

QoE Aware Application Placement in Fog Environment Using SAW Game Theory Method

M. Mirzapour-Moshizi

Department of Computer Engineering; Islamic Azad University;
Kerman, Iran Email: mmairr_2007@yahoo.com

V. Sattari-Naeini*

[Corresponding Author] Department of Computer Engineering;
Shahid Bahonar University of Kerman; Kerman, Iran;
Email: vsnaeini@uk.ac.ir

Received: 15 Mar. 2022


Revised: 14 May 2022

Accepted: 28 Jun. 2022

Abstract: Cyber-physical Today, fog computing plays an essential role in human life. One of the challenges in the fog and cloud environment is the hierarchical service process. Requests are sent to Fog, and if Fog cannot provide service, they are sent to cloud, which is a time-consuming process. This paper provides a framework that specifies when a request is sent, in which environment it can be serviced, and provides interfaces for properly managing nodes and domains and managing the service of requests. Two new architectures have been presented in the management interfaces. In one of these management interfaces, the most appropriate domain is determined using the SAW method of game theory and user expectations for placing the application. Then, in the other management interface specified in the domain gateway, it suggests the most appropriate node using the PSO algorithm. Since the placement of the application is based on the expectations of the users, it increases the quality of the service. The proposed method has been implemented in iFogSim and its results have been evaluated with authentic articles. It was observed proposed method has better performance and better service speed than the state-of-the-art research works and significant improvement in service response time.

Index Terms: Application Placement, Cloud Computing, Fog Computing, Game Theory, Internet of Things, QoE.

Citation: M. Mirzapour-Moshizi and V. Sattari-Naeini, "QoE Aware Application Placement in Fog Environment Using SAW Game Theory Method," *Journal of Communication Engineering*, vol. 11, no. 1, pp. 137-156, Jan.-Jun. 2024.

 <http://dx.doi.org/10.22070/jce.2024.17679.1243>

I. INTRODUCTION

Internets of Things (IoT) applications are rapidly growing in significant areas of life due to network technologies' rapid inventions and development. With the proliferation of the IoT, the number of devices connected to the Internet is increasing. These devices generate a large amount of data, and we need services such as processing, storage, etc., for them. These operations can rarely be performed within IoT devices, because such devices typically have limited computing resources, storage, and network resources [1]. It is anticipated that until 2025, such systems are projected to have more than 1 trillion IoT devices with a 50% increase in demand for latency-sensitive applications [2]. Therefore, IoT needs to support more powerful resources, and the most common one is the use of Cloud resources.

Cloud is considered a basic computing model to deal with this large number of geo-distributed IoT devices and related applications. However, Cloud data centers are located within several hubs of IoT devices, which increases the effect of latency on both data transfer and application service reception [3]. For latency-sensitive applications such as healthcare and smart city, this long-delayed interaction between IoT devices and Cloud data centers is unacceptable and can drastically reduce the quality of service. In addition, IoT devices can generate vast amounts of data in a minimum of time [2]. On the other hand, if all IoT data are sent to the Cloud for processing or storage, the global Internet will be overloaded [4].

Many researchers have suggested an intermediate layer between measurement sources and the Cloud-IoT architecture called Fog computing. Fog nodes in the Fog layer are used to analyze and operate this diverse volume of data. Fog computing extends Cloud services to the edges of the network, which are used to facilitate access to IoT devices by Cloud capabilities. Fog nodes are geographically distributed and accessible very close to devices. Fog computing helps reduce data transfer time, receive services, and increase data processing by increasing storage and analytical resources. Using Fog, decentralized and distributed Cloud resources are located adjacent to IoT devices and cause to reduce latency and traffic, and overcome many network service problems.

Fog computing, the same as Edge computing, has many advantages, including:

- ◇ Reduces network congestion and load and dramatically reduces the amount of traffic sent to the Cloud [5].
- ◇ Makes decisions and controls IoT devices based on operational policies.
- ◇ Stores data on Cloud servers for subsequent use and to analyze other extensive data [6].
- ◇ Critical applications require real-time data processing to complete the mission.
- ◇ Solves scalability problems caused by increasing the number of endpoints [5].

Therefore, Fog computing plays an essential role in minimizing service delivery delays of various systems with IoT capability and network comfort from dealing with large volumes of data loads [4]. Fig.1 shows the relationship between IoT devices and the Fog and Cloud computing environment. In this paper Visio 2016 is used to create the artworks.

Finding the most suitable Fog instance for embedding an application due to its various parameters is challenging. If user expectations are considered in this placement, it will increase the quality of experience (QoE) about system services.

This paper presents a framework for improving the placement of IoT applications in the Fog environment. For this purpose, some domains consider each domain has some Fog nodes and one gateway. Here it is assumed that the nodes, gateways, and users are fixed. An interface between domains and IoT applications is provided that manages the sending and receiving of IoT applications.

The main steps are as follows:

- ◇ Nodes in the Fog environment are divided into categories according to their proximity to each other, and each category is considered in a domain.
- ◇ Each domain has a gate, which contains the information of all nodes of that domain.
- ◇ Some management interfaces are considered. These interfaces will manage their adjacent domains
- ◇ The management interface has some information about the domain nodes in its service range.
- ◇ Each user submits their applications to their respective management interface.
- ◇ According to the information in the management interface and the applications' specifications, with the help of a game theory model, it is determined which domain can serve each application.
- ◇ After determining the appropriate domain to provide the service to an application, the most suitable node domain is determined using a placement policy on that gateway.

The rest of the paper is organized as follows: In Section 2, related works are reviewed. In section 3 the problem is explained. In section 4, the proposed work is described in detail. The illustrative example and simulation results are shown in Section 5. In section 6 of the article, the conclusion is presented.

The basic parameters for expressing applications and virtual machines in the Cloud and Fog nodes are summarized in Table 1.

