Article

Towards an Effective Service Allocation in Fog Computing

Rayan A. Alsemmeari [1], Mohamed Yehia Dahab [2], Badraddin Alturki[1], Abdulaziz A. Alsulami[3], Raed Alsini[3]

[1] Department of Information Technology, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; ralsimmeari@kau.edu.sa (R.A.A.); baalturki@kau.edu.sa (B.A.)
[2] Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; mdahab@kau.edu.sa (M.Y.D.);
[3] Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; aaalsulami10@kau.edu.sa (A.A.A.); ralsinie@kau.edu.sa (R.A.)
* Correspondence: aaalsulami10@kau.edu.sa (A.A.A.)

Abstract: The Internet of Things (IoT) generates a large volume of data whenever devices are interconnected and exchange data across a network. As a result, there is a range of services with varying needs, for example, capacity requirements, data quality, and latency demands. These services operate on fog computing devices, which are limited in power and bandwidth compared to the cloud. The main challenge is deciding where to implement services on the fog, the cloud, or the hybrid. This paper proposes an efficient allocation technique that pushes processing closer to the network's fog side. It investigates which devices and services may best allocate while preserving resource usage in the IoT architecture. It also examines the importance of allocating services to devices and optimizing resource use in fog computing. In IoT settings, there is a wide range of services and devices; thus, it is critical to effectively assign the services to the devices. We propose Priority-based Service allocation (PSA) and Sort-based Service Allocation (SSA) techniques, which is used to enable an optimum order to employ devices to perform the various services. The experimental results indicate that our proposed technique minimize the data communication over the network by 82% by allocating most of the services locally in fog. We have maximized the number of distributed services to fog devices by 90% while minimizing the wastage of fog resources.

Keywords: IoT, IoMT, fog computing, service allocation, optimisation, Cloud Computing

1. Introduction

IoT devices generate a large amount of data as they are interconnected [1]. Most current proposals focus on centralized, or cloud architecture [2]. The goal of a centralized architecture is to process data in one place of decision. Consequently, a significant amount of data must be uploaded to the cloud. Heavy data transmission via the network is one of the challenges of this design introduced [3]. This suggests that an alternative design is necessary to address these shortcomings. Since the IoT architecture connects several devices with varied levels of computing, storage capacity, battery life, and Internet access, device constraint awareness is a crucial part of its design.

Also, a variety of services will be available, each with a different set of expectations, such as those for capability, quality of data, and latency. These services operate on fog computing devices, which are limited devices in terms of power when compared to the cloud [4], and they demand bandwidth. This implies that fog devices and services have a strong connection. The main challenge is to decide, while considering overall efficiency, whether services should be run using a fog layer, cloud layer, or a combination of fog and cloud in a certain IoT architecture.

Furthermore, resource management at the network's edge [5] is critical for evaluating the advantages of fog computing. Developing an effective fog infrastructure presents a number of issues. Local data storage is an instance when resolving these issues becomes essential. The execution of services within distributed architecture becomes more challenging

 doi:10.20944/preprints202305.1523.v1

as the size and complexity of the IoT system increase, necessitating a method to allocate services to the node(s), resulting in the discovery of the ideal allocation strategy.

Computing every incoming raw data on the cloud has a detrimental impact on various elements, including higher network congestion, latency, the time it takes to return actions to a user, energy usage, and privacy [6]. As the Internet of Things expands, so there is a need to address these challenges. IoT devices are restricted devices because of their limited computing power when compared to cloud devices. As a result, huge workloads cannot be processed on fog nodes. In addition, determining the amount of computing load that may be allocated to a fog device is challenging. Furthermore, distributing services among fog devices is difficult since the large number of services in IoT might demand a lot of computing power [7]. As a result, we must understand the nodes' capabilities and services' demands. Then we must optimize the process of service allocation to the nodes while keeping optimal resource use.

This paper's overarching goal is to provide an effective allocation technique for processing data with reduced bandwidth utilization, faster reaction time, optimized resource usage, and identifying an optimum technique to process data on a big scale. One of the goals is to evaluate and test the proposed technique using a simulation. We propose an efficient allocation strategy that pushes processing closer to the network's fog side. Moreover, we investigate which devices and services may be best allocated while preserving resource use in the IoT architecture. In addition, we offer a service allocation technique for allocating services to devices depending on their capabilities. Our main contributions:

- Service allocation technique is significant because providing services to devices in the IoT is a difficult process due to the many types of devices and their capabilities. As a result, we propose Priority-based Service allocation (PSA) and Sort-based Service Allocation (SSA) techniques, which utilize a list of every fog device connected to the network. This method makes it possible to use fog devices in the best possible sequence to conduct a wide range of services. As a starting point, we use packing problems as a baseline to help solving the allocation issues in the IoT environment.
- We examine the importance of allocating services to devices and optimizing resource use in fog computing to enhance service quality while meeting the optimal resource usage demands of IoMT. As there will be a large variety of services and devices in the IoT settings, it is vital to allocate the services to the devices and effectively optimize resource consumption.
- We evaluate the PSA and SSA techniques using a Synthetic dataset that mimics the IoT services and devices. We do a tradeoff analysis to illustrate the effectiveness of the service allocation approach. The results reveal that the data communication over the network decreased by 92% since most services are allocated in fog. Additionally, the latency is reduced by approximately 86%.

The remainder of this paper is organized as follows: Section 2 presents related works in the field of service allocation, Section 3 describes the research problem and provides a motivational scenario, Section 4 provides the methodology including the algorithm and the architecture, Section 5 presents the experimental setup and reveals the details of the experiments, Section 6 show the results that obtained from experiments and provide the description of the results, followed by Discussion and evaluation, finally, Section 8 presents our conclusions and then recommendations for future research.

2. Related works

Fog computing has become an increasingly popular topic of research in recent years, as it offers a number of benefits for various industries [8]. One of the main challenges in fog computing is data distribution and allocation. This literature review will explore the current state of research on service allocation in fog computing and highlight some key references in the field.

Analyzing data closer to the fog lead to reduced latency, and increased efficiency, as well as improved security and privacy [9]. Fog computing also enables the deployment of

 doi:10.20944/preprints202305.1523.v1

computing resources closer to the data source, reducing the need for data transmission over long distances. This can lead to improved performance and reduced energy consumption.

In terms of service allocation, research has focused on the use of optimization techniques to allocate resources in fog computing environments efficiently. Optimization techniques, such as game theory, packing, linear programming, and scheduling, can also be used to model and solve service allocation problems in fog computing environments.

