INTERNATIONAL JOURNAL OF SOFTWARE ENGINEERING & COMPUTER SYSTEMS (IJSECS)
ISSN: 2289-8522 e-ISSN: 2180-0650
VOL. 8, ISSUE 1, 11 – 21
DOI: https://doi.org/10.15282/ijsecs.8.1.2022.2.0092

ORIGINAL ARTICLE

Hybrid load-balancing algorithm for fog computing environment

A. Abuhamdah, M. Al-Shabi

Department of MIS, College of Business Administration, Taibah University, 30002, Madinah Munawarrah, Saudi Arabia.

ABSTRACT – Fog computing has become a new trend in the Internet of things domain and cloud computing applications. It is a novel model to achieve the availability, flexibility and better responding time. In spite of that, there is so many challenges facing computing environments such as the misuse of the resources and load-balancing between them, which has a major effect on performance. The requirement of effective and robust load-balancing algorithms is one of the most significant interest in this field. Many researchers suggested various load-balancing algorithms in fog computing, but there is still inefficiency in the system performance and misalignment in load - balancing. This paper will provide a description of numerous concepts such as computing fog, fog nodes, load balancing and then we recommend a load-balancing algorithm to enhance the fog-computing environment performance, which is a hybrid algorithm benefits from the optimizing processing time (OPT) algorithms. In order to explore the proposed algorithm performance, a comparison made with other algorithms. Results indicates that using the proposed optimizing processing time algorithm in load-balancing algorithm has superior response and processing time than the compared algorithms to user requests, and the data total cost centre's as well.

ARTICLE HISTORY
Received: 2 Oct 2021
Revised: 29 Nov 2021
Accepted: 5 Feb 2022

KEYWORDS
Cloud Computing
Computing Architecture
Internet of things
Fog Computing
Load balancing

INTRODUCTION

Generally, there is a huge data used to the expansion of Internet of things which is influences storage capacity and cloud computing, also many applications are growing day by day relying on the Internet of things. Therefore, delaying the data in an application is not constantly suitable, where the data transfer through cloud to end-user [1]. Cloud computing is suggested wherever cloud services can operate rapidly to reduce data delays and guarantee user satisfaction. Fog computing stands for simulated platform that offers storage, computation services and networking amongst cloud platforms and end devices [2]. Table 1 demonstrates the parameters of fog and cloud computing system [3].

Table 1. Fog and cloud computing Parameters.

Parameters	Fog Computing	Cloud Computing
Node Location	In the local network edges	In the Internet
Delay	Low	High
Delay Frequency	Low	High
Security	Almost High	Low, Indefinite
Navigation	Yes	No
vulnerability	Low	High
Geographical Distribution	Dense	Focused
Node Number	Very High	Low
Mobility	Supports	Limited
Reliance on the Internet	Medium	High

Cloud computing, is reflect a central layer that communicates with users directly and provides confirmation, security, dependability, and latency. To realize the fog systems and the Internet of Things (IoT) in real-time analysis, a few problems must be overcome, such as fog service distribution, resource allocation, and load balancing. In the workload functioning of distribution systems on computing resources, load balancing is an effective approach. Load balancing outperforms traditional approaches in terms of cost and performance. In virtual machines, several high-work-load functions are avoided or added to low-work-load machines [2].

Load balancing techniques that are employed toward improve the system performance, preserve system constancy, and develop fault tolerance systems via baking at a low cost. A variety of load balancing techniques are described, each with its individual set of benefits and drawbacks based on the system's capacity to distribute workload.

*CORRESPONDING AUTHOR | A. Abuhamdah | aabuhamdah@Taibahu.edu.sa

Load balancing is a strategy for dividing a workload among numerous computing resources such as computers, hard drives, and networks. This fair distribution of client request resources is attempted to achieve within the best possible to ensure proper resource usage. It also aims to fix the problem of each processor in the system and each node in the network having to share the same amount of workload that is assigned to them. It can be accomplished with the use of appropriate hardware or software, such as a multilayer or domain name system process. Backup plan simply in case the system fails a tiny amount, assuring system stability, throughput, reaction time, minimal latency, minimum network delay, execution time, low overhead, low delay, and scalability are the important characteristics that create effective load balancing [1-3]. A load balancing technique may evenly distribute load across all virtual machines. Within the fog layer, it has become necessary to distribute all workload fairly and equitably across the existing VMs within the segment. Many load balancing approaches for fog computing have been presented so far.

We shall examine and explain various fundamental topics in this work, including computing fog, fog nodes, load balancing to propose a load-balancing algorithm that relay on optimizing processing time (OPT) algorithm in order to improve the performance with efficient manner in a fog-computing environment by using three parameters: the processor speed, the allotted load of the Virtual Machine (VM), and the network bandwidth.

The paper is prepared as follows: Section 2 discuss related studies on distinction among fog computing, cloud computing, Fog Computing Architecture and Load Balancing. Section 3 present the implementation design. Section 4 presents the experiments with results. Section 5 present the conclusion of this work.

RELATED WORK

In this section, we will present the Fog Computing, fog node concepts, Fog Computing Architecture, Load Balancing and related studies.

Fog Computing

Traditional computer paradigms, such as cloud computing, are incapable of processing enormous amounts of data [4]. There is a vast amount of information using 50 billion devices linked to the Internet (according to Cisco estimations). With the recent expansion of IoT-related services such as eHealth, smart cities, industrial scenarios, and smart transportation systems, privacy gaps, significant communication delays, and linked network traffic loads that connect cloud computing to end-users for unforeseen aims have emerged [5].

These are some of the issues that have an impact on cloud computing performance. To discuss some of the limits of cloud computing and to carry cloud service characteristics nearer to "Things," such as vehicles, mobile phones, embedded systems, and sensors.

The scientific community has proposed the fog computing notion [6], which is considered as a platform that brings cloud computing closer to end-users. Cisco [6] coined the term "fog," which has a similar meaning to real-life fog. Clouds rise higher in the sky when fog is closer to the ground, and fog computing leverages this perception once the virtual fog platform is placed nearer to end users, merely among their devices and the cloud. Another description of fog computing is "computing at the network edge," which enables the transmission of new services and applications designed expressly for the Internet's future [7].