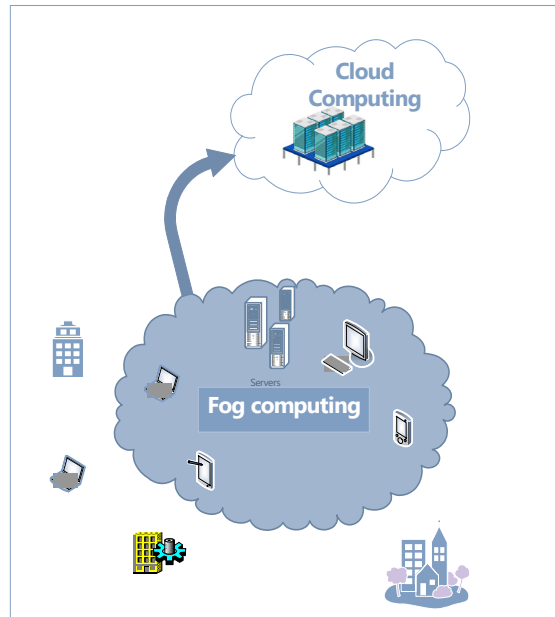


Fig. 1. The relationship between IoT devices and the Fog and Cloud computing environment

Table 1. Basic parameters: symbols and descriptions

symbol	Description
L	Application length
RTTi	Proximity of the application with GWMi
PT	Time required to process a task
VRTT	VM distance to GWM desired
PS	Processing Speed
PC	Power Consumption
S	VM storage space per unit of application
lengthTHi	The Application length threshold in GWMi
CS	Cloudlet storage required
C	User suggested cost

II. RELATED WORKS

Many articles have been presented in the field of fog and clouds. In this article, special attention has been paid to articles that are more consistent with our purpose.

A. Articles submitted in order to application placement in the cloud environment

There have been a lot of works and articles in the field of cloud, and many of them have solved some challenges in this environment by providing methods. Researchers studied various areas such as application placement, security, increasing response time, load balancing, request management, resource management, network optimization, etc. Here are reviewed articles that aim to put the application in the right place in the cloud.

Spinnewyn et al. [7] deal with geographically distributed Clouds and try to reduce response time by providing solutions and using optimization algorithms. Nardin et al. [8] use a cloud-based micro service application to reduce energy consumption and allocate the most appropriate resource to an application. In an article, ChoI [9] presents an algorithm for deploying a virtual machine to save energy in a cloud data center, taking into account the prevention of high heat generation and server reliability. Badri et al. [10] modeled the problem of energy-aware application placement in edge computing systems as a multistage stochastic program, and their goal is to maximize the QoS of the system and take into account the limited energy of edge servers. In their paper, Lackow et al. [11] present a strategy to investigate the placement of requests across the cloud. Li et al. [12] present dynamic multi-objective optimized relocation and migration strategies for applications in the cloud environment. They have used fast sorting genetic algorithm and shown that their placement algorithm minimizes effective network utilization, reduces response time and ensures load balance of data nodes. It also effectively reduces migration time when migration is required, minimizes response time, and improves network resource utilization. In their paper, Elgamal et al. [13] propose a scalable dynamic programming algorithm called DROPLET to distribute operations in IoT applications across cloud resources while minimizing response time. Xu et al. [14] first analyze the resource usage, energy consumption, and data access time in the cloud data center with a topology, then propose an application placement method based on the non-dominated sorting genetic algorithm. Elhoseny et al. [15] present a new model to optimize the selection of virtual machines in cloud health service applications in order to properly manage big data. Farhadi et al. [16] use an algorithm for placing requests with the aim of optimization. Mishra et al. [17] present an algorithm for placing requests on virtual machines in the cloud, where tasks are classified according to their resource requirements, and then searches for the appropriate virtual machine.

B. Articles submitted in order to application placement in the fog environment

In this section, articles have been reviewed that aim to application placement in fog environment. Mouradian et al. [18] consider FNs to be mobile and based on that, they calculate the cost of

program implementation. Brogi et al. [19] present methods for solving the problem of application placement in Fog or Cloud and identify some challenges. Kim et al. [20] propose a user-based participatory Fog computing architecture based on motivation. Mahmud et al. [21] propose an informed application policy for integrated Fog-Cloud environments, which increases profits and guarantees QoS. Goudarzi et al. [22] present a model that minimizes Internet device response time, energy consumption, and possible migrations. Also, a clustering technique to enable the execution of shared tasks among servers and an application placement technique to minimize costs in implementing IoT applications are presented. Kayal et al. [23] present a distributed location strategy that optimizes the energy consumption of Fog nodes and the communication costs of applications. Xia et al. [24] present a model, an objective function, and a mechanism for addressing the problem of IoT-distributed applications in Fog. The proposed model can deal with large-scale problems and reduce the response time of applications. Baranwal et al. [25], in their work, Mohammad and colleagues propose a QoE-aware application placement policy in Fog computing using a modified TOPSIS that prioritizes applications and Fog instances, respectively, based on their expectations and computational capability for placement. Mann [26] does application placement for individual Fog colonies and reduces the scalability problem. Smani et al. [27] propose a resource-aware multilayer partitioning method to minimize resource wastage and maximize service placement and deadline satisfaction rate in a Fog environment. Mahmud et al. [28] place programs in appropriate Fog instances and increases QoE. Aldossary [29] has proposed an efficient optimization approach for placing IoT applications in a multi-layer fog-cloud environment using a mathematical model. In an article, Canali et al. [30] propose a genetic algorithm-based optimization model for placing time-sensitive requests on fog infrastructure. Mirampalli et al. [31] present a QoE-aware strategy for placing requests on nodes based on fuzzy logic in a fog environment. In their method, they consider user expectations in requests and fog nodes and determine the priority of applications using fuzzy logic. Samani et al. [32] in their paper propose a multi-layer resource-aware partitioning method that minimizes resource wastage and response time. In order to analyze the scheduling and allocation problem of fog nodes, Sabireen et al. [33] propose a new hybrid model based on a meta-heuristic approach called advanced multi-objective particle swarm optimization with clustering and fog selector. Li et al. [34] present a novel cost-effective and QoS-aware approach to query placement in fog environments and show that their proposed algorithm improves cost, response time, and energy consumption. In an article by Zare et al. [35], A3C algorithm is used as a new deep reinforcement learning approach to solve the appropriate placement of requests.

