International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Effective Heuristics for Enhancing QoS in Fog Environment through Migration and Replication Techniques

[1]Bekkouche Khawla, [2]Brahimi Said

Submitted: 17/03/2024 **Revised**: 27/04/2024 **Accepted**: 05/05/2024

Abstract: Fog computing has arisen to address the needs of IOT (Internet of Things) applications that are currently unmet by existing solution. However, inef- efficient service management in this dense and geographically dispersed environment degrade service accessibility to users and lead to poor quality of service(QoS)in terms of response time ,congestion network, and the consumption of energy.
This paper introduce three fog service placement methods, which allows users to retrieve service in a faster and more efficient way, to improve service accessibility by replicating service on fog nodes considering Fog Computing characteristics. In these three methods, we take into account the prediction based on service history from fog to each service, we also show the results of simulation experiments using IFOGSIM regarding the performance evaluation of our proposed heuristics.

Keywords: *Fog Nodes, Migration, Accessibility, Duplication, Replica, Prediction.*

1. Introduction

Connected objects are capable of perceiving the environment, exchanging service with each other and with the Cloud, and can therefore transmit information and possibly receive commands and even behave intelligently by providing adequate environmental services. Generally, IoT service is processed and stored in the cloud [1, 2].

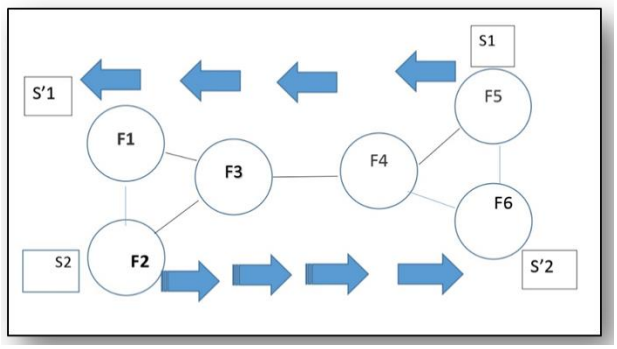

Fig. 1. *Effective service replication and migration*

The latter satisfies the majority of IoT application needs. In terms of ubiquitous access, availability and scalability of processing performance and storage capacity. However, as the Cloud is a centralized data center paradigm, the service generated and the queries sent by the connected objects are transmitted to these centers in the core network. Thus, with the growing number of connected objects and the amount of service produced, sending all service to the cloud will generate bottlenecks [1].

This increases service transmission latency and, subsequently, degrades the quality of service (QoS) [3].

The term fog computing was coined by Cisco [4]. The major aim of fog computing is to increase efficiency, performance and reduce the amount of service for processing, analysis and storage transferred to the cloud.

Service processing and storage equipment are called 'Fog Nodes' in Fog computing. With regard to their performance, these devices are heterogeneous in nature [5]. These items can be modest or resource-rich devices as set-top-boxes, switches and routers, access points.

In fog computing, it is a very important issue to prevent degradation of service accessibility. To overcome this problem a possible solution is by migrating replicas services at fog nodes, which are not the owners of the original service.

In Figure 1, if the replicas of services 1 and 2 are created and migrated, every fog node can access both services. Service replication is very useful for enhancing service accessibility in this way.

In this paper, we present three methods for enhancing quality of service (QOS) with the aim of reducing the response times, minimizing the overall usage consumption of the system without congestion of the network, and enhancing service accessibility. Each method has its advantages and what distinguishes it from the other approaches.

2. Paper organization:

The rest of the paper is organized as follows: After a brief summary of most relevant related work, Section III and IV describe, respectively, our proposed solution as well as simulation results. Finally, conclusions are given in Section V.

[1]- *(LICUS) Laboratory, the University of Skikda, Algeria, Computer Sciences Department. E-mail: kh.bekkouche@univ-skikda.dz,*
[1]- *(LIRE) Laboratory , university of Constantine 2, Algeria, Computer Science department, university of Guelma, 8 May 1945, Algeria. E-mail: brahimi.said@univ-guelma.dz.*

3. Related works:

Several research in the literature have looked into the problem of where IoT service instances should be placed in the fog.

Gupta et al developed IFogSim in the Fog and edge-computing paradigms, the toolkit is used to simulate and mimic IoT resource management approaches. The most difficulty is developing resource management strategies that determine ana- lytic application distribution among edge devices, hence increasing throughput and lowering latency [1].