Bonomi et al. [6] provide a better description for fog computing, assuming that fog computing is not restricted to the network edge. It was, however, a platform that provided networking virtually, storage, and computations between traditional cloud computing data centers and end devices.

Fog and Edge computing are sometimes conflated. Fog computing is hierarchical, with applications running in a multi-layer architecture that connects/interconnects software and hardware processes, allowing for active reconfiguration of a wide range of applications while sending services and performing intelligent computing. However, edge computing is restricted to a little peripheral devices and creates a direct transmission service as well as managing unique applications in a set logic location. In addition to networking and compute, fog computing may deal with data-processing control, storage, and acceleration [3], [8].

To distinguish fog computing from other computing standards, when using a fog computing service, an IoT client or smart end-device must use some of the following capabilities [4], [9]:

- Low latency and contextual location awareness: Fog computing achieves the lowest feasible latency by being aware of fog node latency costs for interacting with other nodes as well as their logical location in the overall system context. Fog computing, which enables and links rich services at the network edge, requires low latency in applications. Fog nodes can evaluate and respond to information faster than cloud service or centralized data center as they are often co-located with smart user devices [10].
- Geographical distribution: Fog computing provides services to geographically identifiable, distributed deployments, as opposed to a more centralized cloud. Fog computing will be important in providing high-quality spilling services to moving vehicles thru strategically located access points and proxies beside railways and highways.
- Heterogeneity: Fog computing facilitates the processing and gathering of a wide range of data from a variety of network connections.

- Federation and interoperability: Certain services demand the collaboration of numerous providers, therefore services must be unified across domains and fog computing components must be compatible.
- Real-time interactions: As an alternative of batch processing, real-time interactions are used in fog computing applications.
- Fog-node clusters, federated agility, and scalability: At the cluster or cluster-of-clusters level, fog computing is naturally adaptive, with network condition variations, data-load changes, resource pooling, and elastic compute support to name a little adaptive functions provided.
- Wireless access dominates: In wireless IoT access networks, the fog system is prominent. Although fog computing is commonly consumed in wired systems, where huge number of IoT wireless sensors necessitate distributed computation and analytics.
- Mobility support: Supporting mobility methods such as the locator/ID separation protocol, which decouples identifying the host from location, necessitates the use of a distributed directory system, which is required for a variety of fog computing applications that require straight mobile devices communication.

Revisiting fog node concepts

In recent researches that fall into the first group, there is a diversity of viewpoints including the suggestion that the manufacturer direct it [11], [12], Smart Gateway [13], eHealth services [14], and fog nodes helping as storage spaces in data networks based on data [15]. The second category involves with fog computing [16], which proposes a system comprising of three layers as well as cloud data centers, fog nodes spread around the edges of the network, and hardware devices as endpoints [17], [18]. The relevant within the second group also invests in the reality that categorizing apps into several levels, and raw data obtained by sensors is assembled and processed via edge devices. Figures 1 and 2 illustrate instances of the two categories to help with comprehension.

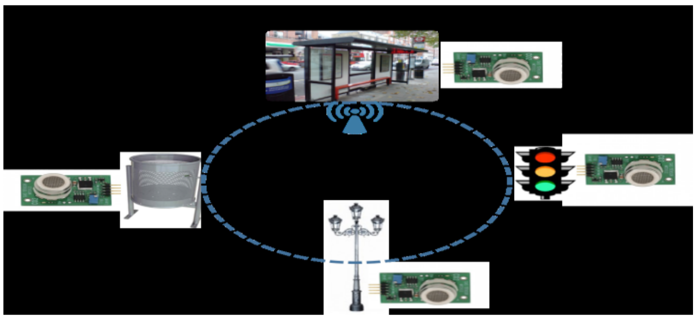


Figure 1. Fog node processing sensor's data.

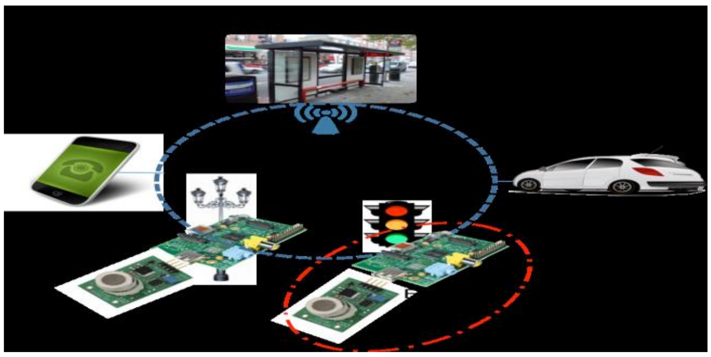


Figure 2. Fog node aggregating IT capacities of edge devices.

Figures 1 and 2 depict a smart city platform with a number of bus stops fitted out with fog nodes, which are created as mini-clouds and feature at minimum one server with processing competences. By assuming that a smart city stand has a large number of bus stations that are fed by fog nodes, which are small clouds with at least one processing server. The bus station is likely to be one of the initial group of fog nodes in Figure 1; in a fog environment, edge devices are unable to send data. When we say "maker/consumer", we mean that each bus station is capable to management and supervision a particular area through numerous motors and sensors). Each sensor, for example, may measure or gather a certain type of data (temperature and the number of cars and trucks found out), and sensors relay data to a fog node for processing, which used to be on trucks or bus stations.

Let's say the main purpose of this processing is to alert municipal management about a large number of trucks, autos, and vehicles in a specific area of the country.

Figure 2 shows the next group, which comprises a sensor connected to a mobile phone, a vehicle, and compute board, all of which include processing capabilities in addition to sensing capabilities (where in these devices and vehicles we assume that they have a temperature sensor). The data would be generated, processed, and sent to the fog node's little cloud at this point. Fog node gathers and interprets sensor data as well as edge device IT capabilities. The most significant component that the two fog node groupings are constructed on, we would argue, is the sort of edge device capabilities.