A review of recent articles shows that the realization of IoT services depends on providing

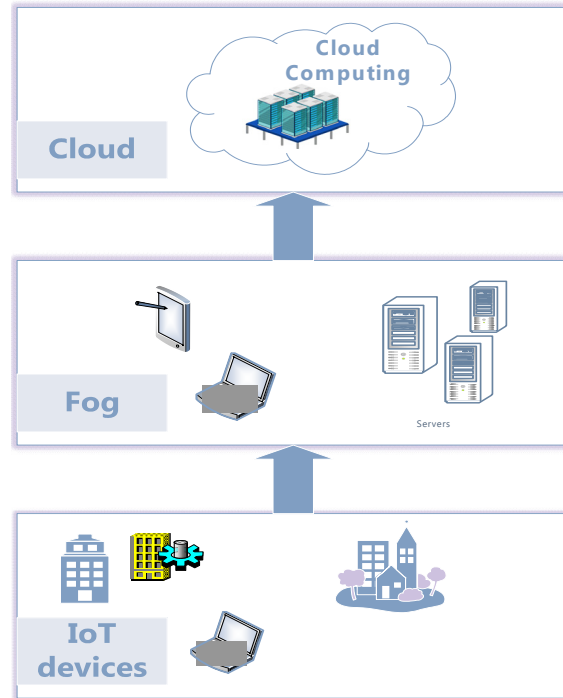


Fig. 2. Process of servicing the requests

a new infrastructure to meet the needs of these applications. Fog computing has emerged to answer this important need. Due to the increasing number of IoT requests with heterogeneous characteristics and on the other hand due to the heterogeneous resources of the fog environment, determining the appropriate

place to serve the requests is considered a challenge. On the other hand, many IoT requests require a low response time, and attention to this time is very significant in the fog environment. In this paper, a framework for processing IoT requests is proposed, in which user expectations and heterogeneous characteristics of fog nodes are considered in the appropriate and optimal placement of requests.

III. PROBLEM DESCRIPTION

Fog computing is a computational model that essentially extends the Cloud services to the network's edge [5]. As seen in Fig.2, the process of servicing requests is hierarchical. Every request that is sent enters the Fog environment; if the Fog nodes cannot serve it, it is sent to the Cloud:

Requests that cannot be served by Fog are sent to the Cloud with a delay. This delay is often inconvenient, especially for real-time requests. In this article, considering this challenge, a policy has been considered to determine by sending a request from the user at the very beginning whether

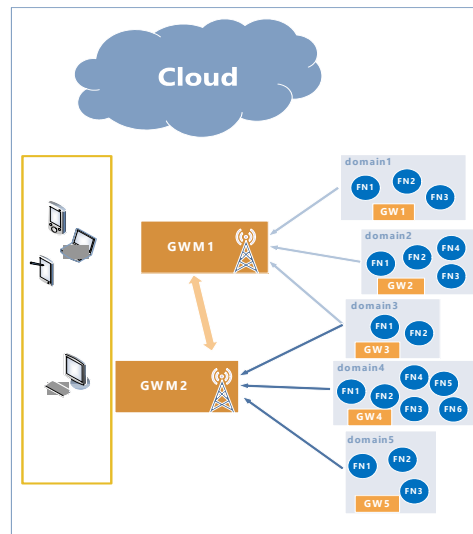


Fig. 3. General processed framework

the nodes can handle that request or should be sent to the Cloud.

Because the Fog environment is distributed, domains have been considered to manage this environment better. Some management interfaces are considered between the user and the domains, in which special attention is paid to the user's expectations. A vital feature considered in the user's expectations is the amount of money he wants to spend to service his request. This article presents a context in which all the challenges and expectations will be addressed.

IV. SYSTEM OVERVIEW

In this section, the proposed method and different parts of the proposed architecture are described in detail. In section 4.1, the proposed architecture is generally described. In section 4.2, the interior gateways architecture of each domain is checked in detail. Section 4.3 describes the internal architecture of the Management interfaces in detail. In section 4.4, the steps of processing a task in the proposed framework are explained thoroughly.

A. Suggested Method

Fig. 3 shows the work of our article, which will be explained separately in the following. Adjacent Fog nodes (FNs) are considered in a domain. Each domain has a gateway called GW that contains the information of the nodes within each domain. Between domains and IoTs, interfaces called GW Management (GWM) are shown, which are associated with some domains in their range. A request will be sent from IoT devices. GWM, with which the request is in a range, receives the request features. GWM uses domain information related to its subset, request specifications, and game theory to determine which domain can serve the request. Once the domain is specified, the

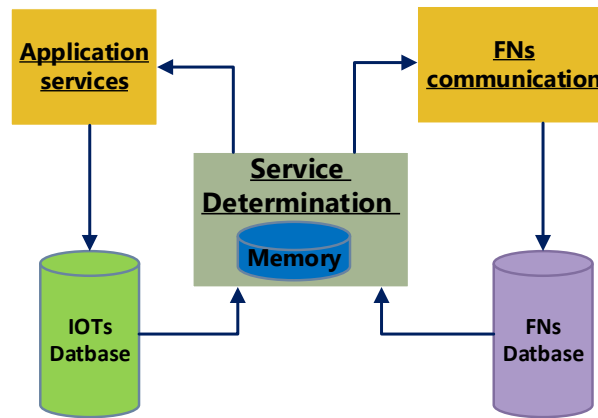


Fig. 4. Internal structure of a GW

request details are sent to that domain's gateway, where the job is sent directly to that node after selecting the most appropriate node. If the request is not serviceable in the domains, it will be sent to the Cloud initially. In this article, the response time and energy consumption for processing application on the selected node are calculated and evaluated.

B. GW Architecture

GW is responsible for managing each domain. The information of all the nodes of each domain is in its GW. Each gateway is connected with the domain nodes on one side and the GWMs in their range on the other. The internal architecture of a GW is shown in Fig. 4.

