge Computing

As mentioned in [2], the logic behind edge computing is that the processing must occur in the vicinity of the data sources, that is, in the access layers where the devices and clients are connected. In addition, edge computing can be interchangeable with fog computing, differing in that the concentration of processing devices occurs closer to the devices, while fog computing is concentrated on the infrastructure side, in a multilevel, hierarchical layer between the edge and the cloud.

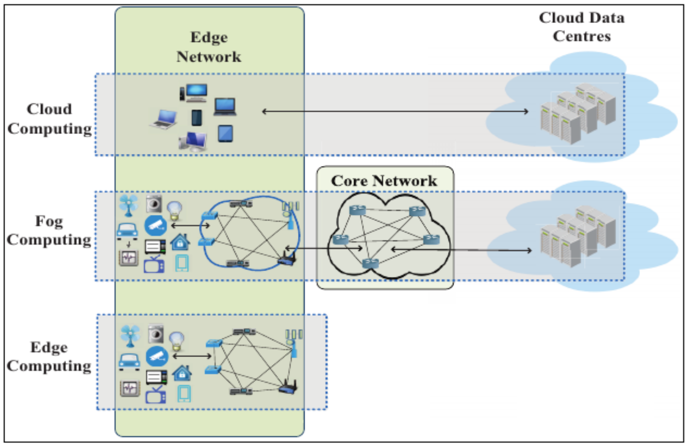

Fig. 1. Computational domain of cloud computing, fog and edge - adapted from [5].

At the edge, data can be delivered, processed, and stored, or even redirected to resources with higher capacity allocated in the cloud. For this reason, the edge needs to be well designed to meet the requirements with efficiency, reliability, security, and privacy. Fig. 1 illustrates a scenario where the data generated by the sensors in the network access layer are sent to the edge to be processed or stored. In this way, the use of the available bandwidth on the internet communication link can be reduced and responses could be generated more quickly so that processing or storage occurs in the cloud only when necessary [6].

B. Fog Computing

Although the computational power of many edge devices has increased, the available resources can not be sufficient to process the high number of tasks and data coming from the application's sensors. Given this scenario, [1] first proposed, in 2012, the concept of Fog Computing as "a highly virtualized platform that provides processing, storage, and network services between devices and traditional cloud datacenters, typically, but not exclusively, located at the edge of the network". In [11], the authors define the Fog Computing as a scenario where several heterogeneous and decentralized ubiquitous devices can communicate and exchange information between themselves and the network to process the tasks without requesting devices or applications allocated in the cloud.

Besides the definitions presented by [1] and [11], it is important to highlight that Fog Computing can be composed of several hierarchical layers between the edge and the cloud.

Summarizing the Edge and Fog computing concepts, edge presents its devices usually one hop away from the devices at the network layer. Whereas the fog computing [1] presents itself as a layer between the edge and the cloud, moreover, it may offer one or more layers, at different levels, two hops, or more apart.

C. Simulators

As with Cloud Computing and all the services made available through it, researchers and developers also seek to test scenarios and management mechanisms in Fog and Edge computing in order to foresee adverse scenarios. Also, issues and challenges related to deployment costs and scalability make simulations even more interesting due to their lower cost and flexibility, in addition to the time control factor, where long periods may be shortened during simulations. In general, all these factors in conjunction collaborate so that both risks and design errors are reduced in different projects and applications [10].

Although the amount of research on Cloud, Fog, and Edge computing paradigms is relatively abundant in the literature, the same can not be said about simulations in edge and fog environments. In [10] and [8], it is possible to verify the main simulation tools and their characteristics in the edge and fog paradigms, many of which derive from simulation tools widely used in cloud computing. Among the most common in the literature, it is possible to quote: FogNetSim++, iFogSim, EdgeCloudSim, IoTSim, FogTorchII, EmuFog, Fogbed, and YAFS. In the next subsections, the two tools used in this work (iFogSim and EdgeCloudSim) are described in detail.

D. iFogSim

iFogSim [4] was created with the initial objective of allowing the creation of simulation environments in Fog, with the focus on presenting metrics related to latency, energy consumption, and network usage. It uses as a ground the CloudSim [12] simulator, developed in Java and widely used in cloud simulations.

Natively, iFogSim works with two main policy of allocation tasks : (a) allocation and processing of services in Cloud; (b) allocation and processing of services in Fog, a layer between the edge and the cloud. In addition to native methods, dynamic policies can be implemented, where tasks are distributed and processed collaboratively in cloud and fog. The tool also has extensions to support data allocation strategies according to specific objectives such as, for example, low latency, network congestion and energy consumption as cited in [9] and [10].