According to certain studies, site fog nodes should be placed in high-capability equipment like routers or smart gateways. The gateway is likely to act as a middleman among sensors that associated to the Internet and human body, collecting data from the sensors, performing several protocol transformations, and providing data gathering, purification, and dimensionality reduction to the fog node [13], [19], [20].

Researchers propose that the previous gateway transform be improved so that it can process data from an ECG (Electrocardiogram) device [14]. Researchers suggest and propose a novel method known as a smart getaway, which is capable of processing IoT data such as sounds and video and sending it to the cloud over the internet [13]. This method is suggested for locating a new center capable of boosting source estimation and oversight careers. Fog computing is founded by the idea of creating a unique link between cloud resources and edge devices that isn't reliant on a specific device [6], [16], [21], [22].

In the Smart city, it is suggested that the three fog computing layers be used to analyze and process the huge amount of data [17]. Each fog layer has its own name, therefore the first layer for intermediate computing nodes, the second layer for edge computing nodes, and the third layer for cloud computing nodes (or lowest fog layer). Each fog layer is made up of diverse pieces of equipment, such as computers with intermediate processing capability, small computing nodes, and sensors with only sensing capabilities [17].

Aside from offering computation, storage and application services, the fog layer refers to edge devices like cameras and sensors, and the fog layer is operated at edge devices like network routers. The types of fog nodes that we previously discussed, the type that is "dumb" by its edge devices and be capable to only produce data, and the form that is "smart" by its edge devices be able to generate data and reprocess it, while we do not anticipate to reach the required expectation in the future.

To construct services, we'll need to tap into a distinct set of resources that don't exist in the prior groups to provide more processing and storage capacity. We can use a number of strategies, including a resource management plan and anticipated services. For example, we can acquire computing skills in environmental problems or disasters like earthquakes by enlisting the help of volunteers, such as vehicles stationed near the disaster site or people who can deal emergency personnel in their smart phone resources in case they are near and their phone is linked to the grid [23], [24]. As a result, increasing the processing capacity of edge devices will not suffice to handle extremely demanding scenarios, and we will need to design new approaches. These concepts add to the difficulty of the overall system, which necessitates mutual as well as standardized abstractions of diverse edge devices.

Fog Computing Architecture

In this section, we'll go over why the Fog Computing paradigm is vital, as well as its key functions in providing IoT services in real time. We also highlighted the premium and valuable architecture, as well as its essential components and qualities.

Fog Computing [25] is a computing platform with wireless distribution that allows a collection of shared resources at the IoT gateway level in a specific place to accomplish sophisticated latency-sensitive activities. According to another description [26], fog computing is architecture of horizontal system that extends storage, control, compute, and networking capabilities closer to users beside a cloud-to-device continuum. Whereas under the first definition, processing and storage capabilities are distributed across a large number of IoT devices that are placed alongside layer devices. The fundamental impetus for the creation of (fc) was the crucial necessity to develop and decrease the processing and analysis time of data acquired in the cloud platform, as stated in the second description. As a result, the answer is more appropriate, and the decision-making process is aided. The parts of the fog computing model (i.e. storage, computation, and networking) are fog nodes, and the fog computing model exist in the device layer and the cloud. Figure 3 demonstrates the Fog Computing paradigm's high-level architecture, demonstrating in what way a collection of heterogeneous IoT devices be able to use fog computing to connect through a cloud platform.

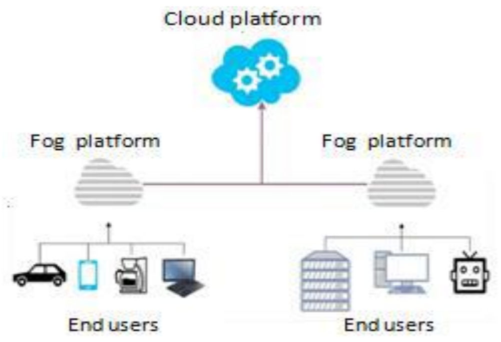


Figure 3. A high level architecture of Fog Computing model [27].

Load Balancing

Load balancing in a fog system allows for an even allocation of burden across resources, with the goal of continuing to deliver services even if a service component fails. It is performed by the provisioning and de-provisioning of application instances, as well as the efficient utilization of resources. Fog computing requires an effective load balancing solution to boost application performance and network efficiency since data centers procure variation across hosts and disclose unique traffic characteristics [28].

Load balancing as a mechanism spreads the workload across multiple resources to avoid any overflow or under-load on resources. Load balancing, which distributes the load over many resources, can be done in both physical and software environments [29]. Throughput maximization, reaction time minimization, and traffic optimization are some of the objectives of load balancing. Other objective of load balancing include resource consumption optimization on the server side, request processing time reduction, and distributed environment scalability improvement [30].

Load balancing in fog networks can take many forms, including static, dynamic, or a combination of the two. Because user behavior is unpredictable and static load balancing methods are not always effective in the network, where the rule should be coded in the load balancer using static methods that include primary system information.

Furthermore, since of the dynamic load distribution depending on the pattern defined in the load balancer, dynamic approaches outperform static methods [31]. Dynamic load-balancing mechanisms use specific policies to apply the present system state to the end, such as [32]:

- Transfer: This specifies the conditions in which a job must be relocated from one node to another. The transfer policy's arriving tasks are transferred or processed according to a determining rule depending on each node's workload. Task rescheduling and migration are addressed by the policy.
- Selection: This determines whether a task would be transmitted or not, as well as a few other factors such as the amount of overhead required for migration, task execution time, and total non-local system calls.
- Location: It identifies under-utilized nodes and assigns jobs to them. The accessibility of required services for job migration or rescheduling is verified on aimed nodes.
- Data: This policy collects all available data, including system nodes, which is then used by other policies to make decisions. This policy specifies when the data should be gathered.

The following are some of the links between various policies: Incoming tasks are grabbed by the transfer policy that determines whether or not they should be sent to a remote node to balance load. The job will be processed locally if it is not eligible to be moved. When a task's transfer policy indicates that it must be moved, the location policy is triggered to discover a remote node to complete a job. When a far partner cannot be discovered or the work cannot be transmitted to a remote node, the task is processed locally. The essential information for location and transfers is provided by the information policy to assist them in making a decision.