In resource allocation in edge and fog computing, we reviewed the research publications that focus on fog systems. The underlying infrastructure is assumed in these studies to be cloud-fog [10–24]. These options fall under the system elements aspect and have a significant impact on the researchers' optimization goals. When considering cloud and fog computing, many academics believe that the load is originally stored on the cloud and that the edge system must select where to duplicate and how to divide the user load among them [11]. As a result, they offer a framework for pushing applications that require lots of resources to the fog and reducing average data communication in the edge network across access points by duplicating cloud services to some of the edge servers. Workload distribution over systems that are heterogeneous has to consider the availability of various resources [25]. When spreading the workload between fog and cloud, the objective is to reduce the energy consumption in order to meet service latency needs [26]. When a specific research project does not assume the use of a central cloud but instead addresses multi-fog situations, the issue may arise from the combined optimization of job distribution, virtual machine placement, and resource allocation [12]. The authors in [14] attempt to reduce the load of users by determining user association, joint service placement, and joint allocation.

However, most of the publications have the same Optimization objective(s), namely, service completion latency [12,15,16,20–23], numerous research endeavors have been undertaken to address the trade-off between energy consumption and delay in data transmission, for example, [25,26]. In addition to delivering quick service completion to users, researchers aim to cover many users with the edge fog [19,22]. Cost minimization includes several aspects, such as resource usage, quality of service, and its associated revenue. Authors in [27] calculated the total cost of deployment by considering the wireless communication cost and the function placement computation cost, authors in [13,24] for maximizing user allocation numbers in their cost they considered the usage of edge servers to have the quality of service. In addition, the data communication over the network is also considered one of the aspects of the cost.

Optimization techniques play a crucial role in the efficient management of resources in fog computing, and IoT environments [28]. One popular optimization technique used in these environments is bin packing. Bin packing is a combinatorial optimization problem [29,30]that involves packing a set of items into a fixed number of bins, with the goal of minimizing the number of bins used or the overall cost of the solution. In fog computing and IoT environments, bin packing can be used to optimize the placement of services and devices, taking into account factors such as network conditions, service requirements, and device characteristics to minimize the overall cost of the solution by reducing the number of fog nodes used.

There are many variations of the bin packing problem, including the multi-dimensional bin packing problem [31] and the multi-constraint bin packing problem [32]. These variations can be used to put extra constraints and requirements in fog computing and IoT environments. For example, the multi-dimensional bin packing problem can be used to take into account the different resource requirements of services and devices, and the multi-constraint bin packing problem can be used to take into account additional constraints such as security and privacy.

In the literature, there are several works that have proposed the use of bin packing for the optimization of services allocation and task scheduling. Authors in [33] attempt to enhance task scheduling by transforming it into a bin packing problem. Three modified versions of bin packing algorithms based on the minimization of makespan were presented for use in task scheduling (MBPTS). They have used Cloudsim simulator [34]

 doi:10.20944/preprints202305.1523.v1

open source simulator. When compared to scheduling algorithms such as First Come First serve (FCFS) and Particle Swarm Optimization (PSO), the results of the proposed MBPTS were adequate to optimize balancing results, reducing the waiting for time and resource utilization improvement. Authors in [13] presented the edge user allocation (EUA) problem as a bin packing problem and presented a unique, optimum solution based on the Lexicographic Goal Programming technique. They ran three sets of tests to compare the suggested strategy to two sample baseline approaches. The experimental findings reveal that their strategy performs better than the other two alternatives substantially. In [35], the authors presented a methodology for minimizing resource waste by resource consolidation, which is accomplished by allocating many requests to the same machine. Bin packing is offered to perform semi-online workload consolidation. The suggested approach is built on bins, with each job allocated a bin that is subsequently allocated to a machine. The suggested approach addresses the issue of request reduction in real-time resource assignment. The suggested technique obtains information during a brief time frame, allowing for more accurate decisions. Their findings reveal that, during periods of high demand, their optimal policy can result in saving up to 40% more resources than the other policies and is resistant to unpredictability in task lengths. Finally, they demonstrate that even slight increases in the permitted time window result in considerable improvements but that bigger time windows do not always improve resource use for real-world data sets. In summary, bin packing is a powerful optimization technique that can be used to efficiently manage resources in fog computing and IoT environments. By taking into account factors such as network conditions, device capabilities, and additional constraints, bin packing can be used to minimize the number of resources used and reduce the overall cost of the solution.

Most of the literature focused on allocation strategies in the cloud, and they did not give good attention to service allocation in fog and IoT environments as these environments have various capability devices ranging from constrained devices and high capability devices. However, still, they are not powerful as the devices in the cloud. This makes the processing of allocation services to devices more challenging. It requires a good allocation strategy while optimizing all the aspects like data communication, energy usage, resource wastage, and response time. Moreover, the proposals in the literature have focused on the aspects of the network conditions and device characteristics, but they did not give attention to the technical requirements of service and task when they are in the allocation process. This is important as the services and tasks have technical requirements similar to the device's capabilities. We have considered the device capabilities and the technical requirements of the services in our proposal to have a full understanding of the allocation process and to allocate the services to devices efficiently.

3. Research Problem and Motivational Scenario

As the number of IoT devices linked to the Internet has grown, so has the number of services, and businesses have begun to install additional services for various objectives. Most IoT devices have limited resources such as RAM, CPU, and storage, as well as a lack of battery capacity. Furthermore, each deployed service has a comparable constraint represented regarding similar resources. Additionally, it is crucial to take into account the data processing capacities of IoT devices while implementing and distributing services. As a result, before providing information about services to IoT devices, we must understand their limitations.

For example, Magnetic resonance imaging (MRI) and X-ray produce videos and image data in healthcare. The analysis of these data requires more processing capacity than blood test results or electrocardiogram (ECG) results, which are numerical analyses. Other kinds of data analysis, such as video image analysis, needs greater processing power owing to their volume and the methods they employ, suggesting that the capabilities of the fog layer have to be robust enough to conduct these services. As a result, because fog devices have limited processing capability when compared to cloud devices, it is not viable to

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

implement these services on them. Due to resource-constrained devices [36] in regard to hardware, service allocation is a crucial part of fog architecture. Moreover, some fog devices are not utilized due to their limited power capabilities to execute a service, implying that certain fog devices are needed for data processing but are disregarded due to power limitations. This can be made worse when billions of services are sent to billions of devices and executed by them. This signifies that there is a waste of network devices, which may cause the computation time to be delayed. The waste of devices occurs when devices are not used owing to restrictions, and they remain in the network unused, losing the available resources. The key problem is determining which service should be allocated to which device in a fog architecture while keeping overall effectiveness in mind. This is comparable to the optimization problems, which can be considered the most suitable for allocation problems.