Each GW consists of the following parts:

- ◇ **FNs communication:** Information, data, and programs are transferred through this unit between nodes and GW.
- ◇ **FNs Database:** It is a database that stores information about nodes in that domain. This information includes:
 - ◆ The proximity of each node to GW that calculated by Round trip time
 - ◆ Processing Speed of FNs Which is expressed with Instruction per second (IPS)
 - ◆ Free storage of FNs
- ◇ **Application Services (AS):** This unit exchanges information and data between GWM and GW.
- ◇ **IoT Database:** The information of the programs that this domain will be served will be stored in this database. This information includes:
 - ◆ The proximity of each task to GWM that calculated by Round trip time
 - ◆ App Length
 - ◆ Cost
 - ◆ App Processing Time

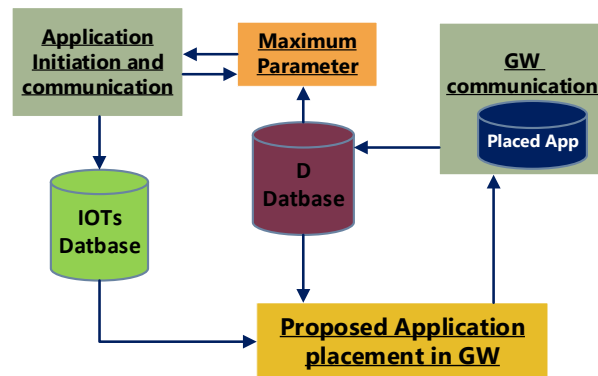


Fig. 5. Internal structure of a GWM

- ◇ **Service Determination (SD):** In this unit, the most appropriate node for the service is determined for each request using the FNs Database and the placement policy. Specifying the proper node sends a message to the AS and the original program stored in Memory to the desired node through the FNs communication component to place and provide the service. Here you can use any placement policy.

C. GWM Architecture

The management of the requests sent between users and Fog nodes is with the GWM interface. The internal architecture of a GW is shown in Fig. 5.

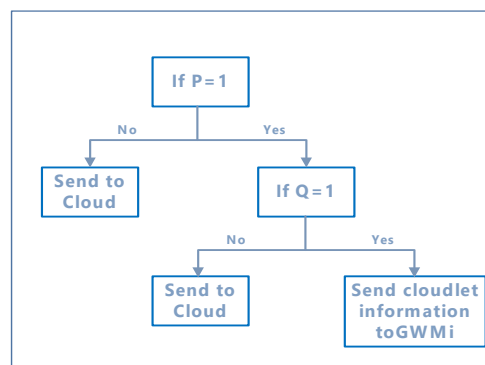


Fig. 6. Decision tree for determining the place of service

Each GWM has several sections as follows:

- ◇ **Application Initiation and Communication:** Requests within this interface's scope are specified through this unit.
- ◇ **GW Communication:** Through this unit, communication is established between GWM and GWs. In Placed App section, the applications that are being serviced are specified. The information in this section is used to search or is migrated applications.
- ◇ **D Database:** Essential information of nodes in each domain within the scope of this

management interface is stored in this database. This information includes:

- ◆ Processing Speed of FNs
- ◆ Storage of FNs
- ◆ User suggested cost
- ◇ **Maximum Parameter:** In this unit, the threshold for the App length and the required resources are considered using the information in the D Database. Each program is initially determined to be sent to Fog or Cloud using a decision tree based on the parameters specified in this section.
- ◇ **IoTs Database:** The information of all requests within the scope of a GWM and specified that Fog can handle them are stored in this database. This information includes:
 - ◆ The proximity of each task to GWM that calculated by Round trip time
 - ◆ App Length
 - ◆ App Processing Time(real Time)
- ◇ **Proposed Application placement in GWs (PAP):** Using two databases, IoTs Database and D Database, and a game theory model determines which domain each application should be sent for service.

D. Steps of Processing a Request in Proposed Architecture

Each request sent by the IoT device is received by the Application Initiation and Communication Unit, GWM, within its scope. This request is obtained using a decision tree. With the help of the information in the Maximum Parameter, it is checked at the beginning whether this request can be serviced with these parameters in the sub-domains and the range of this GWM or not. If it is declared non-serviceable, it is sent to the Cloud at the beginning. Fig.6 shows the desired decision tree for determining the place of service.

Equation (1) calculates the P value for the time it takes to service a request in the decision tree.

$$P = \begin{cases} 1 & \text{if } r \leq PT \\ 0 & \text{if } r > PT \end{cases} \quad (1)$$

where r is the time required to run an application with a length of L according to the maximum PS of a domain. The value of r is calculated by equation (2).

$$r = L / PS_{max} \quad (2)$$

If r is longer than the PT of the job, this job will be sent to the Cloud at the beginning. Otherwise, the value of Q is obtained. The Q value is used for the memory required for a request in the decision tree and is described by equation (3)

$$Q = \begin{cases} 1 & \text{if } m \leq CS \\ 0 & \text{if } m > CS \end{cases} \quad (3)$$

where m is the memory required to run an application, with a length of L according to the maximum S of a domain. The value of m can be calculated by equation (4):

$$m = L * S_{max} \quad (4)$$

If m is larger than its CS, the job will be sent to the Cloud. Otherwise, it turns out that this domain can serve this job. The PS_{max} and S_{max} are updated according to Algorithm 1 in the desired domain. If Fog can service the job, some of the information in this request is stored in the IoTs Database. In PAP in GWs, using the two databases IoTs Database, D Database and SAW method, the most suitable domain that can handle this request is determined.

For this purpose, we will obtain a series of weights from the D Database for each domain using the following formulas. We consider the information in the D Database as the following matrix:

$$D = \begin{bmatrix} d_{11} & \cdots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{m1} & \cdots & d_{mn} \end{bmatrix} \quad (5)$$

represents the number of domains and represents the number of parameters of each domain.

We use equation (6) to get each domain's weight.

$$\omega_{ij} = \vartheta_{ij} / \sum_{j=1}^n \vartheta_{ij} \quad (6)$$

$$\vartheta_{ij} = I - \lambda_{ij} \quad (7)$$

$$\lambda_{ij} = \left[\frac{-1}{\ln(n)} \right] * \sum_{i=1}^m \Omega_{ij} \ln(\Omega_{ij}) \quad (8)$$

$$\Omega_{ij} = \frac{d_{ij}^*}{\sqrt{\sum_{i=1}^m (d_{ij}^*)^2}} \quad (9)$$

$$d_{ij}^* = \frac{1}{d_{ij}} \quad (10)$$

This weight is obtained for each domain. We use equation (11) to find the most suitable domain for a request according to the features, especially the proposed costs:

$$\gamma = 1 / \sum_{j=1}^n f_{ij} \omega_{ij} \quad (11)$$

Where is the IoTs Database matrix.