Naas et al in [6] have extended IFogSim, a simulator of fog and IoT environ- ments. The goal of this extension is to create a platform for creating and evaluating data placement techniques in a fog and IoT environment. The authors contributed three original components. To lower overall system latency, the first to define and compute data location using a linear programming method. The second step is to partition the infrastructure using graph theory to reduce the time it takes to place data.

[7] Proposes hybrid-scheduling systems that combine diverse scheduling crite- ria to cover various types of applications such as workflows and batch processes. Because there is no single solution/criteria that fits all types of queries in a fog environment, this feature is especially critical for fog resource scheduling.

Huang et al.in [8] used a multi replicas model termed 'iFogStorM' technique to solve the installation of numerous data replicas in a Fog Computing infrastructure. They consider the following three data replication issues: The number of replicas and their location. Furthermore, considering the difficulty of deploying many repli- cas in a large-scale infrastructure, The authors proposed a heuristic method they dubbed "MultiCopyStorage." To decrease the searching space for solving the target model, the suggested method employs a greedy algorithm.

Vales et al. [9] suggested a hierarchical hybrid architecture fog storage system. The suggested file system centralized Meta data and policy administration while al- lowing for dispersed data storage and distribution. Furthermore, the authors devise a replication technique that replicates data to edge devices based on node local- isation, which is defined by the distance between users and the data, as well as spatio-temporal data popularity.

The authors in [10] suggested a distributed data placement technique based on dynamic replica formation, replacement, and deletion driven by continuous monitor- ing of data requests from edge nodes.The concept Edge, according to the authors, encompasses the following paradigms: Fog Computing , and Cloudlets. The storage nodes hosting the replicas examine the observed request on the replicas and act as local optimizer in this method. The authors used the FLP to model this situation as a combinatorial optimization problem .

In [11], Ahmed et al described an algorithm for routing requests to the nearest node housing the data requested by the user, which might be a connected object, while balancing the workload. They presented a migration and replication approach named "RMS - HaFC" to lower the average response time and optimize the sys- tem's total energy consumption In comparison to the Centralized and Decentralized approaches, this strategy has yielded promising outcomes.

Despite the existence of a suggested contribution to the placement performance optimization, it found that the issue we are trying to fix is relatively recent.

This work was done with the goals of reducing response time , lowering energy usage, congestion network and guarantee the access to the service , So that whenever the user requests the service, he finds it ..

4. Proposed Solution

1.1 Assumptions

In this section , we describe the general model, as well as we give some details on our proposed methods, then we present three replica placement methods for enhancing service accessibility in the fog computing. The system environment is assumed to be a fog layer where fog nodes access services cached by other fog nodes as the originals by migrating replica created by the node who is the owner of the original service

.Each fog node caches replicas of the services in its storage space. In the case, when a fog node requests access to a service, the query is granted in two situations: (i) the original/replica of the service is cached in the fog node. (ii) The original/replica is cached by at least one fog node that is linked to the query issue node through a one-hop/multihop connection. As a result, the query issue node first determines if it has the original/replica of the target service.If it does, the query is automatically fulfilled. If it does not,it diffuses a demand of target service. The query is also effective if it receives a response from other node(s) that carry the original/replica of the target service. Otherwise, the query would be rejected.

In our system environment, we also assume that: F represents the collection of all fog nodes in the system. where k denotes the total number of fog node .

- F=$\{F1, F2, F3..Fk\}$.
- The set of service is denoted by S .
- Fj where j is fog identifier for each fog node in the system. ($1 \leq j \leq k$).

• All services are of the same size, a fog node can cache at least one original service.

• Every fog node has G service storage space for producing replicas, excluding the space for the original service kept by fog node.

• We shall suppose, every fog node has service access history and depending on this history in (t) time we predict the future query of service (t+1) as shown in table 1 and 2 [13].

• The services have not been updated; this assumption has been made for the purpose of simplicity.

• Also, suppose we have 10 users and each requests a service from the nearby fog node.

Table 1 . Prediction based on service history.

service	FN1	FN2	FN3	FN4	FN5	FN6
S1	63	23	15	20	29	22
S2	42	60	39	38	40	44
S3	33	42	48	23	43	35
S4	29	13	8	58	7	8
S5	49	39	41	36	69	18
S6	6	7	3	13	18	60
S7	36	30	35	31	38	30
S8	20	31	19	21	22	15
S9	16	14	17	15	22	19
S10	7	6	4	9	10	7

• Our aim is to find a solution that offers the best accessibility of service and quality of service (network congestion, Energy consumption and response time). When taking into account the parameter of Prediction based on service history. For this we propose three methods as mentioned below. We take in the consideration that the service is replicating, migrating and the replicas are caching based on the Prediction of query of service from each fog node to each service. Finally, the duplicate replicas are deleting between all two adjacent nodes in the second method and between clusters in the third method to give the other services a chance to deploy and to be ready to use.