In Load Balancing, several criteria (or metrics) are essential to assess a load balancing mechanism and associate it to other mechanisms in order to determine which is superior and to recognize the benefits and drawbacks. Some qualitative paradigms are required for the metrics. Various qualitative measures, such as reaction time, cost, and energy, are utilized in articles. The following are the key metrics for load balancing in fog computing:

- Response time: is the time for a request (or task) needed to be accepted and for a server request to be responded to in a fog environment.
- Cost: The exchange of money for the performance of a desired action.
- Energy consumption: This refers to the amount of energy consumed by a fog network.
- An excellent load balancing technique can reduce energy consumption.
- Scalability: It demonstrates in what way the system be able to implement a load balancing technique through only a few hosts or machines.
- Security: This aspect of service ensures non-repudiation and confidentiality by requiring parties to authenticate each other and encrypting messages.
- Flexibility: Fog nodes that are always connected to a system pro tempore but leave on a regular basis. As a result, this technique must be flexible in order to reflect both newly joined nodes and node revocation.
- Resource utilization: This is the percentage of a cloud system's resources that are used to their full potential.
- Deadline: The most recent time a fog system service request can be performed.
- Processing time: The amount of time that the fog system need to reply (or response) to a service request.
- Reliability: refers to a fog network's capacity to fulfill its requests within a certain amount of time and under certain conditions.
- Throughput: The supreme requested service rate that the fog system able to process is referred as throughput.
- Availability: An increase in service requests or resource application that able to keep a computing system running smoothly, displaying its capabilities.

Load Balancing In Fog Computing

Load balancing is consider as a challenge that involves distributing load among operators to optimize system performance. The load balancing technique on data centers is depicted in Figure 4.

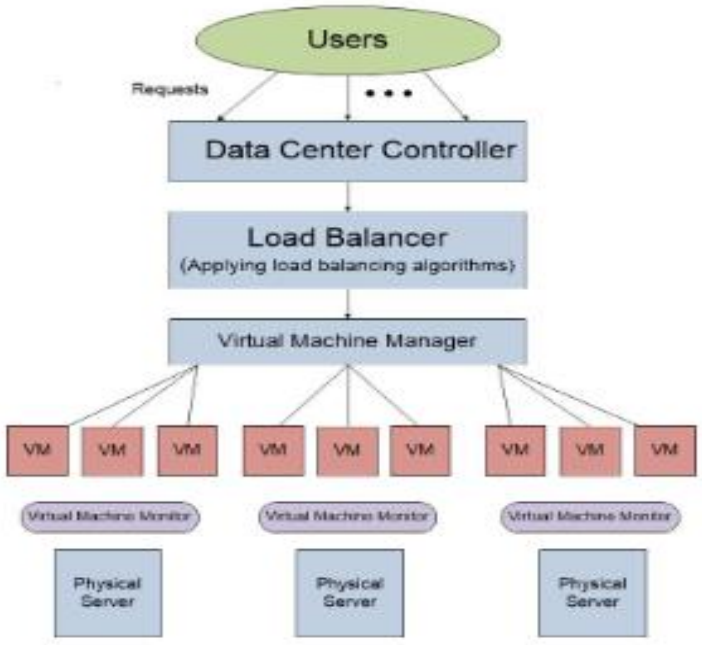


Figure 4. Data Centers Load-Balancing Mechanism.

In Figure 4, end users send demands to data centers, which the data center supervise routes to the right server, and the load balancing process efficiently distributes the demands among established Virtual Machines [2], [33], [34].

Related Studies

In fog computing, various load balancing approaches have been suggested, with only a handful achieving good results.

The authors present an OLBA framework for Load Balancing in Fog Computing Environments in [35], which balances the load between fog devices while optimizing QoS variables including turnaround resource utilization, latency, and delay parameter. That method finds the local best using the Particle Swarm Optimization (PSO) technology, then matches all of them to discover the final global best solution. A study with recognized approaches such as FCFS, Max Min, and SJF approaches is also conducted by comparing the results to gain a better understanding of the load balancing mechanism in Fog Computing.

The authors of [36] present a dependable scheduling method for assigning client requests to Cloud-Fog environments' resources. By classifying needs as real-time, important, or time tolerant, the Load Balanced Service Scheduling Method or Approach (LBSSA) handles load balancing across resources while allocating requests. In their load balancing operations, they used a variety of methods, including the quantity of computing resources, resource use, load balance adjustment, and running time.

To solve the resource management problem, the authors of [37] suggested a modification to deal with heterogeneity on two distributed load balancing algorithms. Evaluating the performance done by a basic setting, where undertake various sensitivity tests in relation to the aspects that generate infrastructure heterogeneity, as well as in a real-world smart city scenario.

The authors of [38] suggested a modern approach for load stabilizing in a fog computing environment. They and study the load balancing process and use a load balancing algorithm. In their load balancing assignments, they employed Active Virtual Machine Load Balancing Algorithms, Round Robin, Throttled, and Particle Swarm Optimization (PSO).

In [10], authors introduces a novel load balancing algorithm Abbreviation of the first task (MSJF) for regulating the load across many Virtual Machines (VMs) on Fog servers. A lot of factors influence the algorithm's level (e.g. processing time and Response time).

The authors in [39] explain how to employ an ant colony method for cloud load balancing that is time-efficient and threshold-approved. A hybrid of two scheduling techniques. The survey has created an Ant Colony Mixture Increases Efficiency System with a threshold applied Load Balancer algorithm. A virtual machine was chosen using the suggested Threshold-based ACO technique. They looked into whether or not work allocation to the specified virtual machine is smaller than the criterion in this paper.

The authors suggest a metaheuristic using virtual machine job migration method for cloud load balancing utilizing a high-efficiency particle swarm approach in [40]. The suggested method detects excessive burden on some virtual machines (VM) in a cloud computing environment and distributes the task to other machines. The purpose of cost functions is to simulate the actual cost of task transfer. To identify overloaded VM, the VM task migration technique employs an ineffective discriminate function. A novel discriminating function has been proposed in this research to identify actual overloaded VM. To find effective job migration techniques, they propose a particle swarm optimization (PSO) method.