In its service allocation policy, iFogSim does not provide communication between devices under the same hierarchical level, providing the allocation of services only at different levels to be simulated, although updates are in progress to make this feature available [4].

In iFogSim, data is represented by sequences of values, called tuples, and their flows occur in both directions: from the application devices (sensors) to the fog or cloud processing devices and vice versa, once processed.

A monitoring layer is available, which controls the use of resources, such as the energy consumption of the various devices in the topology, the availability of sensors, actuators, and the fog devices to then generate the outputs corresponding and also so that the resources can be better used according to the availability of each one to reduce their idle time.

As weaknesses, iFogSim does not provide, in a native way, models of mobility of devices, in addition to having a limited scalability of resources and devices, as discussed in [4].

E. EdgeCloudSim

Like the iFogSim, the EdgeCloudSim [3] is based on CloudSim [12]. It was specifically designed to assess the needs of computational and communication links at the edge of the network. Unlike iFogSim, it provides mobility scenarios as a native resource. In addition to the features presented, it allows the creation and configuration of devices in the simulation scenario through XML files instead of defining them in each of the classes or methods in the source codes.

The Edge Computing topology proposed by EdgeCloudSim is illustrated by Fig. 2A:

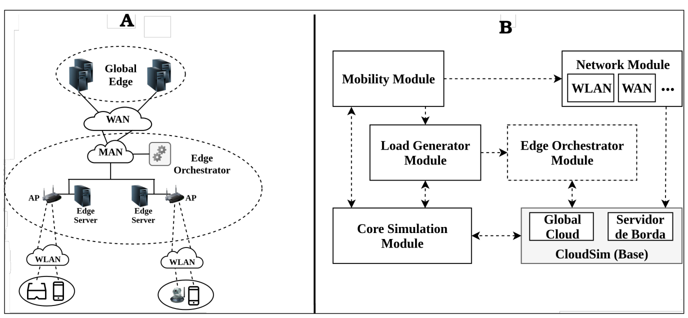

Fig. 2. EdgeCloudSim - Topology and modules - Adapted from [3].

- **Client Devices**: devices that take advantage of the architecture (sensors and smartphones);
- **Edge Server**: nodes that perform processing at the edge of the network. These nodes are linked to an Access Point and therefore to the MAN (Metropolitan Area Network) and the edge orchestrator;
- **Edge Orchestrator**: performs the delegation of tasks and data from client devices between edge and cloud nodes;
- **Global Cloud**: can receive tasks from users or edge nodes by offloading from the edge orchestrator;
- **Communication networks:**
  - **WLAN**: communication network between client and edge devices;
  - **WAN**: network between the edge orchestrator and the cloud;
  - **MAN**: metropolitan communication network between the edge orchestrator and the edge nodes.

In addition to the topology, EdgeCloudSim is composed of other complementary modules (mobility, task generator, orchestrator, and simulation core), as shown in Fig. 2B.

III. TESTBED

In order to evaluate and compare the functioning of two simulation tools for edge and fog computing, the data used as a reference for this work was based on a smart campus application designed for the University of Campinas (UNICAMP).

The testbed is composed of client devices (sensors) such as, 300 water consumption meters (smart water meters), 300 electricity consumption meters (smart energy meters), 50 smart battery collection points, 6 location sensors (GPS) installed on public transport were initially taken into account (whose objective is to check the location of the buses in real-time) and 100 Augmented Reality (AR) devices (to offer greater interaction between users and services offered by the campus), reaching a total of 756 devices, as shown in Table I.

Due to limitations in replicating the same data format and tasks used in the smart campus application in both simulators, they were treated as text strings exchanged between clients (sensors) and servers, generating the necessary traffic to the application. In the EdgeCloudSim is possible to set the network and hardware requirements (e.g., cpu time, mips, network bandwidth, and so on) through a xml file.

For testing and simulation purposes, 800 client devices were considered. The total number of sensors also varied in the simulations, between 1,600 and 8,000, in EdgeCloudSim.

TABLE I
NUMBER OF DEVICES USED IN THE SCENARIOS

Devices	Number of Devices
Smart Energy	300
Smart Water	300
Smart Bus	6
Smart Battery	50
AR	100
Total	**756(∼800)**

A. Simulations - EdgeCloudSim

The simulations in EdgeCloudSim were conducted in different scenarios and configurations, such as: i) Single Tier - task allocation only at the edge; ii) Two Tier - task allocation at the edge and cloud; iii) Two Tier with Orchestrator - task allocation at the edge and cloud with orchestrator.