1.2 The replication and migration heuristics

The replication and migration phase enables to precisely identify which services in the system need to be replicated or migrated. This step allows defining the number of replicas to produce while also identifying the fog node that will be affected by the replication. To address these issues, each services access history is investigated. The benefit of such a technique is that it may be used to detect whether service in a fog node has been recently requested and is more likely to be requested again in the near future. This stage is indispensable in our three methods.

The value of a service is a measurement of how many queries have placed on it. It is crucial information since it indicates the significance of the service. This estimate is based the prediction of service access as shown in table 1.

In addition, we will figure out how many copies are required for each service in each fog node. The following is the formula for calculating the number of replicas of a fog node:

in which:

$$Number\ of\ Replicas = \frac{TNSQ}{VSDP} \quad (1)$$

TNSQ: The total number of service queries made by fog nodesVSDP:The value of a service according to the demand prediction.

1.2.1 • RMSAP (Replication and Migration and Based on Service Ac- accessibility Prediction):

In this heuristic, we assume that six fog nodes (F1, F2, F3, .. F6) exist.

The service is denoted by rectangles.

We also assume service access history of each fog node to each service is shown in the Table 1.

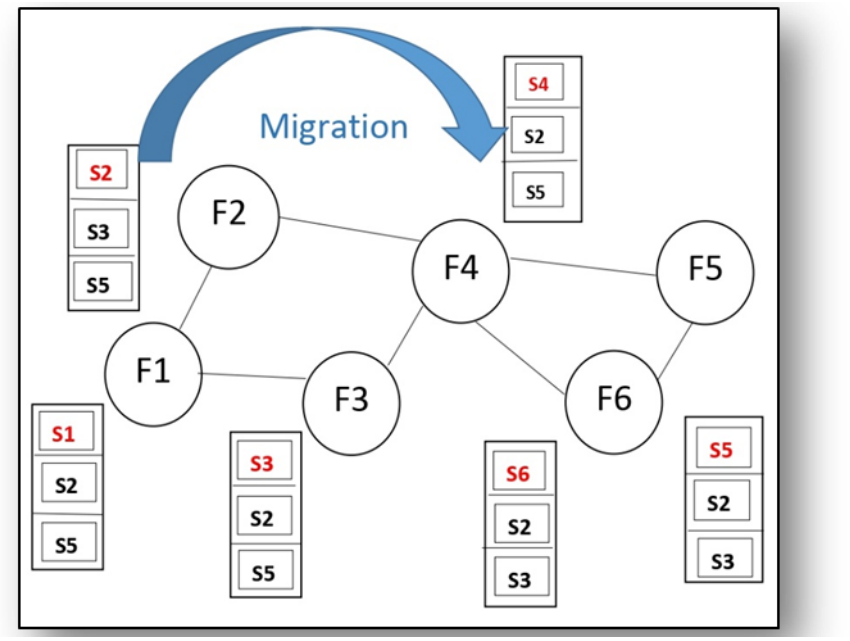

Fig. 2. Execution of RMSAP method.

Every fog node in the RMSAP method create service replicas in descending order of frequencies from service access prediction. A fog node may not connect to another fog node that has a duplicate or original of a service that the fog should store during replica caching. The memory space for the replica is kept free in this situation. When the fog node connects to another fog node that has the original or the replica, the replica is generated at this period. The blue arrow shows the replica migration from the node hosting the original service to the node querying this replica .

The execution of the RMSAP approach is shown in figure 2.

The steps of execution of RMSAP heuristic are below.

#Step 1: Initialization

fog nodes = ['F1', 'F2', 'F3', 'F4', 'F5', 'F6']

Services = {'S1', 'S2', 'S3', 'S4', 'S5'}

service_access_history = {

'F1': {'S1': 5, 'S2': 3, 'S3': 7},

'F2': {'S1': 2, 'S3': 4, 'S4': 6},}

 # Step 2: Service Replication Decision

Replica decision = {}

For fog node:

Access history = service_access_history. Get (fog node, {})

Sorted services = sorted (access_history.items (),

Key=lambda x: x [1], reverse=True)

Replica decision [fog node] = [service for service, _ in sorted services]

 # Step 3: Replica Caching

For fog node, service list in replica_decision.items ():

For service in service list:

If not any (service in replicas [other fog] for other fog! = fog node):

Replicas [fog node].add (service)

 # Step 4: Resource Management

assuming each fog node has a limited capacity

fog_node_capacity,

While l (replicas [fog node]) > fog_node_capacity:

Replicas [fog node] () # Remove least recently used or some criteria

Print ("Final replicas stored in fog nodes: » replicas)

However, fog nodes typically have limited resources that, making it difficult for them to have duplicates of all services.