In Figure 1, several fog, IoT, and healthcare types of equipment are present in a hospital's environment. There are many patients (P_1, ..., P_n) in a hospital, and they are checking up on their health status. Furthermore, there are many services (S_1, ..., S_n) that the hospital provides to patients. Some of the patients are there for teeth problems, immunization, lungs, kidney and internal medicine, diabetes, eye issues, brain disorders, pregnancy, and heart issues. In our scenario, we consider these health problems that patients face as a part of services (S). In other words, every user can have one or more than one service, and when a user has a service, the service (S_i) will have all the data of the patients and the data of the tests.

This means that when a patient registers with the hospital, and the physician requires the patient test, x-ray, MRI, or other diagnosis checks, also, the physician decides whether the patient is a high priority or low priority, then the system will take this process as one service while taking into consideration the priority. These services with priority levels will be allocated to the fog devices (FD_1, ..., FD_n) for processing, then the fog device starts firstly, processing the high-priority services, then lower-priority services will be processed, and finally, when the fog devices are not capable of processing the services (low priority/high priority), the cloud devices (CD_1,..., CD_n) will get the services to apply further analysis. For example, if a patient is listed as a high priority, then the system will send the patient's service to the fog device for processing and getting a fast response. However, if the patient is in low priority and the fog devices are not capable, then the service will be allocated to the cloud. In only one situation, when the fog devices are not capable of processing high-priority services, then they will be allocated to the cloud for processing.

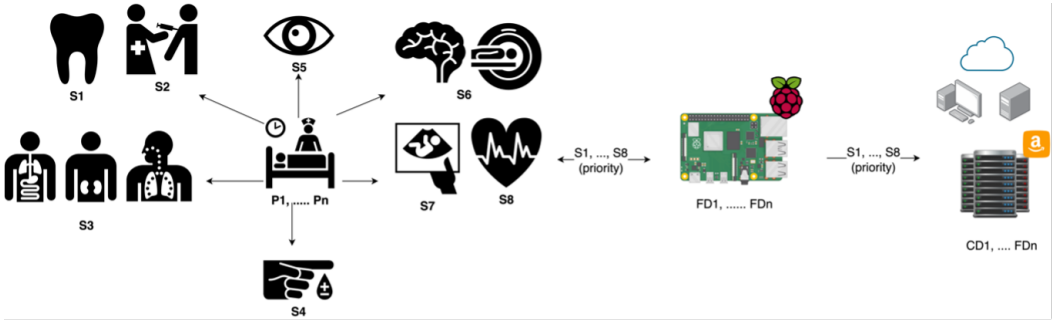


Figure 1. Scenario: A patient in a hospital

Assumptions

- We assume that the fog layer's devices have limited RAM capabilities compared to cloud devices. For the experiments, we produced Synthetic data.
- There are 800 services with varying technical requirements. Similarly, we gathered data for fog devices with various capabilities.
- The services' technical requirements, the fog devices' capabilities, and the priorities of services are randomly generated.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

- The technical needs of services and the device capabilities are known.
- We do not examine the connection between devices in our experiments since it is outside the scope of our study.

Process

- The model starts by building synthetic data for both the requirement of services and the capability of devices to prepare them for the allocation model.
- The fog devices' capabilities are predefined, with fog devices being less capable when compared to cloud devices.
- The allocation technique is used to allocate between service needs and service priority, and device capabilities.
- Depending on the needs of the services, taking into consideration device capabilities and service priority, the services will be allocated across fog or cloud devices.

4. Methodology

We propose PSA and SSA technique, which is technique of allocating resources based on a list that has all fog devices that are connected to the network. With this method, we may determine the sequence in which different services are carried out by various devices. The actual capacities of devices, such as RAM, are used to arrange the list of devices. As a result, for each physical aspect, all devices capabilities are maintained in a list. The main purpose is to allocate services to devices in an effective and optimized manner. The allocation technique is utilized to allocate all or a part of the services to a specific number of fog devices or to various cloud devices with varied capabilities according to their capability. Furthermore, the allocation approach aims to maximize fog device usage and minimize data communication over the network. The overview of our proposed strategy is presented below with equations.

The main goal G is to allocate the services s_i to fog devices D_F as much as possible in an effective manner. We can represent this as:

$$\max_{D_F} \sum_{i=1}^{N} s_i \cdot A_{i,F}$$

Where $A_{i,F}$ is a binary variable that represents whether or not service s_i is allocated to fog device D_F. If $A_{i,F} = 1$, then service s_i is allocated to fog device D_F, and if $A_{i,F} = 0$, then service s_i is not allocated to fog device D_F. The notation $\sum_{i=1}^{N} s_i \cdot A_{i,F}$ calculates the total number of services allocated to fog device D_F, where N is the total number of services. The objective is to maximize this quantity over all possible allocations to fog devices.

The services will be allocated by the fog device to either fog devices D_F or cloud devices D_C, dependent on the capabilities of the devices and the computing needs of the services. We can represent this as:

$$s_i \rightarrow \begin{cases} D_F(s_i), & \text{if } D_F \text{ can handle } s_i \\ D_C(s_i), & \text{otherwise} \end{cases}$$

The services will then be allocated according to their requirements to the fog devices since this is the focus of our strategy. We can represent this as:

$$s_i \rightarrow D_F(s_i)$$

If the fog devices are unable to manage the load, the remaining services will be allocated to the cloud-based devices. We can represent this as:

$$s_i \rightarrow D_C(s_i)$$

Therefore, to have efficient results we used both fog devices and cloud devices for service allocation. We can represent this by combining the previous two relations as follows:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

$$D_F(s_i), D_C(s_i) \rightarrow G(s_i)$$

4.1. Objective function

We have a multi-criteria optimization problem. The goal is to maximize the weighted sum of two objective functions, f1(x) for decreasing delay and f2(x) for optimizing the usage of resources. The weighting variables w1 and w2 are utilized to balance the relevance of the two objectives and govern the trade-off between the opposing aims. These could be objectives for performance for our strategy. The function that has to be maximized is as described below:

$$\text{maximize } fm(x) = -w_1 \cdot f_1(x) + w_2 \cdot f_2(x) \tag{1}$$

where $f_1(x)$ is the objective function for minimizing latency (i.e., the time it takes for a service to be allocated from fog to cloud), $f_2(x)$ is the objective function for maximizing fog resource utilization (i.e., using the fog devices as much as possible while minimizing resource wastage), x is a vector of variables, and w_1 and w_2 are weighting factors used to balance the importance of the two objectives.