The metrics applied to evaluate the simulations were the number of failed tasks and the hit rate. The number of failed tasks corresponds to the number of tasks that were not completed within the deadline stipulated by the application. Besides, a task may not be accomplished due to some factors, namely: i) lack of computing capacity of the device at the edge; ii) high latency; iii) mobility of the client device. Finally, the hit rate comprises the normalized value of the percentage ratio of tasks successfully served by edge devices.

In order to clarify the evaluated scenarios, Table II presents the configurations of the four groups elaborated and tested (A, B, C, and D). For each scenario group, different orchestration policies were employed: Next **(N)**, Random **(R)**, Worst **(W)** and Best **(B)** Fit. Similar to the allocation of RAM, Next, Random, Worst, and Best Fit are policies for orchestrating tasks between edge devices taking into account their respective computational capabilities. The Next Fit policy assumes the distance between devices. Best and Worst take into account the computational capacity, Best being the device with the best resources to meet the task and Worst, the worst. Finally, Random randomly assigns the device to be allocated.

In addition, different combinations of edge devices were used: 8 and 16 Raspberry Pi 3 and 4, as well as the combination of both: 32 RPx (16 Raspberry Pi 3 and 16 Raspberry Pi 4).

TABLE II
TABLE II - SCENARIO GROUPS IN EDGECLOUDSIM

Group	# Client Devices	#Edge Server Devices	WAN/GSM/ WLAN	Orch. policy
A	800-1600	8/16 (RP3-4) and 32 RPx	50/—/300	N
B	6400	8/16 (RP3-4) and 32 RPx	[20-50] / [2-5] / [54,150,300]	N,R,W,B
C	8000	8 RPx	50/—/300	N,R,W,B
D	1600-8000	8 RPx	50/—/300	N

Group A was not described in detail, as in all executions the number of Failed Tasks was null and, consequently, the hit rate was 1 (100%).

Group B aims to assess the edge architecture limits since with 800 to 1,600 client devices it was possible to obtain a maximum hit value. In this scenario, the number of devices reached 6,400, combined with different configurations of Edge devices (8 and 16 RP3 and RP4) and also a heterogeneous configuration (16 RP3 + 16 RP4 $\rightarrow$ 32 RPx). Regarding communication, it was possible to note whether the network infrastructure could achieve high traffic values. For this, the configurations were alternated as shown in Table II. Finally, the different orchestration policies (Next, Worst, Random and Best Fit) were observed.

Fig. 3 shows the results obtained from the Group B simulation. The graph on the left shows the comparison of the different orchestration policies (Random, Best, Next and Worst Fit). For this, 8 Raspberry Pi 3 were used. In this case, it is possible to notice that the use of an orchestrator brings greater efficiency in the execution of tasks, because in Single Tier 146,014 tasks failed, and with the use of an orchestrator, this amount decreased to 80,467 in Worst Fit. In addition, different orchestration policies can bring even greater efficiency. In terms of the hit rate, it is possible to summarize that the use of the orchestrator showed a gain of approximately 8%.

The right-hand side of Fig. 3 shows the comparison between two edge devices in different configurations: 16 Raspberry Pi 3 and 8 Raspberry Pi 4. In this scenario, the devices were subjected to the same number of tasks to be distributed. Thus, at the end of the simulation, it was possible to verify the superiority of the Raspberry Pi 4 even without the use of an orchestrator. Finally, with the use of an orchestrator (Next Fit), both configurations obtained a maximum hit rate (100%).

For group C, the focus was on comparing edge devices (8 Raspberry Pi 3 versus 8 Raspberry Pi 4), where 8,000 client devices were inserted. In view of the high efficiency presented by the 6,400 Group B devices, 8,000 devices were tested for excess traffic in the topology. Fig. 4 suggests that orchestration policies have an impact on performance. With this, it is possible to verify that the Next and Worst Fit policies have a better performance than Random and Best Fit. At the end of the simulation, even with an equivalent number of devices, the Raspberry Pi 4 proved to be superior to the Raspberry Pi 3, even in a Single Tier scenario.