The goal is the smallest value for, which determines in which domain there are more suitable conditions for serving a request.

After determining the domain, the desired request is sent to GW Communication, and through this unit, it is sent to the GW related to the specified domain. In the GW, the request is received by the Application Services and stored in the IoT Database of that GW. The most suitable node for the service to the request is determined in the Service Determination, using the FNs Database and the desired placement policy that we have used the PSO algorithm. Selecting the appropriate node sends a message to the Application Services, and the request stored in the memory is sent

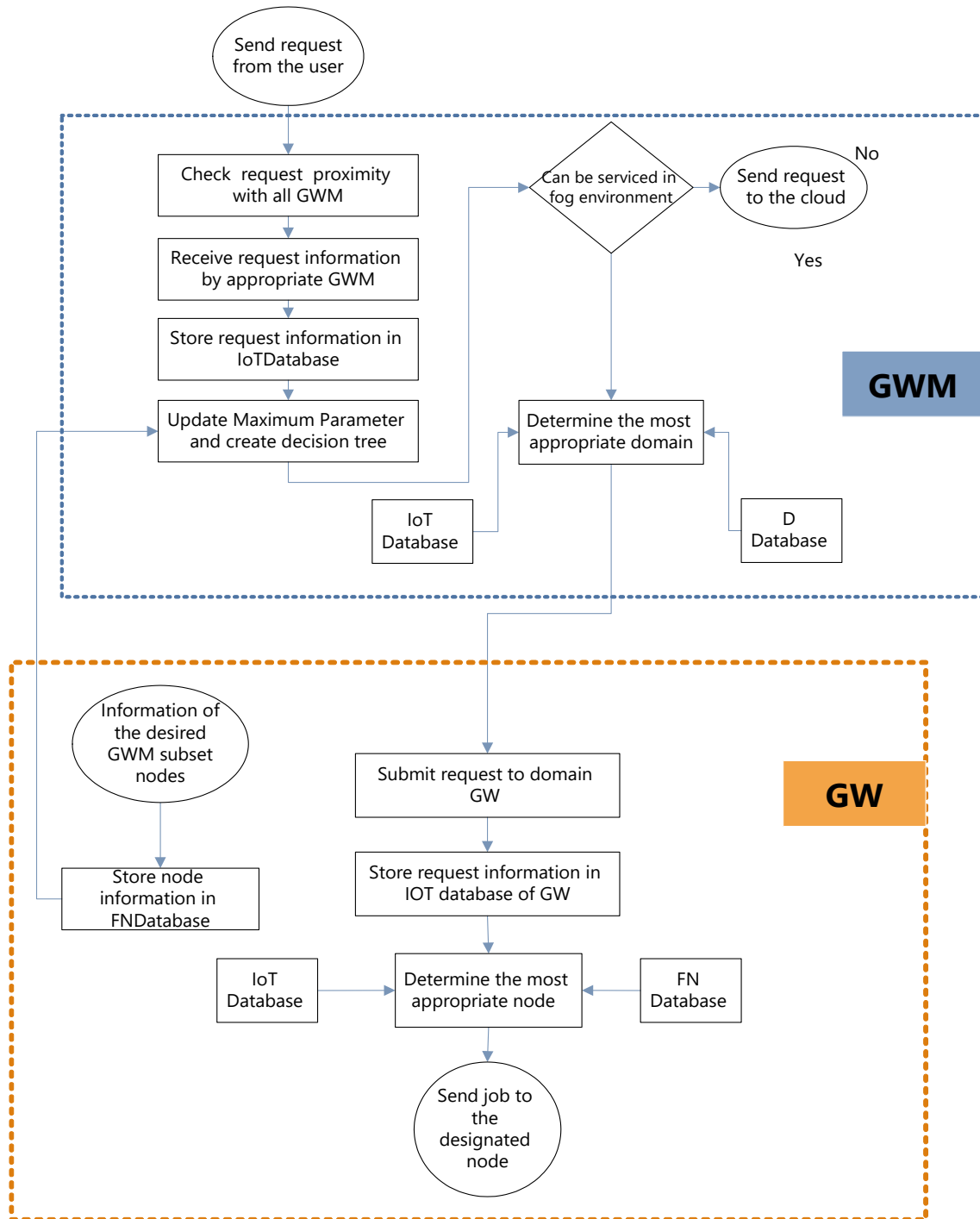


Fig. 7. Flowchart of work steps in the proposed architecture

to the desired node through the FNs communication unit to place and provide the service. When sending a message to the Application Services, the job specifications, which include the complete information of the request and the node designated to serve this request, are stored in the Placed App database in the GW Communication in the relevant GWM. After this, the information of