The minus sign in front of $w_1 \cdot f_1(x)$ denotes that we are maximizing the negative of $f_1(x)$, which is the same as minimizing $f_1(x)$. Similarly, we are maximizing $f_2(x)$ by multiplying it with a positive weight w_2. The goal of integrating two objectives into a single objective function is to discover the optimal trade-off between the two of them. By maximizing $f_m(x)$, we mean finding x values that concurrently minimize $f_1(x)$ and maximum $f_2(x)$, with suitable weightings.

Best-Fit

We give best-fit code in Algorithm 1 to maximize device usage while delivering services to devices based on their capabilities. We need the requirements for service *serReq* and device capabilities *devCap* as input. Then, for each service, we find the smallest possible device capability that may accommodate the current service.

$$Service_{present} = \text{find\_min}(devCap_1, devCap_2, \ldots, devCap_n)$$

If a device is located, assign it to the present service. However, if a device cannot be discovered, discard that service and keep inspecting the other services. In the aforementioned method, we do not decompose the services into smaller services, but rather allocate them to one of the devices, either a fog device or a cloud device, according to their capabilities.

If a device is found, it should be assigned to the current service. If a device can not be found, disregard it and keep going through the other services. We do not break down the services into smaller ones, but rather assign them to one of the devices, either a fog device or a cloud device, according to their capabilities.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

Algorithm 1 Best Fit

Input: $devCap$[], $serReq$[]
Output: $allocID$[]
  for $x \leftarrow 0$ **to** $length(serReq) - 1$ **do**
    $bestFitID \leftarrow -1$
    for $y \leftarrow 0$ **to** $length(devCap) - 1$ **do**
      if $devCap[y] \geq serReq[x]$ **then**
        if $bestFitID = -1$ **then**
          $bestFitID \leftarrow y$
        else if $devCap[bestFitID] > devCap[y]$ **then**
          $bestFitID \leftarrow y$
        end if
      end if
    end for
    if $bestFitID \neq -1$ **then**
      $allocID[x] \leftarrow bestFitID$
      $devCap[bestFitID] \leftarrow devCap[bestFitID] - serReq[x]$
    end if
  end for

In other words, services are assigned to available devices according to best-fit criteria. The algorithm receives two parameters: $devCap$ and $serReq$. Each device's capacity is represented by $devCap$, and the service request is represented by $serReq$. The algorithm begins with two for-loops that are nested. The outer loop begins at 0 and continues to a length of $serReq$, whereas the inner loop begins at 0 and continues to the length of $devCap$. The outer loop processes every service request one at a time, while the inner loop checks every device's capacity to see if it can handle the present service request.

The $bestFitID$ is first assigned to -1. If the capacity of the device at index y is more than or equal to the capacity of the service request at index x, the $bestFitID$ is assigned to y within the inner loop. If $bestFitID$ remains -1, it indicates that no device is currently allocated to the service, and so the $bestFitID$ is assigned to y. If $bestFitID$ has recently been assigned, a comparison is done between the present device's capacity ($devCap[y]$) and the device assigned previously ($devCap[bestFitID]$). If the current device's capacity is smaller than that of the earlier assigned device, the $bestFitID$ is changed to the present device (y).

After the inner loop completes, if $bestFitID$ is not equal to -1, the current service is assigned to the device with the best fit ($bestFitID$). The allocation is documented by updating the $allocID$ list, and the capacity of the device is reduced by the service request. The algorithm will continue to run the outer loop until all of the service requests have been completed and the devices have been allocated to the services. The method returns $allocID$, which is the index of the device allocated to the service. The algorithm's result is the services allocated to the devices.

Worst-Fit

We give the worst fit code in Algorithm 2 for optimizing device usage while assigning services to devices and taking device capabilities into account. We need service requirements $serReq$ and device capabilities $devCap$ as input. Following that, we select each service and identify the most powerful device capable of supporting the current service.

$$Service_{present} = \text{find\_max}(devCap_1, devCap_2, \ldots, devCap_n)$$

If a device is found, it should be assigned to the current service. If a device cannot be found, disregard it and continue investigating the other services.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

Algorithm 2 Worst Fit

Input: $devCap$[], $serReq$[]
Output: $allocID$[]
  for $x \leftarrow 0$ **to** $length(serReq) - 1$ **do**
    $worstFitID \leftarrow -1$
    for $y \leftarrow 0$ **to** $length(devCap) - 1$ **do**
      if $devCap[y] \geq serReq[x]$ **then**
        if $worstFitID = -1$ **then**
          $worstFitID \leftarrow y$
        else if $devCap[worstFitID] < devCap[y]$ **then**
          $worstFitID \leftarrow y$
        end if
      end if
    end for
    if $worstFitID \neq -1$ **then**
      $allocID[x] \leftarrow worstFitID$
      $devCap[worstFitID] \leftarrow devCap[worstFitID] - serReq[x]$
    end if
  end for

The method then iterates through each service request in the $serReq$ array. It changes the value of $worstFitID$ to -1 for each service request, indicating that no device has been allocated to the service yet. The method then runs over the $devCap$ device capacities. If a device has adequate capacity to satisfy the current service request, the algorithm determines whether it is the first device discovered with sufficient capacity or if its capacity is more than the current $worstFitID$. If the device's capacity is greater, the algorithm assigns $worstFitID$ to the present device's ID.

Following the completion of the inner loop, the algorithm determines whether a device has been allocated to the present service request. If a device is allocated, the algorithm updates the $allocID$ list by distributing the current service request the value of $worstFitID$. The algorithm also decreases the allocated device's capacity by the magnitude of the service request. The aforementioned stages are repeated by the algorithm for all service requests. After the outer loop has been completed, the algorithm returns the $allocID$ list, which contains the list of allocated services to devices.

First-Fit

To maximize device utilization while allocating services to devices based on their capabilities, we give the first fit code in Algorithm 3. We need service requirements $serReq$ and device capabilities $devCap$ as input. Following that, we select every service and find out whether it is compatible with the current service. If the $devCap$ is equal to the $serReq$, assign and examine for the next $serReq$. If otherwise, proceed to investigate the next $devCap$.