A multilayer structure was employed to allocate resources in [41]. In this model, load balancing techniques like genetic algorithms, round robin, and throttle were employed to create packing techniques and allocate resources.

In [42], the authors suggested original load balancing strategy for verifying edge data centers and discovering lightly laden data centers for task distribution. The strategy boosted load balancing efficiency while also ensuring security by validating target data centers, according to the findings.

IMPLEMENTATION (THE PROPOSED ALGORITHM OPTIMIZING PROCESSING TIME (OPT))

The load balancing method works by first calculating the total demand on the CPU, memory, and network. Second, by examining how the nodes interact with one another, assessing load, and comparing nature, we can determine their fitness as a system. Finally, the load balancer should not be a single point of failure. It has become a need as the internet's popularity grows owing to increased usage of social networking websites, massive databases, and E-Commerce, which is causing many businesses to rely on it on a regular basis, requiring high bandwidth. It is referred to as a server farm since it serves a single Internet service from several servers. In a fog environment, our suggested method would address challenges such as latency, bandwidth, deadlines, and resource availability.

This is a new algorithm that will be used prior to the work being sent to the servers. It is planned by using three parameters: the processor speed, the allotted load of the Virtual Machine (VM), and the network bandwidth. Where it saves the current number of requests assigned to each VM as well as information in each VM of the system. When a request comes in for scheduling, the suggested method identifies the machine with the least load, and if there are multiple machines with the least load, it identifies the first one using the proposed load balancing technique with Fuzzy logic. This new proposal method called Optimizing Processing Time (OPT). In this paper, a fog-based load balancing method that achieves the best latency, resource usage, and processing time in the Virtual Machine (VM) environment is proposed.

We adopt fuzzy logic in this study for a variety of reasons, including its ease of use and naturalness. Its benefits include the ability to deal with imprecise data, ease of understanding, flexibility, and the ability to represent nonlinear functions [43], [44]. Fuzzy inference is a technique for producing a mapping between input and output using fuzzy logic. It provides a foundation for making choices or recognizing patterns through the mapping route.

The fuzzifier in this work turns each VM's inputs into a single output. To demonstrate the process workflow, we utilize fuzzy logic with three inputs: processor speed, allocated load in VM, and bandwidth. The goal is to achieve the highest achievable value in order to balance the system's load. The three parameters are used as input to the fuzzifier, and the balanced load is a measure for these parameters to get the result. This algorithm has determined the most effective VM for processing client requests. When a job arrives at the job manager, it is scheduled to the best machine for the job based on the data categorization. If the data is large, it necessity to be controlled in cloud computing and delivered to the data center; else will be guided to the fog. If it is busy, or in the middle, according to the stability and statuary of the fog. The following algorithm 1 (Optimal Processing Time (OPT)) will be used to determine it.

Algorithm 1: ***Optimal Processing Time (OPT)***

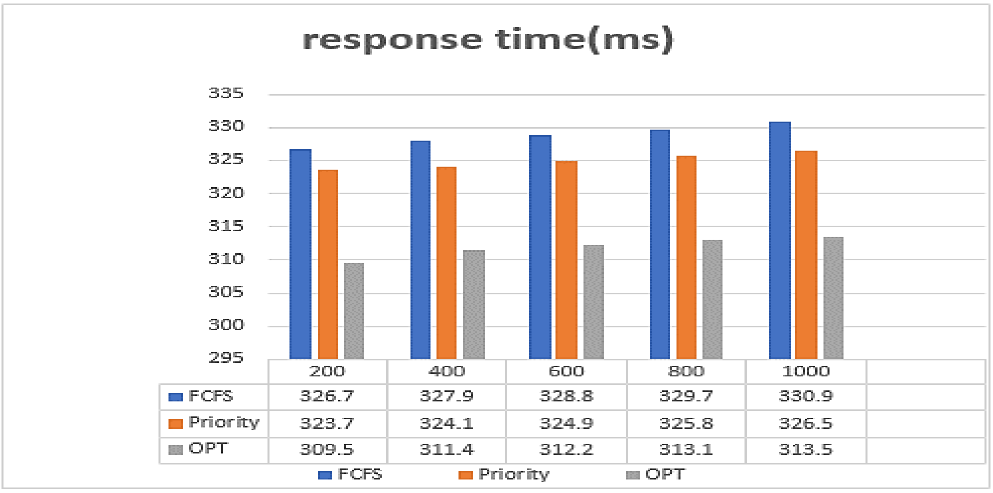
```
    Input:
    Fog Node (F0n);
    Fog Capacity (F0c);
    Current Load (C01);
    Over Load (H01).
    VARIABLE: Offload Value (F0v);
    Load To Share (L0s);
    Tasks Queue(q0).
    Output:
    List Of Nodes=(F0n,L0s);
    Having (L0s≤H01).
    Initialization:
    F0n=μ ; F0c=μ ; C01=μ ;H01=μ ;F0v=μ ; L0s=μ .
    Result:
Distribution the Overload requests on fogs
```

```
Procedure 1:
1. select  the overload by
2 if F0n = μ then {
3    C01=F0n q0 ∪T0n: T0n new requests
4    F0c =get Capacity (F0n)
5    H01=C01-F0c }
6 else
7    F0n← getNode(out F0n=C01>=F0c)
8    go to 3
9 end
10 if H01> 0 then {
11    return F0n← H01
}
12 else
13    return F0n
14 end
15 end
Procedure 2.
16 select Best Neighboring by
17 F01=list {μ } F01 start of fog list
18 F01 get Fog Nodes ()
19 F01 =sort (F01,by F0c DESC)
20 for each F0n ∈ F01 do
21 if F0n (C01 >=F0cmax) then
22     pop (F01,F0n) delete busy nodes
23 else
24     go to 20
25 end
26 end
27 cost function (F01, DESC) get best nodes
28 F01← F01{F0n,L0s}
29 return F
30End.
```