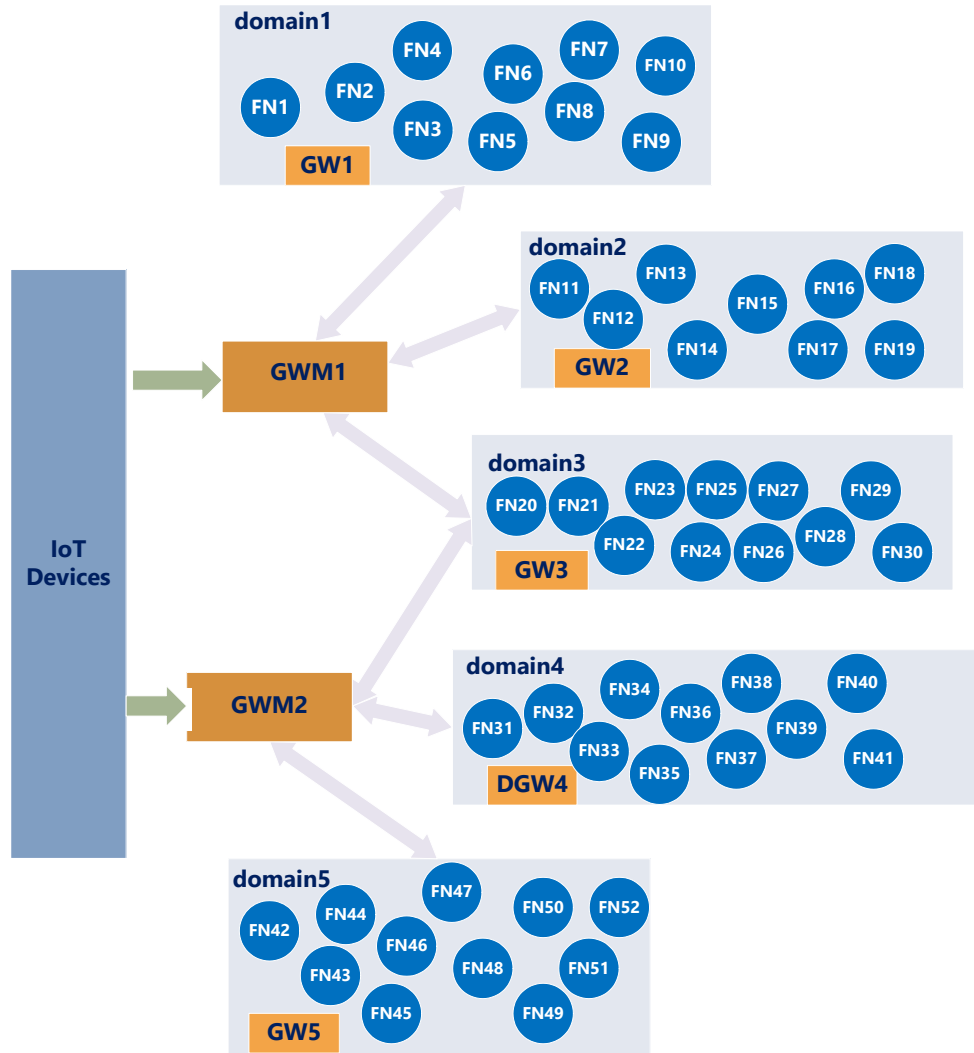


Fig. 8. Simulated Fog environment

requests is deleted from both databases in GWM and GM.

The response time and energy consumption for processing request on the selected node are calculated and evaluated. The energy used to process request on the selected node is calculated based on equ. (12).

$$E = L * PS \quad (12)$$

The processing step of a Cloudlet in the proposed architecture is shown in Fig. 7.

V. ILLUSTRATIVE EXAMPLE

In this part, the proposed method with a different number of inputs and various features and some Fog nodes is implemented in iFogSim simulator environment, and simulation results are shown.

Table 2. Nodes characteristics

VM	VRTT	PC	S	PS	VM	VRTT	PC	S	PS
#1	100	10	30	80	#27	97	14	25	84
#2	150	15	15	65	#28	52	17	20	67
#3	45	4	10	100	#29	210	8	9	75
#4	300	18	11	90	#30	150	40	24	97
#5	70	9	55	98	#31	47	12	16	105
#6	99	23	40	102	#32	301	7	86	73
#7	51	18	59	68	#33	69	5	19	69
#8	81	23	33	73	#34	46	13	42	31
#9	77	17	12	36	#35	100	19	29	79
#10	69	9	10	74	#36	52	7	20	88
#11	92	12	25	91	#37	70	26	11	77
#12	189	3	15	74	#38	94	15	70	66
#13	209	10	90	62	#39	191	20	22	101
#14	84	42	46	105	#40	220	12	29	75
#15	137	23	25	83	#41	87	25	15	63
#16	49	11	15	71	#42	162	19	88	51
#17	72	15	16	71	#43	140	6	18	100
#18	201	20	79	87	#44	196	14	27	58
#19	91	16	46	82	#45	288	10	28	72
#20	190	18	39	48	#46	44	24	13	69
#21	148	33	18	96	#47	150	15	40	47
#22	200	25	34	85	#48	66	9	33	93
#23	82	12	48	61	#49	84	18	19	55
#24	140	31	23	98	#50	123	11	22	74
#25	299	14	31	70	#51	61	30	48	102
#26	211	19	13	92	#52	128	17	15	56

The specifications of all nodes are shown in Table 2.

Table 3. Jobs characteristics

Cloudlet	L	RTT1	RTT2	PT	CS	C
#1	1000	20	19	400	20000	58000
#2	3500	14	85	600	38000	32000
#3	1800	22	28	200	60000	75000
#4	8000	50	36	700	2000	142000
#5	2000	15	12	400	50000	63000
#6	4000	81	88	500	51000	89000

Table 4. average parameters of domains

Domain	PS	S	PC	VRTT
#1	79	29	15	108
#2	80	36	16	122
#3	75	27.5	21	156
#4	75	15	15	116
#5	69.5	32	16	131

Table 5. weights of domains

Domain	
#1	0.2500
#2	0.2456
#3	0.2502
#4	0.2503
#5	0.2536

Table 6. value of Cloudlets

Cloudlet	domain#1	domain#2	domain#3	domain#4	domain#5
#1	-	-	0.050313	0.050293	0.049638
#2	0.053909	0.054875	0.053866	-	-
#3	0.029186	0.029709	0.029163	-	-
#4	Cloud				
#5	-	-	0.034626	0.034612	0.034162
#6	0.027649	0.028145	0.027627	-	-

The jobs reviewed in this article are considered with the specifications of Table 3.

A. Framework Simulation

To evaluate the proposed framework, a Fog environment like Fig. 8 is simulated.

As shown in Fig. 8, two GWM units and five domains are considered in this simulation. The number of nodes in each domain is shown in the figure. In this implementation, only one VM is considered in each node. RDGWM1=8, RDGWM2=5, ST=0.02, lengthTH1=5000, and lengthTH2=7000 are considered.