Algorithm 3 First Fit

Input: $devCap$, $serReq$
Output: $allocID$[]
  for $x \leftarrow 0$ **to** $length(serReq) - 1$ **do**
    for $y \leftarrow 0$ **to** $length(devCap) - 1$ **do**
      if $devCap[y] \geq serReq[x]$ **then**
        $allocID[x] \leftarrow y$
        $devCap[y] \leftarrow devCap[y] - serReq[x]$
        break
      end if
    end for
  end for

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

In other words, the method has three inputs: $devCap$ (device capabilities), $serReq$ (service needs), and $allocID$ (allocated IDs). The algorithm's purpose is to allocate services in $serReq$ to devices in $devCap$ and record the allocation in $allocID$.

The algorithm begins with two nested for-loops, where x is the length of $serReq$ and y is the length of $devCap$. The method examines if $devCap[j]$ is larger than or equal to $serReq[x]$ in every iteration of the inner loop. If the condition is met, the algorithm assigns the present service ($serReq[x]$) to the current device ($devCap[j]$) by changing $allocID[x]$ to y and lowering the current device's capacity by the service requirement ($devCap[j] -= serReq[x]$). Finally, the algorithm exits the inner loop after locating a device capable of allocating the current service and proceeds to the following service in the outer loop. This operation is repeated by the algorithm until all services are assigned to devices.

Priority-based Service Allocation

We present the code of priority-based allocation in Algorithm 4 to select the service allocation process and if the services should be handled in the fog or cloud. We need service requirements $serReq$ and device capabilities $devCap$ as input. The services are then allocated to the fog devices using the $fogDevice(devCap, serReq)$ method. Following that, one of the algorithms presented previously will be executed. Following that, we will evaluate the capabilities of fog computing in order to allocate services. If no fog devices are available to handle the service, we assign it to cloud devices by calling and forwarding the remainder of the services to $cloudDevice(remainingSer)$.

Based on the importance of the service request, the algorithm decides whether to distribute services to fog or cloud devices. The algorithm prioritizes service requests by assigning them to the fog devices initially. The algorithm begins the process by determining the level of priority of the service demand. If the service is high-priority, the algorithm inputs the device capability and service requirement. If fog devices are capable of handling the services, they will be allocated to fog devices. If fog devices are unable to handle the low-priority service, the algorithm will allocate it to cloud devices instead.

The algorithm begins by allocating services to fog devices. To distribute services to fog devices, the algorithm employs one of three allocation algorithms: best-fit algorithm 1, worst-fit method 2, or first-fit algorithm 3. The method then iterates across the length of the service request using a for loop. In every iteration, the algorithm determines whether or not the service has already been assigned to a device by determining whether or not the $allocID$ is greater than -1. If the $allocID$ is not equal to -1, it is increased by one. If the $allocID$ is -1, the service request is saved in the $remainingSer$ list.

The algorithm checks if the $remainingSer$ list is empty at the end of the for loop. If the $remainingSer$ list is empty, the algorithm has been completed and all services have been allocated to the devices. If the $remainingSer$ list is not empty, the method "cloudDevice" with the $remainingSer$ list as input is called. The "cloudDevice" function is in charge of distributing the remainder of services to cloud devices. The priority-based allocation mechanism routes service requests to either the fog or cloud layers based on their priority. The method assigns services to fog devices utilizing one of three allocation techniques: best fit, worst fit, or first fit, with the remaining services given to the cloud.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

Algorithm 4 Priority-Based Allocation

Require: $devCap$: list of available devices, $serReq$: list of service requirements, $allocID$: list of allocation IDs
Ensure: $allocID$

```
remainingSer ← empty list
for each service request x in serReq do
    if priority of serReq[x] is high then
        Allocate to fog devices with sufficient capacity
        if allocID[x] ≠ −1 then
            Increment allocID[x] by 1
        else
            Append serReq[x] to remainingSer
        end if
    else if priority of serReq[x] is low then
        Allocate to fog devices with sufficient capacity
        if allocID[x] ≠ −1 then
            Increment allocID[x] by 1
        else
            if there is sufficient capacity in fog devices then
                Allocate serReq[x] to a fog device
            else
                Append serReq[x] to remainingSer
            end if
        end if
    end if
end for
if remainingSer is full then
    Allocate remaining services to cloud devices
end if
```

Sort-based Service Allocation

We employed Dual-Pivot Quicksort, an efficient sorting algorithm commonly used in computer science and data processing applications, especially for sorting primitive data types such as int, double, and float. Dual-Pivot Quicksort, according to the authors' findings [36], is typically quicker and more effective than alternative quicksort algorithms, especially on big datasets. They point out that Dual-Pivot Quicksort works effectively with both randomly ordered and partly sorted data, and that it has a minimal amount of comparisons and swaps.

Sorting may be a valuable technique in the context of fog computing for optimizing service allocation and lowering the level of complexity of service distribution for fog devices. Sorting the data prior to it reaching the fog devices makes it easier to deploy resources and maximize the network's overall performance. We classify the technical needs of services in ascending order, from least to greatest, to assist service allocation techniques for fog devices. We also sort the capabilities of fog devices. This allows for faster and more efficient service allocation.

4.2. Architecture

Our architecture is organized into three major sections: Sensor layer, Fog Layer, and cloud layer. First, the Sensor layer has IoMT sensors and IoMT devices that send data to the fog layer. Second, the fog layer is responsible for distributing services to devices in an effective and optimized manner by ensuring that all available resources are utilized and serving users by offering accessible services. Last, the cloud layer can manage all of the data and services, as well as provide essential services to the edge and fog layers.