1.2.2 RMSAPC (Replication and Migration based on Service Accessibility Prediction and contiguous fog nodes):

The RMSAPC approach eliminates replica duplication across directly connected fog nodes to overcome the problem with the first technique that there are numerous replica duplication after migration of all services.

Similar to the RMSAP technique, this approach first determines the migration and the replication of service

mentioned in section 4.2. The algorithm of deleting the duplication of this heuristic is explained as follows:

1- Each fog node uses the RMSAP approach to identify the migration of the replicas.

2- Each fog node transmits its fog identifier as well as information on service ac- cess history. After all fog nodes have completed their diffusion, each fog will be able to determine its directly connected fog node based on the received fog identifiers.

3- The following operation is repeated, in each fog node that is connected to the others. When a service item (original/replica) is duplicated between two fog nodes that are directly connected, and one of them is the original, the fog that holds the replica query the migration of another replica.

4- If there are both replicas between two nodes, the fog with the lower access his- tory to the service change the replica to another replica. When changing the replica, a new service duplicated is selected for migration from among the services whose replicas are not assigned at either of the two fog nodes, where the access history to this service is the highest between all possible services.

During the diffusion period of identifiers between fog nodes.A fog node may not connect to another fog node that possesses an original or a replica of a service that the fog need. In this instance, the replica's memory space is temporarily filled with one of the replicas that have been migrated since the last diffuse period but are not currently being migrated.

This temporary allocated replica is picked from among the available replicas with the highest access history to the replica . The memory space is kept free if there is not a plausible replica to be temporarily assigned. The memory space is filled with the right replica when a service access whose replica to be transferred succeeds.

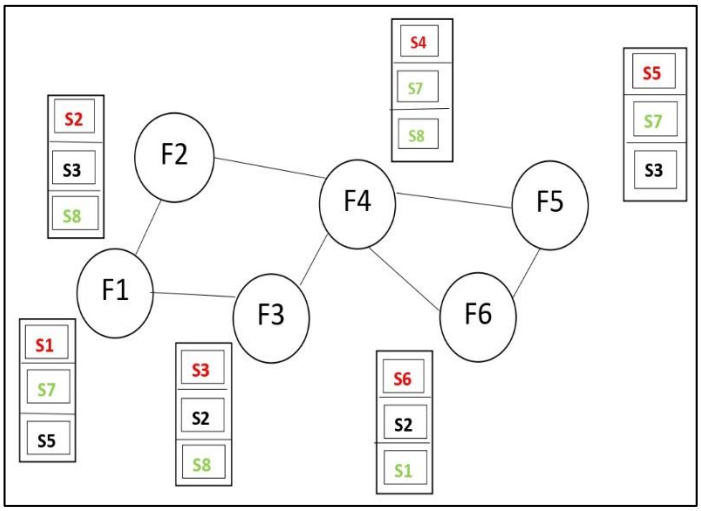

Fig. 3. Execution of RMSAPC method.

Although the RMSAPC approach does not totally reduce duplicate duplication be- tween two directly connected fogs, it does provide the most service accessibility than RMSAP method .

Figure 3 depicts an example of using the RMSAPC approach . A green rectan- gular in Figure 3 denotes a replica that assigned to avoid replica duplication after all migration . We note that duplicate replicas do not exist completely between fog 3 and fog 4,and between fog 4 and fog 6 . Every combination of two contiguous fog nodes results in a change in replica as indicated in the diagram below.

Node1-Node2: S2 → *S*7(*Node*1), *S*5 → *S*8(*Node*2).

*Node*1 − *Node*3 : *S*5 → *S*8(*Node*3). *Node*2 − *Node*4 : *S*2 → *S*7(*Node*4). *Node*3 − *Node*4 : *NOredundancy*. *Node*4 − *Node*5 : *S*5 → *S*8(*Node*1). *Node*4 − *Node*6 : *NOredundancy*.

*Node*5 − *Node*6 : *S*2 → *S*7(*Node*1), *S*5 → *S*8(*Node*6).