In the simulated scenario with the Group D settings, a new task category was added to the simulation: Augmented Reality (AR). Thus, the simulation Group D consists of 8,000 client devices and 8 Raspberry Pi 3 and 8 Raspberry Pi 4 (16 in total). The results obtained can be seen in Fig. 5, where, again, the superiority of the Raspberry Pi 4 even in environments with complex tasks, presenting superior performance with the use of an orchestrator, where only 1 task failed.

Finally, based on the groups established and the results obtained after the simulations, it is possible to conclude that the Edge Computing paradigm is effective in many cases. In scenarios averaging 3 to 4 thousand client devices, 8 Raspberry Pi 3 as edge devices are sufficient. However, as this number increases, the number of failing tasks increases considerably. Unlike the Raspberry Pi 3, the Raspberry Pi 4 proved to be superior in denser and more demanding scenarios (with more than 6,400 client devices).

B. Simulations - iFogSim

Although derived from CloudSim [12], iFogSim [4] also has its specificities and, therefore, difficulties were encountered when replicating the scenario and topology tested and evaluated in EdgeCloudSim, which is more suitable to simulate applications to be processed in edge and cloud, not offering fog processing capabilities.

Even with these difficulties, a topology was created based on the settings of the best simulated scenario in EdgeCloudSim, as specified in Table III. Parameters related to energy consumption could be configured according to the use of data processing and transmission devices, that is, different energy consumption values are configured when the device is in use (Power Busy) or idle (Power Idle).

The Level column specifies in which layer the device is allocated, where lower values correspond to processing devices located closer to the cloud (identified as level 0) and the higher values correspond to processing devices located in layers closer to the edge of the network (the higher the level, the closer to the edge). For communication links, a dedicated bandwidth was used, assuming the same values for upload and download.

Since the task distribution policies, the creation of topologies and metrics of the smart campus environment used in

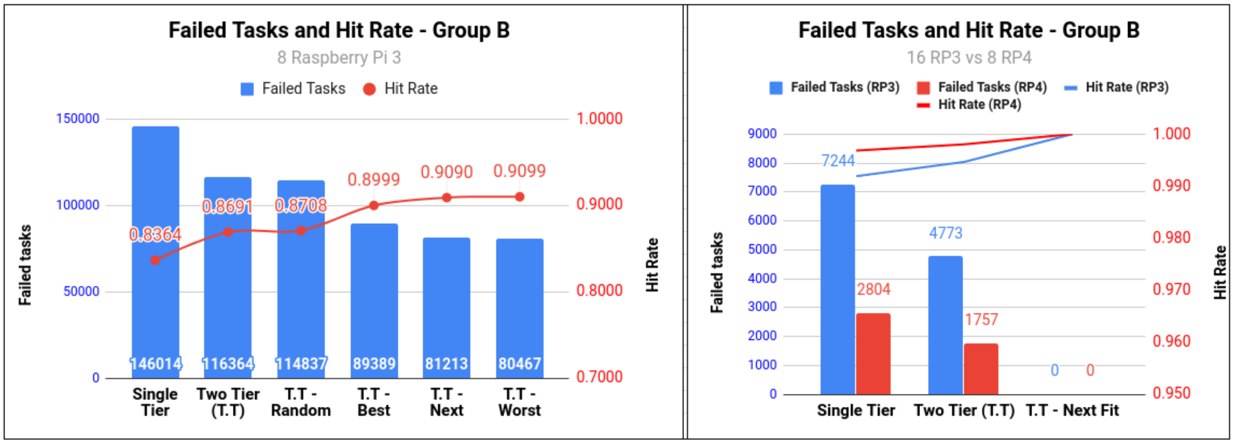

Fig. 3. EdgeCloudSim - Simulation Results - Group B - Comparison of Orchestration Policies and Devices.

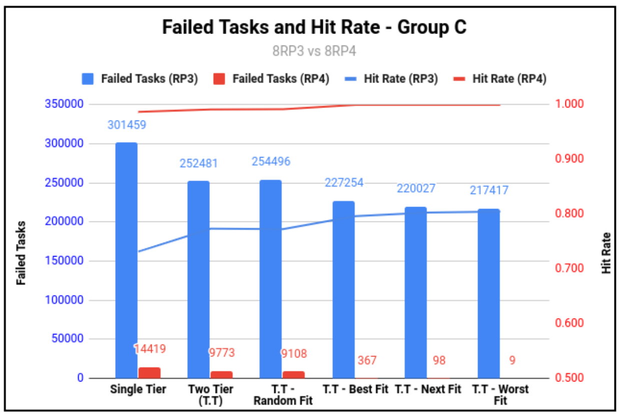

Fig. 4. EdgeCloudSim - Simulation Results - Group B - Comparison of Orchestration Policies and Devices.