Table 7. The designated domain for each Cloudlet

Cloudlet	Domain
#1	#5
#2	#3
#3	#3
#4	Cloud
#5	#5
#6	#1

Table 8. Simulation results

Cloudlet	GWM	Domain	Vm	Response time(ms)	E
#1	2	5	46	21.74	69000
#2	1	3	21	45.01	161000
#3	1	3	29	18.01	135000
#4		Cloud		1.68	-
#5	2	5	42	23.53	80000
#6	1	1	2	40.83	260000

B. Simulation Result

This simulation is done in iFogSim. The unit of response time in the simulation result is milliseconds. Using Table 2 and Table 3, simulation has been done. The average parameters of each domain are shown in Table 4. According to Table 4, the weights of each domain are shown in Table 5. By entering the Cloudlet information with the help of the decision tree in the GWM interfaces at the beginning of the work, it is determined that Cloudlet #4 cannot be surveyed in the Fog environment and will be sent to the Cloud. After it is determined that the Cloudlet can be serviced in the Fog environment, the γ values of each Cloudlet are defined in the corresponding domains of each GWM. The γ values of each Cloudlet in each domain are shown in Table 6.

The lowest value of γ for each Cloudlet determines to which domain the Cloudlet is sent. As can be seen, no value of γ is calculated for Cloudlet #4, which is sent to the Cloud. If a domain that does not have a free node is determined for a Cloudlet, the Cloudlet will be sent to the next appropriate domain. In Table 7, the designated domain for each Cloudlet is specified: Cloudlets are sent to the GW of the specified domain to determine the most suitable node for placement. With the help of the PSO algorithm, the most appropriate node for placement is selected in the desired GW. Table 8 shows the most suitable node and response time for each Cloudlet.

To better evaluate this work, we also simulated [25] and [28] using the data in this article and are calculated the response time. Simulation [28] and [14] are performed using Table 2 and Table 3 in this paper, and their results are shown in Table 9 and Table 10.

Fig. 9 and Fig. 10 show comparison charts of the framework presented in this paper and [25] and [28] using the data in Table 2 and Table 3.

As can be seen in Fig. 9, and Fig.10 the response time and energy consumption of the model proposed in this article are much less than the response time and energy consumption of models presented in [25] and [28]. In [25] and [28], with the increase in the number of nodes or the number of Cloudlets or the evaluated parameters, the calculations and the response time increase significantly. Also, in [25] and [28], although Cloudlet #4 should be sent to the Cloud, results are obtained after all calculations for all nodes and Cloudlets, and if this Cloudlet is real-time, this time is not acceptable for this.

IV. CONCLUSION

IoT devices are increasing, and Fog is one of the most suitable environments for processing, storage, etc., for these applications. With the increase of such requests or the number of Fog nodes, to improve the proper servicing of a large volume of requests, a policy should be considered that these requests are placed in the most suitable node according to their characteristics. In this paper, a framework is presented in which we can consider any number of requests and any number of nodes in the Fog environment. With the help of the considered management interfaces, we can provide services in a suitable response time. In the proposed method, entering a request in the Fog environment before sending the request to the gateways determines at the earliest possible time whether the Fog can handle the request or should be sent to the Cloud. The simulation results show that the proposed method has a suitable and acceptable performance, and in contrast to the other method, it shows better and more efficient results. In choosing the correct node for placement, we also consider the user's expectations, which increases the system's QoE. In the future, we plan to evaluate the mobility of users or nodes in this framework and consider more parameters in this work.

REFERENCES

- [1] F. C. Delicato, P. F. Pires, and T. Batista, "Resource Management for Internet of Things," *Part of the Springer Briefs in Computer Science book series (BRIEFSCOMPUTER)*, Springer Cham, pp. 7-18, 2017,
- [2] S. Tuli, R. Mahmud, S. Tuli, and R. Buyya, "FogBus: A Blockchain-based Lightweight Framework for Edge and Fog Computing," *Journal of Systems and Software- Elsevier*, vol. 154, pp. 22-36, Aug. 2019.