1.2.3 RMSAPCC (Replication and Migration Based on Service accessi- bility Prediction with Connectivity based on Clusters :)

The RMSAPC approach, on the other hand, does not totally eliminate replica du- plication between nearby nodes. Figure 3 shows replica duplication between ; node 2 and ; node4 (service 8)and between ; node3 and ; node4, and (service 7) replica duplication between ;4 and ;5.

Moreover when the the user is near to the every two adjacent nodes and request the service 9 or 10. He do not find them because they have not major access frequency between the set of chosen services ,it means there is just 6 kinds of services in the RMSAP method and 8 in the RMSAPC method .

The algorithm of RMSAPCC heuristic is as follows:

1. First , each fog node diffuses its identifier as well as information based on service access prediction . After all fog nodes have completed their diffusion, each node will determine the linked fog nodes based on the received node

identifiers.

2. Each fog node is assigned to a cluster.

3. In each cluster, the access frequency of the cluster to each service is determined as the sum of cluster access frequencies to the service of fog nodes . The fog node in the cluster with the lowest suffix of node identifier performs this calculation.

4. Replicas of services are cached in the order of the access frequencies history of the cluster until all fog nodes in the memory space of the clusters is complete . Replicas of services that fog node hold as originals are not cached here. Each replica is stored on a fog node with the highest access frequency to the service among nodes with enough free memory to construct it.

5. If there is still free memory space at fog nodes in the cluster after caching replicas of all kinds of services, replicas are cached in the order of access frequencies until the memory space is complete. Each replica is stored at a fog node whose service item access frequency is the highest among nodes memory space to build it and do not keep the replica or its original. The replica is not cached if there is no such fog node.A fog node may not connect to another fog node having an original or a replica of one that the node should cache. In this case, the memory space for the replica is temporary filled with another replica, and it is filled with the proper one when a service access to the corresponding service succeeds. Table II is extension the of table I, and it depicts the access frequencies history of the two clusters . The execution of the RMSAPCC method is shown in figure 4 . In this example, two clusters consisting of cluster1(green nodes) and cluster2 (blues nodes)are created.

The purple color in the rectangle in this figure represents a copy that is cached in the second cycle. The RMSAPCC method caches all ten types of copy across the entire of all fog nodes.The service is expected to be more accessible since many several types of replicas can be exchanged.

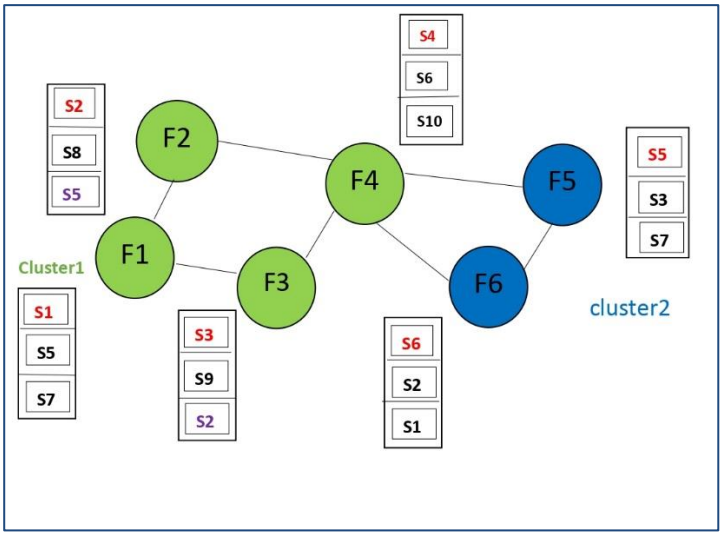

Fig. 4. Execution of RMSAPCC method.

TABLE 2. Prediction based on service history Of Clusters.

service	FN1	FN2	FN3	FN4	FN5	FN6	C1	C2
S1	63	23	15	20	29	22	121	51
S2	42	60	39	38	40	44	179	84
S3	33	42	48	23	43	35	146	78
S4	29	13	8	58	7	8	108	15
S5	49	39	41	36	69	18	165	87
S6	6	7	3	13	18	60	27	78
S7	36	30	35	31	38	30	132	68
S8	20	31	19	21	22	15	91	37
S9	16	14	17	15	22	19	62	41
S10	7	6	4	9	10	7	26	17

5. Simulation and Result Analysis

To meet the objectives of our work,we extended the iFogSim as shown in Figure 5. This Toolkit is made up of numerous classes that we can categorize as follows:

(i) set of components that allow the the physical equipment of the infrastructure to be modeled.

(ii) a set of components that allow the logical parts of the simulated scenario to be modeled (for example the instances of IoT services and their dependencies, ...).