EXPERIMENTAL RESULTS

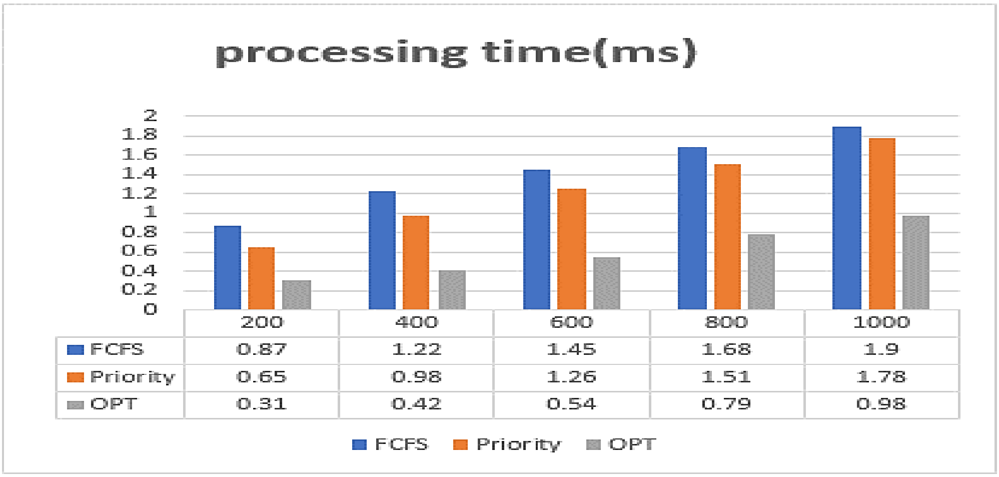
In this section, we will verify the proposed Optimizing Processing Time (OPT) theoretical outcomes by an evaluation prepared between the results for the proposed algorithm and other algorithms results for response time and resource utilization. Also, analyzing the performance of three VM load balancing algorithm by our experimental results .The results obtained by using matlab R2014a and connecting this with iFogSim to find out the best resource to the request. To apply two loads, we adapted the average number of users and the time of Peak Hours and in each data Base (User) base. The number and the speed of the central processing units are alike for all of the actual Hosts.

In order to assess our algorithm's progress and effectiveness, a comparison made between the proposed algorithm and the First Come First Serve (FCFS) algorithm and Priority algorithm. The average response time (which is the time among sending a request and getting a response) and the average processing time (which is the time it takes to complete a task) are the two metrics used. To increase overall performance, both the average response time and the average processing time should be reduced [45]. Figure 5 depicts the response time contrast, whereas Figure 6 depicts the processing time comparison.



	200	400	600	800	1000
FCFS	326.7	327.9	328.8	329.7	330.9
Priority	323.7	324.1	324.9	325.8	326.5
OPT	309.5	311.4	312.2	313.1	313.5

Figure 5. The response time in the algorithms.



	200	400	600	800	1000
■ FCFS	0.87	1.22	1.45	1.68	1.9
■ Priority	0.65	0.98	1.26	1.51	1.78
■ OPT	0.31	0.42	0.54	0.79	0.98

Figure 6. The processing time in algorithms.

Figure 5 shows , that the results execution clarify that the Optimizing Processing Time (OPT) algorithm has the best response time than the First Come First Serve (FCFS) and Priority algorithms, meanwhile, FCFS and Priority algorithms have similar outcomes. We can notice that the response time is very fast and does not exceed 310 ms when the number of tasks was 200, while the value is more than 320 ms for the Priority algorithm and more than 325 for the FCFS algorithm. With the increase in the number of tasks at 400, 800 and to 1000, the response speed of the proposed algorithm remains high and faster than the rest of the algorithms.

Figure 6 shows, that the results execution clarify that OPT algorithm has the finest processing time while comparing with FCFS and Priority algorithms. We can notice that the processing time was 0.30 ms when the number of tasks was 200, which is faster than the Priority algorithm with a value of 0.65 ms, while the FCFS algorithm reached 0.85 ms. When the number of tasks increases (400, 600, 800 and 1000), the processing time remains fast according to the proposed algorithm, and in less time than the rest of the algorithms.

Outcomes in figures 5 and 6 concluded that in the optimizing processing time (OPT) algorithm has superior response and processing time than the First Come First Serve (FCFS) and Priority algorithms. For more understanding, Figure 7 and figure 8 shows the response and processing time improvement percentage, where the blue color indicate the improvement percentage for OPT against FSFS, and the orange color indicate the improvement percentage for OPT against Priority. For example, in figure 7, when the number of tasks is 200, OPT is improved by 64.37% while comparing with FSFC, and OPT is improved by 52.31% while comparing with Priority.

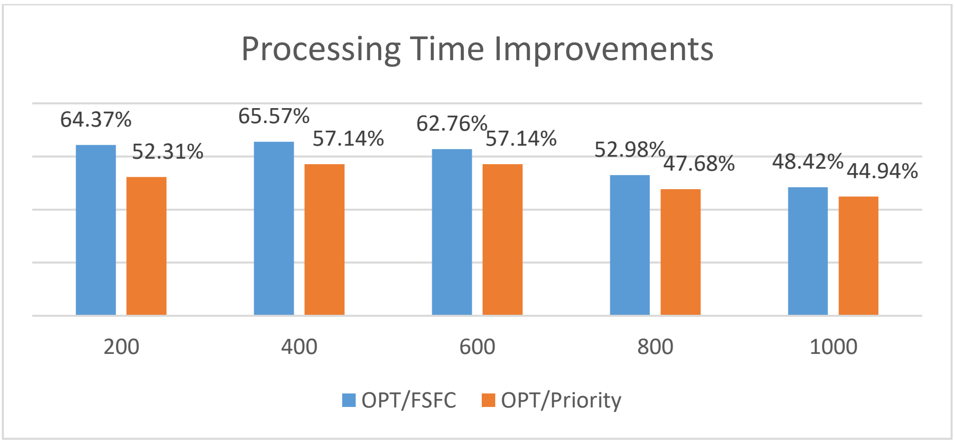


Figure 7. Response time improvement.