- [3] M. Afrin, M. R. Mahmud, and M. A. Razzaque, "Real time detection of speed breakers and warning system for on-road drivers," *Proc. of the IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*, pp. 495–498, Dec. 2015.
- [4] R. Mahmud and R. Buyya, "Modelling and Simulation of Fog and Edge Computing Environments using iFogSim Toolkit," *Fog and Edge Computing: Principles and Paradigms*, R. Buyya and S. N. Srirama, Wiley, pp. 433-464, 2019.
- [5] A.V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, and R. Buyya, "Fog Computing: principles, architectures, and applications. in Internet of Things," *Principles and Paradigms*, pp. 61-75, 2016.
- [6] P. B. Soundarabai and P. R. Chellaiah, "Mechanisms Towards Enhanced Quality of Experience (QoE) in Fog Computing Environments," *Fog Computing*. Springer, Cham, July 2018.
- [7] B. Spinnewyn, R. Mennes, J. F. Botero, and S. Latré, "Resilient application placement for geo-distributed cloud networks," *Journal of Network and Computer Applications*, vol. 85, pp. 14-31, May 2017.
- [8] I. Fontana de Nardin, R. da Rosa Righi, T. R. Lopes, C. A. Costa, H. Yeom, and H. Köstler, "On revisiting energy and performance in microservices applications," *Parallel Computing*, vol. 108, 102858, Dec. 2021.
- [9] J. Y. Choi, "Virtual machine placement algorithm for energy saving and reliability of servers in cloud data centers," *Journal of Network and Systems Management*, vol. 27, pp. 149–165, June 2018.
- [10] H. Badri, T. Bahreini, D. Grosu, and K Yang, "Energy-Aware Application Placement in Mobile Edge Computing: A Stochastic Optimization Approach," *IEEE Trans. Parallel and Distributed Systems*, vol. 31, no. 4, pp. 909-922, April 2020.
- [11] A. Luckow, K. Rattan and S. Jha, "Exploring Task Placement for Edge-to-Cloud Applications using Emulation," *IEEE 5th International Conference on Fog and Edge Computing (ICFEC)*, pp. 79-83, May 2021.
- [12] C. h. Li, Y. Wang, H. Tang, and Y. Luo, "Dynamic multi-objective optimized replica placement and migration strategies for SaaS applications in edge cloud," *Future Generation Computer Systems*, vol. 100, pp. 921-937, Nov. 2019.
- [13] T. Elgamel, A. Sandur, P. H. Nguyen, K. Nahrstedt and G. Agha, "DROPLET: Distributed Operator Placement for IoT Applications Spanning Edge and Cloud Resources," *IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp. 1-8, July 2018.
- [14] X. Xu, S. Fu, L. Qi, X. Zhang, Q. Liu, Q. He, and S. Li, "An IoT-Oriented data placement method with privacy preservation in cloud environment," *Journal of Network and Computer Applications*, vol. 124, pp. 148-157, Dec. 2018.
- [15] M. Elhoseny, A. Abdelaziz, A. Salama, A. Riad, K. Muhammad, and A. Sangaiah, "A hybrid model of Internet of Things and cloud computing to manage big data in health services applications," *Future Generation Computer Systems*, vol. 86, pp. 1383-1394, Sept. 2018.
- [16] V. Farhadi, F. Mehmeti, T. He, T. La Porta, H. Khamfroush, S. Wang, K. Chan, and K. Poularakis, "Service Placement and Request Scheduling for Data-Intensive Applications in Edge Clouds," *IEEE/ACM Trans. Networking*, vol. 29, no. 2, pp. 779 – 792, Feb. 2021.
- [17] S. K. Mishra, D. Puthal, B. Sahoo, P. Jayaraman, S. Jun, A. Zomaya, and R. Ranjan, "Energy-efficient VM-placement in cloud data center," *Sustainable Computing: Informatics and Systems*, vol. 20, pp. 48-55, Dec. 2018.
- [18] C. Mouradian, S. Kianpisheh, M. Abu-Lebdeh, F. Ebrahimnezhad, N. TahghighJahromi, and R. H. Glitho, "Application component placement in NFV-based hybrid cloud/fog systems with mobile fog nodes", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 5, pp. 1130-1143, March 2019.
- [19] A. Brogi, S. Forti, C. Guerrero, and I. Lera, "How to place your apps in the fog state of the art and open challenges," *Software Tools and Techniques for Fog and Edge Computing*, vol. 50, pp. 719-740, Nov. 2019.

- [20] W. S. Kim and S. H. Chung, "User incentive model and its optimization scheme in user-participatory Fog computing environment," *Computer Networks*, vol. 145, pp. 76-88, Nov. 2018.
- [21] R. Mahmud, S. N. Srirama, K. Ramamohanarao, and R. Buyya, "Profit-aware application placement for integrated Fog-Cloud computing environments," *Journal of Parallel and Distributed Computing*, vol. 135, pp. 177-190, Jan. 2020.
- [22] M. Goudarzi, M. Palaniswami and R. Buyya, "A Distributed application placement and migration management techniques for edge and fog computing environments," 16th Conference on Computer Science and Intelligence Systems (FedCSIS), Oct. 2021.
- [23] P. Kayal and J. Liebeherr, "Autonomic Service Placement in Fog Computing," *IEEE 20th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1-9, Aug. 2019.
- [24] Y. Xia, X. Etchevers, L. Letondeur, T. Coupaye, and F. Desprez, "Combining hardware nodes and software components ordering-based heuristics for optimizing the placement of distributed IoT applications in the Fog," *SAC '18, Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, pp. 751-760, April 2018.
- [25] G. Baranwal, R. Yadav and D. P. Vidyarthi, "QoE aware IoT application placement in fog computing using modified – TOPSIS," *Mobile Networks and Applications*, vol. 25, pp. 1816-1832, Aug. 2020.
- [26] Z. A. Mann, "Decentralized application placement in fog computing," *IEEE Trans. Parallel and Distributed Systems*, vol. 33, no. 12, pp. 3262-3273, Feb. 2022.
- [27] Z. N. Smani, N. Saurabh, and R. Prodan, "Multilayer resource-aware partitioning for fog application placement," *IEEE 5th International Conference on Fog and Edge Computing (ICFEC)*, June 2021.
- [28] R. Mahmud, S. N. Srirama, K. Ramamohanarao, and R. Buyya, "Quality of Experience (QoE)-aware placement of applications in Fog computing environments," *Journal of Parallel and Distributed Computing*, vol. 132, pp. 190-203, Oct. 2019.
- [29] M. Aldossary, "Multi-Layer Fog-Cloud Architecture for Optimizing the Placement of IoT Applications in Smart Cities," *Computers, Materials & Continua*, pp. 633-649, Nov. 2022.
- [30] C. Canali, G. Di Modica, R. Lancellotti, S. Rossi, and D. Scotece, "A Validated Performance Model for Micro-services Placement in Fog Systems," *SN Computer Science*, vol. 4, May 2023.
- [31] S. Mirampalli, S. N. Srirama, R. Wankar, and R. R. Chillarige, "Hierarchical fuzzy-based Quality of Experience (QoE)-aware application placement in fog nodes," *Software: Practice and Experience*, vol. 53, pp. 263-282, Feb. 2023.
- [32] Z. N. Samani, N. Mehran, D. Kimovski, S. Benedict, N. Saurabh, and R. Prodan, "Incremental Multilayer Resource Partitioning for Application Placement in Dynamic Fog," *IEEE Trans. Parallel and Distributed Systems*, vol. 34, pp. 1877 – 1896, March 2023.
- [33] H. Sabireen and N. Venkataraman, "A Hybrid and Light Weight Metaheuristic Approach with Clustering for Multi-Objective Resource Scheduling and Application Placement in Fog Environment," *Expert Systems with Applications*, vol. 223, Aug. 2023.
- [34] X. Li H, T. Wang, J. Wang, P. Zheng, T. Liu, and L. Tang, "A cost-efficient and QoS-aware adaptive placement of applications in fog computing," *Concurrency and Computation*, vol. 35, Sept. 2023.
- [35] M. Zare, Y. E. Sola and H. Hasanpour, "Towards distributed and autonomous IoT service placement in fog computing using asynchronous advantage actor-critic algorithm," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, pp. 368-381, Jan. 2023.