(iii) a set of components that allow the management of the processing in fog nodes. These resources are controlled to improve service latency, network traffic, power consumption, and system running costs by placing and scheduling service instances in physical components.

(iv) A set of components that we have created to model the principal parts of our contribution, such as configuration and service management. These are classes that represent the logical aspects of our contribution (for example, configuration and service management), as indicated in the diagram below.

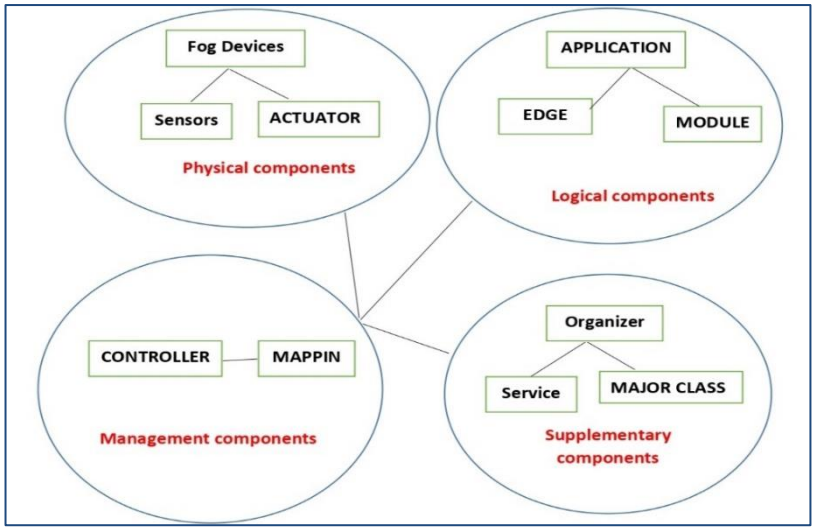

Fig. 5. Design Of IFOGSIM Extension [12].

- MAJOR CLASS: this is the core class, and it provides a collection of options for configuring a Fog and Cloud platform. The logical aspects of the simulated scenario that will be deployed in the infrastructure are likewise created using this class.

- ORGANIZER CLASS: This class has a comprehensive view of the infras- tructure. It specifies a collection of service management strategies for use in the simulated infrastructure. We can also use this class to start our replication and migration approach.

- SERVICE CLASS: (i.e. service deployed in the simulated infrastructure .) It specifies a set of attributes such as its name (for example, service 1), size (in bytes), fog node name hosting the service , and it is type (i.e. original or replica) .In each node there is an array of services.

The units (sensor, fog devices, and actuator) in the iFogSim Toolkit communi- cate with one another using preset events. There are events for launching a service instance in a Fog node, connecting a sensor, and sending service to another compo- nent, for example.

It is worth noting that we have simulating the extra events for specific situations.

To highlight the participation of our methods, we will focus on the three met- rics:(response time, energy consumption and network congestion). In order to evaluate the behavior of our three propositions and to discuss its results obtained through simulation. a series of simulation were launched by the following several parameters.

Table 3 . Parameters in simulation

Parameter	Value
Total number of fog nodes	20
Server bandwidth	8000 Mbps
Access frequency of each fog nodes	$(1 + 0, 01k)$
Number of queries	[100, 300, 500, 700, 900]
Number of service	100
Service size	200 KO

1.3 Average response time

The average response time obtained with the methods applied is measured in this experiment. The average response time for all queries in the simulation is represented by this measure. The formula is as follows:

$$Average\ response\ time = TNQ\ T \quad (2)$$

where:

The total number of queries is represented by TNQ.

T: denotes the amount of time that has passed between the user sending "i" query and receiving a response from the required service.

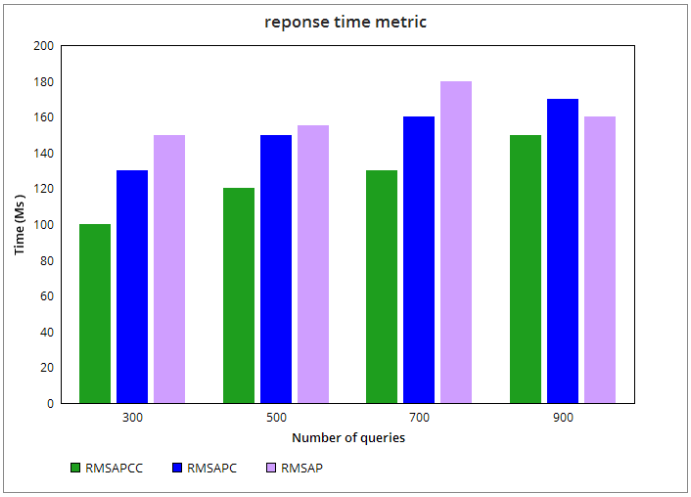

Fig. 6. Response Time metric.