The architecture shown in FIGURE 2 incorporates three devices, including IoMT, fog, and cloud, as shown below:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

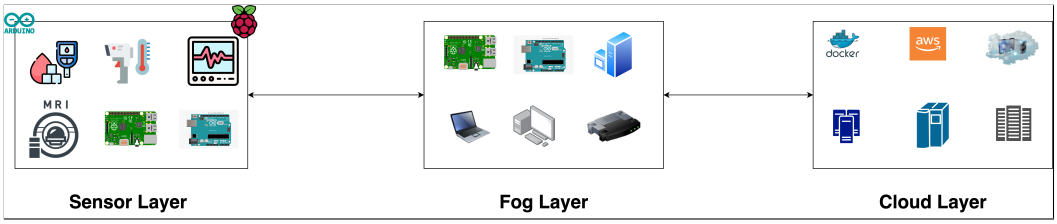


Figure 2. Fog based IoT Architecture

- The IoMT sensors and devices are located at the sensor layer of the system's network and are generally integrated into actual daily objects. IoMT sensors are tiny and affordable, making the process of installing IoMT sensors to things simple and economical. These devices communicate with the Fog devices using wireless communication. The IoT ecosystem should be beneficial in a variety of ways, including energy savings, lower costs, better resource use, and lower data transmission costs via the network. The IoMT sensors and devices produce data for each patient; then these data are fused to the service so that each service will have data about the patient and their medical diagnosis. Then, after fusing all the data into services, the services will be sent to the fog layer. Also, this layer is responsible for sorting and prioritising services to help the fog layer when allocating the services to the devices.
- The Fog devices reside adjacent to the sensor layer or within the communication channel and gather services and use a service allocation strategy to allocate the services to the fog devices as the priority is to process the services closer to the data source. However, whenever the fog devices cannot handle the services due to the lack of power of fog devices, then the services will be sent to the cloud using the proposed allocation strategy. Additionally, the fog devices are responsible for allocating services to either fog or cloud devices based on the priority of the service. Clearly, fog devices have very little power and a narrower global data perspective than cloud devices; thus, they can store less data and provide fewer services.
- The Cloud devices receive services from fog devices. The computational power and data storage capacity of the cloud is clearly greater than those of fog devices and IoMT devices. Cloud devices can be used for further analysis and storage when required to have the full picture of the data.

5. Experimental Setup

5.1. Dataset

We performed experiments in this study to evaluate the effectiveness of fog computing for service allocation. We utilized a customized dataset comprised of several fog devices configurations and services settings to conduct our research. The dataset used in this study was created in order to simulate an IoMT healthcare system.

Fog Devices' Configurations

The fog device configurations utilized in the experiments are detailed in Table 1. The table displays the experiment name, the number of fog devices, and the fog devices' RAM capacities (in Gigabytes). The experiment name is divided into three sections: (1) the allocation approach (Worst Fit, Best Fit, or First Fit), (2) the configuration types (Low, Medium, or High), and (3) the device capabilities (FDC1 to FDC4). Each method includes three tests of varying configurations (Low, Medium, and High).

FDC1, FDC2, FDC3, and FDC4 are the fog devices utilized in the experiment, each with 2 GB, 4 GB, 8 GB, and 16 GB RAM. The low configuration had 50 fog devices totaling 100 GB, the medium configuration had 15 FDC1, 15 FDC2, 10 FDC3, and 10 FDC4 fog devices totaling 330 GB, while the high configuration had 50 FDC4 fog devices totaling 800 GB. Table 1 displays the experiment name, the number of fog devices, and the fog devices' RAM capability (in Gigabytes).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

Table 1. The configurations of fog devices in the experiment

	FDC Setup	Fog devices and capabilities				
		FDC1 2 GB RAM	FDC2 4 GB RAM	FDC3 8 GB RAM	FDC4 16 GB RAM	Total FDC (GB)
1	Low	50 Fog devices	-	-	-	100 GB
2	Medium	15 Fog Devices	15 Fog Devices	10 Fog Devices	10 Fog Devices	330 GB
3	High	-	-	-	50 Fog Devices	800 GB

Service Setups

The technical demands of the services utilized in the experiment are provided in table 2. The services are labeled SR1, SR2, SR3, and SR4, and their RAM needs range from 1 MB to 2 GB. We assumed that the size of the services was the same as the technical requirements of the services in terms of size in GB. We deployed these services in various configurations to assess the performance of fog computing in various circumstances. A total of 800 services with varied technical needs were generated and the generated data has been used in all experiments. The half of the services (400) has high priority and the other half (400) has low priority. To create the dataset, simulations were run with fog devices configured as stated in Table 1 and services with varying technical demands shown in Table 2. The data was gathered and evaluated in the study. Because the dataset was generated at random, it was representative of real-world scenarios and provided various types of data for analysis. The dataset was used to assess the influence of various allocation techniques on system performance and resource consumption in an IoT healthcare system.

This arrangement was created to emulate real-world circumstances in which fog devices of diverse capacities may be required to host a variety of services with varying resource requirements. The research intended to evaluate and compare several methods for service allocation in fog computing environments through varying the number and capacity of fog devices as well as the resource needs of the services. Overall, the dataset utilized in the experiments offers a wide range of fog device and service configurations for evaluating fog computing performance.

Table 2. The setups of technical requirements of services in the experiments

	Services and Technical Requirements				
	SR1 $1MB - 255MB$ RAM	SR2 $256MB - 511MB$ RAM	SR3 $512MB - 1GB RAM$	SR4 $1GB - 2GB$ RAM	Total SR (GB)
no. of services	300 services	300 services	100 services	100 services	383 GB

5.2. Experiments

We categorized the experiments into three categories: those without priority and sort (standard), those with priority, and those with sort. Experiments with no priority or sort are used to allocate services without consideration for characteristics like priority or sort. The experiments with priority concentrate on the priority considerations when distributing services to fog devices; this indicates that services with a high priority will be delivered to fog devices first, followed by those with low priority. When the fog is lacking, the services will be assigned to cloud devices. The experiments with the sort initially sort the services in the sensor layer from small to big depending on their requirements in order to make the allocation process for fog computing easier and to support the algorithms in distributing the services efficiently. The three categories of experiments are used to evaluate our allocation strategy, so when a setup does not require factors, the first strategy is chosen, or if a setup requires a priority, the experiments with a priority are chosen, as we focus on deploying services to fog devices as much as possible.