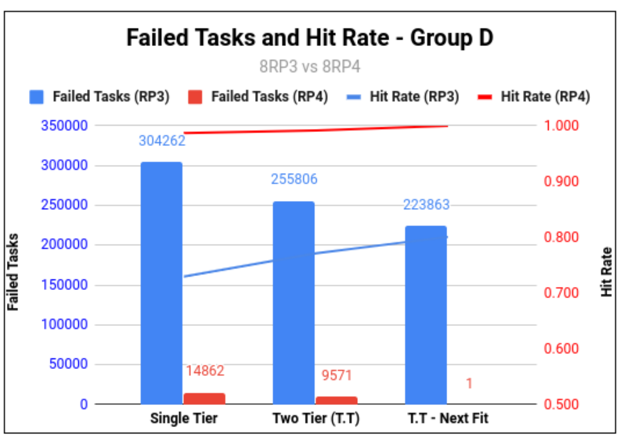

Fig. 5. EdgeCloudSim - Simulation Results - Group D.

EdgeCloudSim could not be replicated with the same fidelity in iFogSim. Fig. 6 presents the implemented topology, taking as reference the game "EEG Tractor Beam" (a latency-sensitive online game) proposed as one of the use cases in [4].

TABLE III
IFOGSIM SIMULATION SETTINGS

Devices	MIPS	RAM (GB)	Up/Down Link (Mbps)	Level	P. Busy (W)	P. Idle (W)
Cloud	75,000	64	100	0	1,200	960
ISP-GW	7,000	4	50	1	100	80
AP (d1,d2)	7,000	4	100	2	100	80
Mobile	7,000	4	300	3	80	50

In the configured topology, 3 sensor devices are associated via WiFi (IEEE 802.11n) to their respective mobile devices, which, in turn, are associated with an Access Point (AP) device, also through the IEEE 802.11n protocol, connected with provider network (Internet).

In this scenario, two sets of devices collect the data and the responses are displayed through the screen of the mobile device (here characterized as an actuator device). Once the topologies and standard allocation policies available in the tool are defined, the processing of data collected by the sensors begins, calculating the costs of processing both in the cloud and in the fog.

For this experiment, 5 simulations were performed so that the arithmetic mean of the values was computed. The operating time used for simulating the implemented scenario was 30 minutes per round.

Based on the results obtained in the simulation, it was possible to create the energy consumption graph of the devices (Fig. 7), where the energy consumption values are compared with cloud and fog processing, demonstrating that, for the scenario and the configurations implemented, the consumption to process the data in the cloud would be considerably higher in the cloud than in fog which would imply higher costs.

It is also important to emphasize that the values presented may vary, as the number of devices and/or the number of sensors increases.

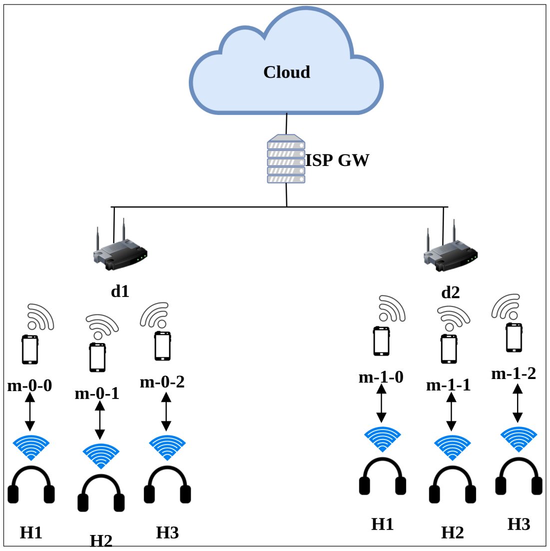

Fig. 6. Simulated topology in iFogSim.

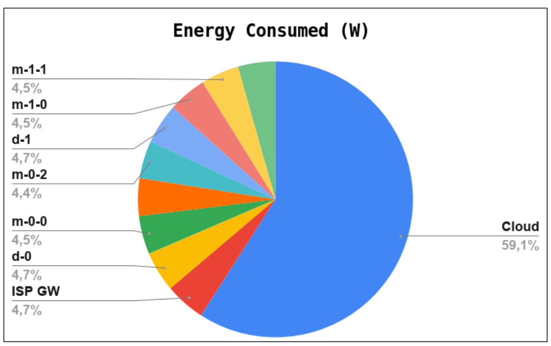

Fig. 7. Graph of the power consumption of devices in iFogSim.