Figure 6 shows a considerable reduction in response time when we use our third technique, which is due to the replication mechanism we used to enhance response time and which practically eliminates duplication while allowing a high number of services to be deployed.

1.4 Energy consumption

We used the metric provided by the iFogSim simulator to analyze the energy use. Figure 7 depicts the simulation results schematically, with the x axis representing the variation in the number of queries and the y axis representing the energy used

. As shown in the simulation results the the third method can be considered as environmental approach among the other ones.

1.5 Network congestion

The x-axis indicates the number of queries, and the y-axis depicts the amount of service transmitted through the network measured in KO, in this series of simulations shown by Figure 8. In all three our techniques, we see a difference in the use of the network. It is worth noting that the iFogSim simulator has this measure as well. The RMSAP and RMSAPC methods have higher Utilisation of the network than the RMSAPCC approach. We can justify this by that fog nodes exchange more information in a wide range .

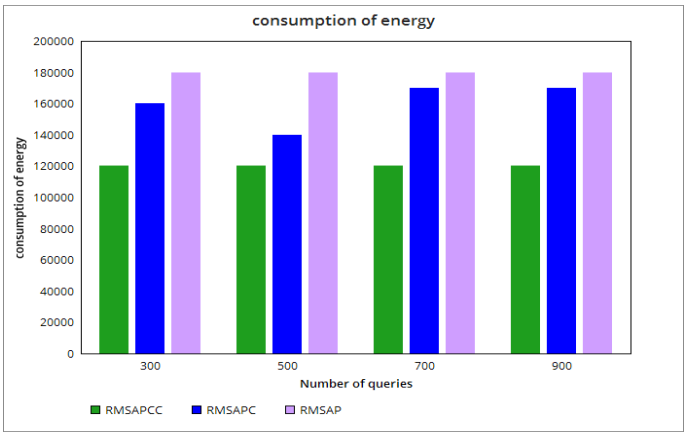

Fig. 7. Consumption of energy.

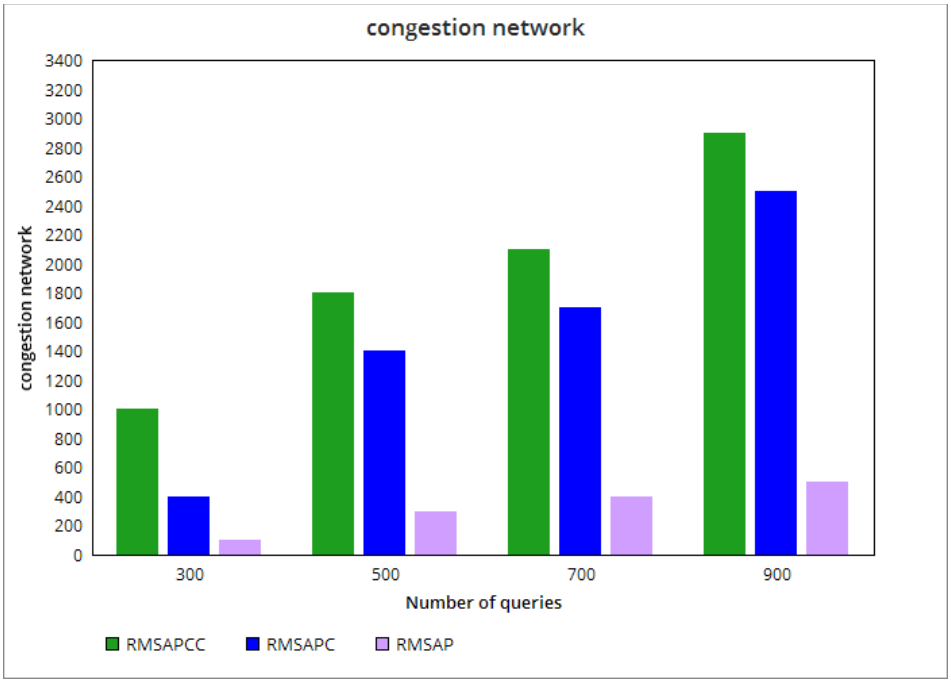

Fig. 8. Congestion Network.