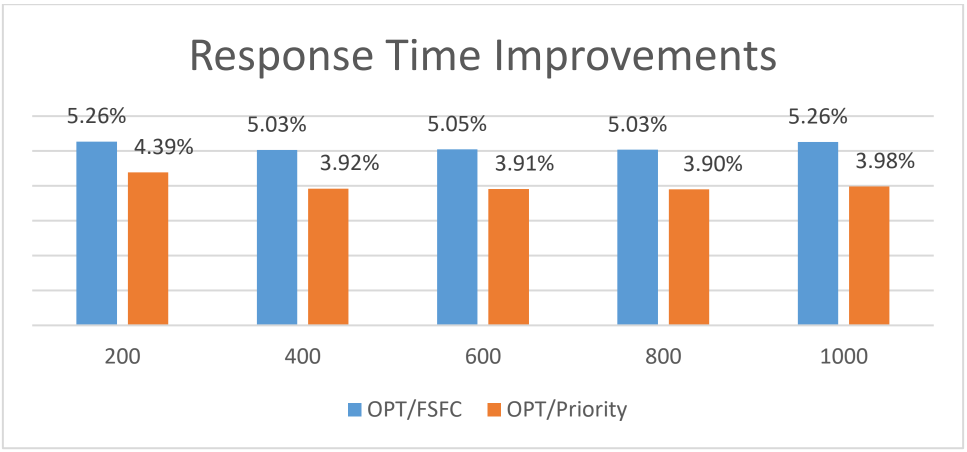


Figure 8. Processing time improvement.

CONCLUSION

IoT has emerged as one of the most essential and necessary technologies in recent years. In a world where the Internet of Things is rapidly evolving, fog computing has arisen as one of the finest models for IoT applications. In the fog, IoT applications are executed by edge and middle computing nodes, along with physical devices in the cloud system. Fog computing has advanced quickly as a recent technology that has been tested on various occasions. One of the most difficult facets of fog computing is load balancing. So far, several load balancing approaches have been proposed, each of which is suitable for unique settings. The fog and cloud-based platform is employed to offer facilities and solve their difficulties, which could be similar to the needs of the old grid system.

In this study, a load balancing technique was proposed to decrease the processing and response time of fogs to clients. The response time and processing time have been reduced as a result of simulation results.

In this paper, we propose and develop an Optimizing Processing Time (OPT) load balancing method in the fog computing environment using the iFogSim tool. When compared to current algorithms like FCFS and Priority scheduling algorithms in the fog computing environment, the results achieved after implementing our suggested architecture and algorithm are really good; it has delivered minimal execution time and quick response to client requests.

In future, we shall endeavor to examine the negative consequences of service relocation in the future and compare them to the good aspects. The cost of data migration, the cost of purging services, the lowered quality of implementation for purging services, and the traffic for various types of computing nodes are all negative impacts.

REFERENCES

[1] C. Li, H. Zhuang, Q. Wang, and X. Zhou, "SSLB: self-similarity-based load balancing for large-scale fog computing," *Arab. J. Sci. Eng*., vol. 43, no. 12, pp. 7487–7498, 2018.

[2] L. Shooshtarian, D. Lan, and A. Taherkordi, "A clustering-based approach to efficient resource allocation in fog computing," in I*nternational Symposium on Pervasive Systems*, Algorithms and Networks, pp. 207–224, 2019.

[3] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Commun.* Surv. tutorials, vol. 20, no. 1, pp. 416–464, 2017.

[4] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: architecture, key technologies, applications and open issues," *J. Netw. Comput. Appl*., vol. 98, pp. 27–42, 2017.

[5] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Internet of Things applications: A systematic review," *Comput. Networks*, vol. 148, pp. 241–261, 2019.

[6] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pp. 13–16, 2012.

[7] E. Marín-Tordera, X. Masip-Bruin, J. García-Almiñana, A. Jukan, G.-J. Ren, and J. Zhu, "Do we all really know what a fog node is? Current trends towards an open definition," *Comput. Commun*., vol. 109, pp. 117–130, 2017.

[8] O. C. A. W. Group, "OpenFog reference architecture for fog computing," *OPFRA001*, vol. 20817, p. 162, 2017.

[9] M. Iorga, L. Feldman, R. Barton, M. J. Martin, N. S. Goren, and C. Mahmoudi, "Fog computing conceptual model," *Special Publication (NIST SP), National Institute of Standards and Technology*, Gaithersburg, MD, 2018, https://doi.org/10.6028/NIST.SP.500-325 (Accessed September 1, 2021).

[10] T. Nazar, N. Javaid, M. Waheed, A. Fatima, H. Bano, and N. Ahmed, "Modified shortest job first for load balancing in cloud-fog computing," in *International Conference on Broadband and Wireless Computing*, Communication and Applications, pp. 63–76, 2018.

[11] C. F. C. Solutions, "Unleash the power of the Internet of Things," *Cisco Syst*. Inc, 2015.

[12] M. Ketel, "Fog-cloud services for iot," in *Proceedings of the SouthEast Conference*, 2017, pp. 262–264.

[13] M. Aazam and E.-N. Huh, "Fog computing and smart gateway based communication for cloud of things," in *2014 International Conference on Future Internet of Things and Cloud*, pp. 464–470, 2014.

[14] T. N. Gia, M. Jiang, A.-M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Fog computing in healthcare internet of things: A case study on ecg feature extraction," in *2015 IEEE international conference on computer and information*

technology; ubiquitous computing and communications; dependable, autonomic and secure computing; pervasive intelligence and computing, pp. 356–363, 2015.

[15] I. Abdullahi, S. Arif, and S. Hassan, "Ubiquitous shift with information centric network caching using fog computing*," in Computational intelligence in information systems,* Springer, pp. 327–335, 2015.

[16] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," in *Big data and internet of things: A roadmap for smart environments*, Springer, pp. 169–186, 2014.

[17] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *Proceedings of the ASE BigData & SocialInformatics 2015*, pp. 1–6, 2015.

[18] K. Skala, D. Davidovic, E. Afgan, I. Sovic, and Z. Sojat, "Scalable distributed computing hierarchy: Cloud, fog and dew computing," *Open J. Cloud Comput*., vol. 2, no. 1, pp. 16–24, 2015.

[19] A.-M. Rahmani et al., "Smart e-health gateway: Bringing intelligence to internet-of-things based ubiquitous healthcare systems," in *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, pp. 826–834, 2015.