IV. Conclusions

Edge and Fog Computing present themselves as distinct paradigms and with specific characteristics, where one can be used as an extension of the other, or in conjunction with resources and applications allocated in the Cloud so that the processing and storage of data and services can be improved. Also, both paradigms present as possible academic and industrial solutions in the various scenarios in which IoT applications are developed to offer agile information processing and provide low latency.

iFogSim, for example, presents characteristics that allow an approach focused on the energy consumption of the application, as well as metrics of bandwidth usage in applications when performing tasks in fog or cloud (independently or collaboratively), assisting the researcher to design the most efficient scenario (prioritizing energy consumption or optimizing bandwidth consumption).

The EdgeCloudSim shows greater efficiency when the objective is to verify the number of tasks served by the processing devices allocated at the edge, in addition to allowing a more detailed analysis of the various methods of task orchestration, helping the researcher to find the more efficient method according to the scenario to be simulated or, allowing the development of a new orchestration policy.

In the scope of the simulations, a greater number of devices will be used to further scale the proposed scenario, increasing its diversification, not only restricting the Raspberry Pi for Edge and Fog processing. In addition to the increase in the number and diversification of devices, it is also interesting to configure the scenario so that tasks can be sent to the cloud (in the case of EdgeCloudSim). In this way, gains in bandwidth usage and decreases in latency can be better measured.

Acknowledgment

This work is part of the INCT of the Future Internet for Smart Cities (CNPq 465446/2014-0, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and FAPESP 2014/50937-1, 2015/24485-9 and 2015/24494-8). The authors are grateful for the financial support from the institutions mentioned above.

References

[1] F. Bonomi, R. Milito, P. Natarajan, e J. Zhu, “Fog computing: A platform for internet of things and analytics”, Stud. Comput. Intell., vol. 546, p. 169–186, 2014.

[2] J. Cao, Q. Zhang, e W. Shi, Edge Computing: A Primer. 2018.

[3] C. Sonmez, A. Ozgovde, e C. Ersoy, “EdgeCloudSim: An environment for performance evaluation of Edge Computing systems”, in 2017 2nd International Conference on Fog and Mobile Edge Computing, FMEC 2017, 2017, p. 39–44.

[4] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, e R. Buyya, “iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments”, 2017.

[5] R. Mahmud, R. Kotagiri, e R. Buyya, “Fog Computing: A Taxonomy, Survey and Future Directions”, 2018, p. 103–130.

[6] W. Shi e S. Dustdar, “The Promise of Edge Computing”, Computer (Long. Beach. Calif)., vol. 49, no 5, p. 78–81, maio 2016.

[7] L. Bittencourt et al., “The Internet of Things, Fog and Cloud continuum: Integration and challenges”, Internet of Things, vol. 3–4, p. 134–155, out. 2018.

[8] I. Lera, C. Guerrero, e C. Juiz, “YAFS: A Simulator for IoT Scenarios in Fog Computing”, IEEE Access, vol. 7, p. 91745–91758, 2019.

[9] M. I. Naas, J. Boukhobza, P. Raipin Parvedy, e L. Lemarchand, “An Extension to iFogSim to Enable the Design of Data Placement Strategies”, in 2018 IEEE 2nd International Conference on Fog and Edge Computing, ICFEC 2018 - In conjunction with 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, IEEE/ACM CCGrid 2018, 2018, p. 1–8.

[10] S. Svorobej et al., “Simulating Fog and Edge Computing Scenarios: An Overview and Research Challenges”, Futur. Internet, vol. 11, no 3, p. 55, fev. 2019.

[11] L. M. Vaquero e L. Rodero-Merino, “Finding your way in the fog: Towards a comprehensive definition of fog computing”, in Computer Communication Review, 2014, vol. 44, no 5, p. 27–32.

[12] R. Buyya, R. Ranjan, e R. N. Calheiros, “Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities”, in Proceedings of the 2009 International Conference on High Performance Computing and Simulation, HPCS 2009, 2009, p. 1–11.

[13] K. Velasquez, D.P. Abreu, M. Assis, et al., “Fog orchestration for the Internet of Everything: state-of-the-art and research challenges”. J Internet Serv Appl 9, 14 (2018).

[14] K. Velasquez et al., ”Service Orchestration in Fog Environments,” 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud), Prague, 2017, pp. 329-336.