1.6 Additional metric S: Summary of metrics at the same time.

This summary is calculated by the sum of the three previous metrics. In comparison," our third technique delivers a very large benefit by evaluating the three metrics at the same time, as shown in Table 4. Even the other two procedures have positive results. This demonstrates the effectiveness of our replica placement methods we proposed.

TABLE 4. Rate comparisons of each approach versus RMSAPCC approach.

Approach	RMSAP vs. RMSAPCC	RMSAPC vs. RMSAPCC
benefit	5,34%	20%

6. Conclusions and Future Work

In this article, we discussed replication and migration to cloud computing to improve service accessibility with the goal of reducing response time, network congestion, and power consumption. We have suggested three approaches that take into account the prediction of access to services. In the RMSAP heuristic, a fog node migrates aftershocks that we predict to have high accessibility. In the RMSAPC heuristic, the replicas are pre-migrated based on the RMSAP method, and then the replica duplication is eliminated between all contiguous fog nodes to allow other replicas to be positioned at the desired location. Finally, we have suggested the RMSAPCC method as an extension of the RMSAPC approach, which eliminates the duplication of replicas between, clusters and shares many types of services that users need. the simulation results were motivating.

References

[1] GUPTA, HARSHIT And VAHID DASTJERDI, AMIR AND GHOSH, SOUMya K and- BUyya, RajkUMAR, IFOGSIM: A toolkit for modeling and simulation ofresource management techniques in the Internet of Things, Edge and Fogcomputing environ- ments, ,Software: Practice and Experience, 47, 9, p-p=1275–1296, Wiley Online Li- brary. 2017.

[2] ZHANC, BEn and MOR, NITESH And KOLB, JOHn and CHAn, DOUCLAS S and- LUTz,KEn and ALLMAn,ERIC And WawRzynEk,JOHn and LEE,EdwaRD And Kubiatowicz, John, The cloud is not enouch: Saving iot from the cloud, 7thUSENIX Workshop on Hot Topics in Cloud Computing (HotCloud 15),2015.

[3] Sarkar, Subhadeep and Chatterjee, Subarna and Misra, Sudip, IEEE Tran-sactions on Cloud Computing, Assessment of the Suitability of Fog Com-puting in the Context of Internet of Things, 6, 1, p-p46–59, ,IEEE 2015.

[4] Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its rolein the internet of things. In Proceedings of the First Edition of the MCCWorkshop on Mobile Cloud Computing-MCC '12, Helsinki, Finland, 17August 2012; pp. 13–15.

[5] S. YI, Z. HaO, Z. QIn, and Q. LI. FOC COMPUTInC: Platform and applications.In Hot Topics in Web Systems and Technologies (HotWeb), 2015 ThirdIEEE Workshop on, pages 73–78. IEEE, 2015.

[6] M.I. NaaS, J. BOUkHOBza, P.R. PaRVÉDy and L. LEMARCHAnd, An EXTEn- SIOn TO IFOCSIM TO EnaBLE THE DESICn OF DaTA PLACEMEnT STRATECIES, In: 2nd IEEE International Conference on Fog and Edge Computing, ICFEC 2018, Washington DC,USA, May 1–3, 2018, 2018, pp. 1–8.

[7] Mahmud, Redowan and Buyya, Rajkumar, Modelling and simulation of fog and edge computing environments using iFogSim toolkit, Fog and edge computing: Principles and paradigms,1–35,2019, Wiley.

[8] T. Huang, W. Lin, Y. Li, L. He and S. Peng, A latency-aware multiple data replicas placement strategy for fog computing, Journal of Signal Processing Systems 91(10) (2019), 1191–1204.

[9] R. Vales, J. Moura and R.N. Marinheiro, Energy-aware and adaptive fog storage mechanism with data replication ruledby spatio-temporal content popularity, Journal of Network and Computer Applications 135 (2019), 84–96.

[10] Redowan Mahmud and Rajkumar Buyya. Modelling and simulation of fog and edge computing environments using ifogsim toolkit. CoRR, abs/1812.00994,2018.

[11] Berkennou, A., Belalem, G., Limam, S. (2020). A replication and migration strategy on the hierarchical architecture in the fog computing environment. Multiagent and Grid Systems, 16(3), 291-30

[12] BEKKOUCHE, Khawla, *et al*. Effective Strategy Based on Migration and Replication Techniques for Service Management in Fog Environment. *T Regulatory Science (TRS)*, 2023, p. 1531-1549.

[13] Hara, T., & Madria, S. K. (2006). Data replication for improving data accessibility in ad hoc networks. *IEEE transactions on mobile computing*, *5*(11), 1515-1532.