[20] M. Aazam and E.-N. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT," in 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, pp. 687–694, 2015.

[21] F. Bonomi, "The smart and Connected Vehicle and the Internet of Things," *in Invited Talk, Workshop on Synchronization in Telecommunication Systems*, 2013.

[22] G. Caiza, M. Saeteros, W. Oñate, and M. V.Garcia, "Fog computing at industrial level, architecture, latency, energy, and security: A review," *Heliyon*, vol. 6, no. 4, 2020, doi: 10.1016/j.heliyon.2020.e03706.

[23] X. Masip-Bruin, E. Marín-Tordera, G. Tashakor, A. Jukan, and G.-J. Ren, "Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems," *IEEE Wirel*. Commun., vol. 23, no. 5, pp. 120–128, 2016.

[24] F. Büsching, S. Schildt, and L. Wolf, "Droidcluster: Towards smartphone cluster computing--the streets are paved with potential computer clusters," in *2012 32nd International Conference on Distributed Computing Systems Workshops*, pp. 114–117, 2012.

[25] H. Sabireen, and V.Neelanarayanan, "A Review on Fog Computing: Architecture, Fog with IoT, Algorithms and Research Challenges, " *ICT Express*, vol. 7, no. 2, pp. 162-176, 2021, 10.1016/j.icte.2021.05.004.

[26] Y. Abuseta, "A fog computing based architecture for IoT services and applications development," *arXiv Prepr*. arXiv1911.02403, 2019.

[27] S. Khan, S. Parkinson, and Y. Qin, "Fog computing security: a review of current applications and security solutions," J. *Cloud Comput*., vol. 6, no. 1, p. 19, 2017.

[28] W. Lin and L. Zhang, "The load balancing research of SDN based on ant colony algorithm with job classification," in *2016 2nd Workshop on Advanced Research and Technology in Industry Applications (WARTIA-16)*, pp. 472–476, 2016.

[29] A. S. Milani and N. J. Navimipour, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends," *J. Netw. Comput. Appl*., vol. 71, pp. 86–98, 2016.

[30] M. Xu, W. Tian, and R. Buyya, "A survey on load balancing algorithms for virtual machines placement in cloud computing," *Concurr. Comput. Pract. Exp*., vol. 29, no. 12, p. e4123, 2017.

[31] A. A. Neghabi, N. J. Navimipour, M. Hosseinzadeh, and A. Rezaee, "Load balancing mechanisms in the software defined networks: a systematic and comprehensive review of the literature," *IEEE Access*, vol. 6, pp. 14159–14178, 2018.

[32] K. Patel, A. Mehta, and K. Solanki "A Survey of Various Load Balancing Algorithms in Cloud Computing," *International Journal Of Engineering Research & Technology (Ijert) Icradl* – 2021, vol. 09, no. 05, 2021.

[33] L. Ruan, S. Guo, X. Qiu, and R. Buyya "Fog Computing for Smart Grids: Challenges and Solutions," *arXiv.org*, arXiv:2006.00812, 2020.

[34] R. Bukhsh, N. Javaid, Z. Ali Khan, F. Ishmanov, M. K. Afzal, and Z. Wadud, "Towards fast response, reduced processing and balanced load in fog-based data-driven smart grid," *Energies*, vol. 11, no. 12, p. 3345, 2018.

[35] S. Harnal, G. Sharma, N. Seth, and R. D. Mishra, "Load Balancing in Fog Computing Using QoS," in *Energy Conservation Solutions for Fog-Edge Computing Paradigms*, Springer, pp. 147–172, 2022.

[36] F. Alqahtani, M. Amoon, and A. A. Nasr, "Reliable scheduling and load balancing for requests in cloud-fog computing," *Peer-to-Peer Netw. Appl.*, vol. 14, no. 4, pp. 1905–1916, 2021.

[37] R. Beraldi, C. Canali, R. Lancellotti, and G. P. Mattia, "Distributed load balancing for heterogeneous fog computing infrastructures in smart cities," *Pervasive Mob. Comput*., vol. 67, p. 101221, 2020.

[38] S. H. Abbasi, N. Javaid, M. H. Ashraf, M. Mehmood, M. Naeem, and M. Rehman, "Load stabilizing in fog computing environment using load balancing algorithm," in *International Conference on Broadband and Wireless Computing,* Communication and Applications, pp. 737–750, 2018.

[39] C. Banerjee, A. Roy, A. Roy, A. Saha, and A. K. De, "A Time Efficient Threshold Based Ant Colony System for Cloud Load Balancing," in *International Conference on Computational Intelligence,* Communications, and Business Analytics, pp. 206–219, 2018.

[40] G. Megharaj and M. G. Kabadi, "Metaheuristic-Based Virtual Machine Task Migration Technique for Load Balancing in the Cloud," in *Integrated Intelligent Computing, Communication and Security*, Springer, pp. 435–446, 2019.

[41] M. Zubair, N. Javaid, M. Ismail, M. Zakria, M. A. Zaheer, and F. Saeed, "Integration of cloud-fog based platform for load balancing using hybrid genetic algorithm using bin packing technique," in International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, pp. 279–292, 2018.

[42] D. Puthal, M. S. Obaidat, P. Nanda, M. Prasad, S. P. Mohanty, and A. Y. Zomaya, "Secure and sustainable load balancing of edge data centers in fog computing," *IEEE Commun*. Mag., vol. 56, no. 5, pp. 60–65, 2018.

[43] L. D. Brown, H. Hua, and C. Gao, "A widget framework for augmented interaction in SCAPE," in Proceedings of the *16th annual ACM symposium on User interface software and technology*, pp. 1–10, 2003.

[44] Y. T. Yu and M. F. Lau, "A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions," *Journal of Systems and Software*, vol. 79, no. 5, pp. 577–590, 2006, doi: 10.1016/J.JSS.2005.05.030.

[45] N. S. Raghava and D. Singh, "Comparative study on load balancing techniques in cloud computing," Open J. *Mob. Comput. cloud Comput*., vol. 1, no. 1, pp. 18–25, 2014.