We performed a total of three main experiments in each category of experiments. In every category (standard, priority, and sort), we use three algorithms with three different

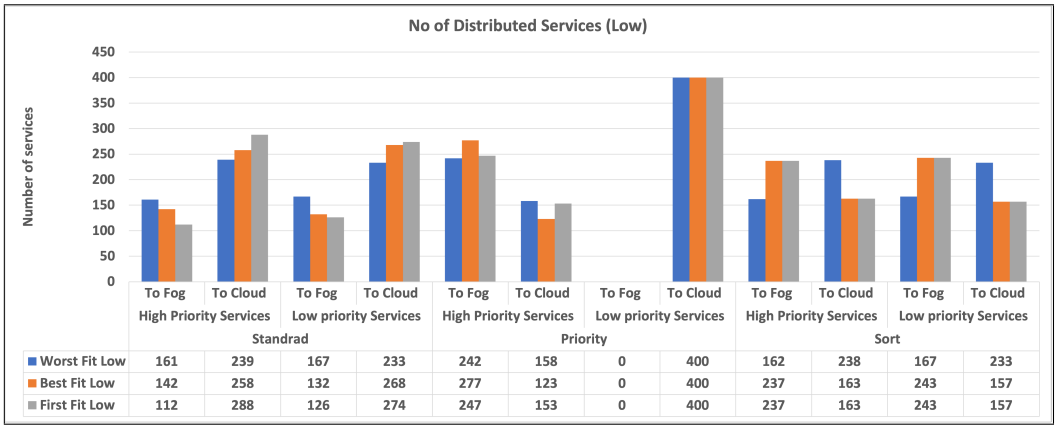
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

configurations namely low, medium, and high as mentioned earlier. In total, we conducted nine experiments for each category to discover the ideal configuration for fog devices in order to distribute services as effectively as feasible across fog devices. The configuration of fog devices may differ based on the experiment, as illustrated in Table 1. The first column lists the titles of the experiments, while the second column lists the capabilities of the fog devices. We deployed 800 services using a variety of capabilities (FDC1, FDC2, FDC3, and FDC4) to each of the 50 fog devices in experiments 1 through 9. The broad range of capabilities includes both highly specialized and relatively common equipment (running all capacities).

6. Results

In this section, we explore the outcomes of several techniques for allocating services to fog or cloud depending on priority, size, and algorithms employed. The outcomes of the allocation approach are shown. As previously stated, the technique would first allocate services to fog nodes based on their capabilities, and then assign services that could not be managed in the fog to cloud devices. We will examine and provide the findings of the three methods namely worst fit, best fit, and first fit. According to Statista [37], the average upload speed utilizing mobile to transmit services from fog to the cloud is 8.5 Mbps. However, the average upload speed using fixed broadband to transfer services from fog to cloud is 28.5 Mbps.

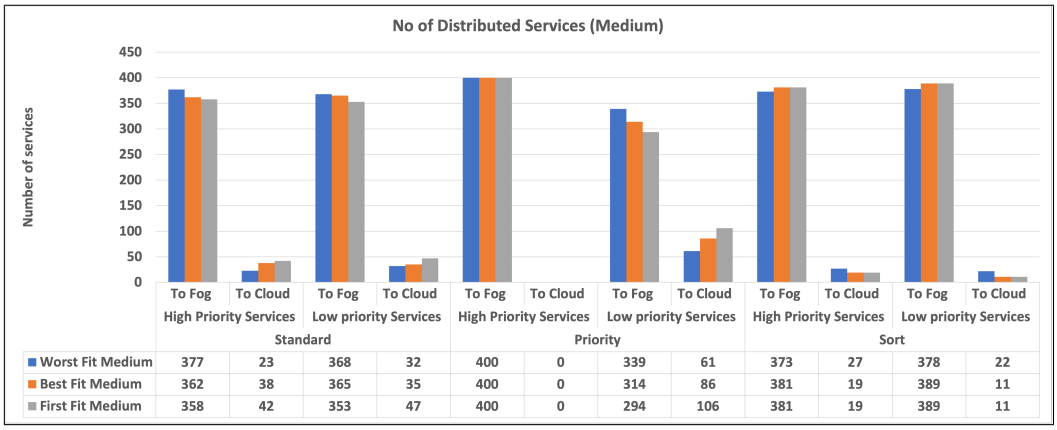
To begin, we give two charts in Figure 3 and 4 that illustrates the distribution of high and low-priority services to fog or cloud using various methodologies. Standard, Priority, and Sort are the strategies used in our experiments. The allocation is provided separately for high- and low-priority services. Second, in Figure 5a and 5b, we provide a bar chart that indicates the number of unused fog devices in GB after assigning services to fog. Furthermore, the values in the table represent the amount of unused RAM in GB for each algorithm and each level of fog device configuration: low and medium. In the charts, we did not show the results of high level configuration of fog devices as all the services were handled in the fog. Then, in Figure 6a and 6b, we show a chart and a table that provide information about allocating services to fog or cloud using three different strategies: standard, priority, and sort. The services are categorized as high or low priority, and their sizes are indicated in gigabytes (GB). Finally, we talk about data from Table 3, which illustrates how long it takes for services to travel from fog devices to the cloud using different techniques. The findings are reported in terms of the time required to assign services via mobile and broadband networks, and the time is determined based on the upload speed supplied by Statista [37].



	Standrad				Priority				Sort			
	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud
	High Priority Services		Low priority Services		High Priority Services		Low priority Services		High Priority Services		Low priority Services	
Worst Fit Low	161	239	167	233	242	158	0	400	162	238	167	233
Best Fit Low	142	258	132	268	277	123	0	400	237	163	243	157
First Fit Low	112	288	126	274	247	153	0	400	237	163	243	157

Figure 3. The number of allocated services to fog and cloud (Low)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1



	Standard				Priority				Sort			
	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud	To Fog	To Cloud
	High Priority Services		Low priority Services		High Priority Services		Low priority Services		High Priority Services		Low priority Services	
Worst Fit Medium	377	23	368	32	400	0	339	61	373	27	378	22
Best Fit Medium	362	38	365	35	400	0	314	86	381	19	389	11
First Fit Medium	358	42	353	47	400	0	294	106	381	19	389	11

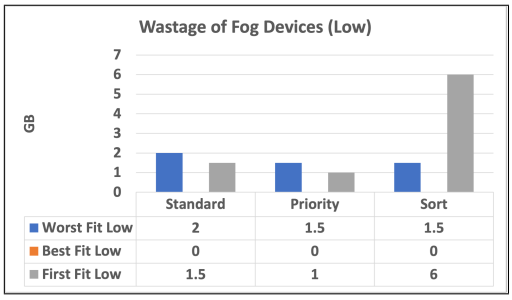
Figure 4. The number of allocated services to fog and cloud (Medium)

Figure 3 and 4 display the quantity of high and low-priority services assigned to fog or cloud using various techniques, including Standard, Priority, and Sort. The allocation is displayed separately for high-priority and low-priority services. The standard refers to a situation in which there is no differentiation between high and low-priority services. Priority denotes a situation in which high-priority services take precedence over low-priority services. Sort denotes a scenario in which services are first sorted and then assigned using the Best Fit, Worst Fit, or First Fit algorithm. According to the findings, the allocation approach has a substantial influence on the number of services provided to fog and cloud environments. In general, the Priority method results in more high-priority services being assigned to the fog environment and more low-priority services being assigned to the cloud environment. When it comes to sorting services, the allocation approach has less of an influence on the number of services distributed to the fog and cloud environments. However, the allocation algorithm utilized does make a difference. For example, the Worst Fit algorithm allocates more services to the fog environment, but the First Fit algorithm allocates more services to the cloud environment.

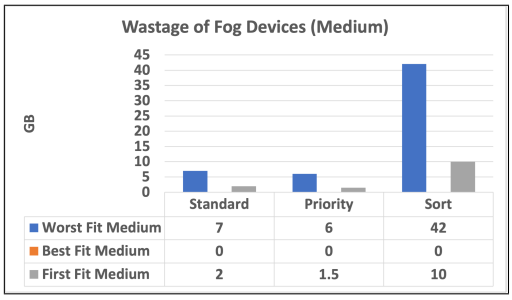
For each combination of distribution strategy and service priority, the chart illustrates the number of distributed high and low-priority services. The charts show that the number of high-priority services allocated to the fog is lower than in the cloud for the Standard strategy, while the converse is true for low-priority services. We did not include the results of high-capacity fog devices in the table since their exceptional capabilities allowed them to manage all services in the fog as mentioned earlier. It is clear from the charts that all algorithms are doing well in terms of allocating services to fog and cloud, but as our intention was to push the processing near the data source therefore priority strategy can be a good choice. Among the three algorithms, we can realize that the best can be selected as the best in most cases.

Overall, our findings imply that careful evaluation of allocation methods and algorithms could be useful in optimizing service allocation in fog and cloud situations. A Priority approach, in particular, that prioritizes high-priority services, can assist guarantee that important services are allocated to the fog environment, where they can be handled fast and efficiently. whereas the standard and sort strategies appear inefficient since they allocate services without regard for priority.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1



(a) Low Configuration

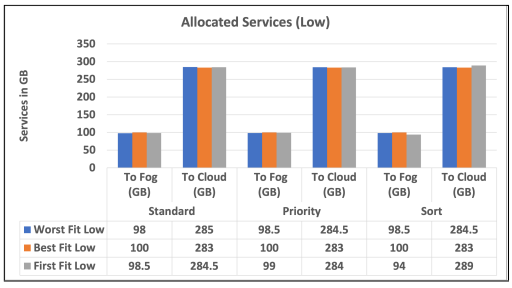


(b) Medium Configuration

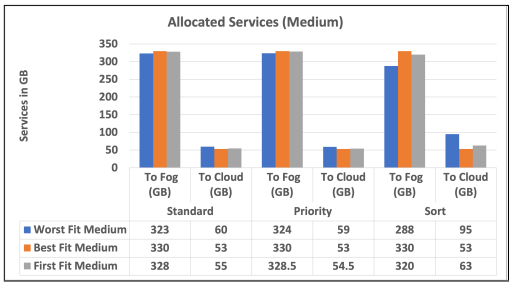
Figure 5. Wastage of Fog Devices

The bar charts in Figure 5a and 5b illustrate the number of unused fog devices in GB after assigning services to fog. The values in the table indicate the quantity of unused RAM in GB for each algorithm and fog device configuration: low, medium, and high. Worst Fit Low: This allocation approach directs resources to the fog gadget that produces the most waste. According to the chart, this technique wastes 2 GB for devices with low capacity within the standard strategy. When priority is set, the wasted space is reduced to 1.5 GB. Worst Fit Medium: This allocation approach produces the largest waste among medium and low configuration devices. According to the charts, this technique wastes 7 GB for devices in standard and 6 GB with priority. However, when medium configuration devices are sorted, the strategy wastes 42 GB. Best Fit: This allocation strategy allocates resources to the fog device with the least wastage. The table shows that this strategy results in 0 GB wastage for devices in all strategies. First Fit Low: This allocation strategy allocates resources to the first available fog device. The table shows that this strategy results in 1.5 GB wastage for devices with low capacity within the standard and 6 GB within the sort strategy, but 1 GB within the priority strategy. First Fit Medium: This allocation strategy allocates resources to the first available fog device among devices with medium capacity. The table shows that this strategy results in 2 GB wastage for devices within the standard and 1.5 GB wastage when priority is given, however, the wastage increases to 6 GB when the sort strategy is used.

Overall, we did not reveal the high-config devices since there was no waste because the fog devices were in high capabilities, which allocated all services to the fog devices. Finally, the table shows that the Worst Fit strategy results in the most waste for both low and medium-capacity devices, especially when the sort is provided. For all circumstances, the Best Fit technique results in the least amount of waste. The First Fit approach stands somewhere between the other two.



(a) Low Configuration



(b) Medium Configuration

Figure 6. RAM of Allocated Services

The charts in Figure 6a and 6b show the RAM size of services allocated to fog or cloud based on three alternative strategies: standard, priority, and sort. The services are classified as high or low priority, and their volumes are measured in gigabytes (GB). The low-configuration devices have a total of 100 GB and the medium-configuration devices have a total of 330 GB as mentioned earlier. This indicates that in each configuration of low, medium, and high experiments, there is a maximum size of RAM to handle services. For

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1523.v1

example, the worst fit low experiment has allocated 98 GB of services to fog devices those size out of 100 GB, the remaining 2 GB is the wastage which is discussed earlier. It is clear that the strategy used most of the fog devices' RAM, but most of the services traveled to the cloud because of the limited capabilities of the fog devices. It is clear that the best fit low and medium strategies are the best among others in terms of the usage of fog devices and allocating all services that can fit the fog devices without wastage. In the low configuration, the sort strategy stands better than the standard strategy in worst fit and best fit, but worse in sort. In the medium configuration, the sort strategy has resulted better than the standard in all algorithms. Overall, the table provides valuable information about the allocation of services to the fog or cloud and highlights the importance of considering service size, algorithm, and strategy when making these allocation decisions.

Table 3. Time of service allocation (fog to the cloud)

	Without		Priority		Sort	
	Mobile	Broadband	Mobile	Broadband	Mobile	Broadband
Worst Fit Low	3d 7h 45m	23h 45m	3d 7h	23h 30m	3d 7h	23h 30m
Worst Fit Medium	15h	4h 30m	15 h	4h 30m	1 day	7h 20m
Worst Fit High	0	0	0	0	0	0
Best Fit Low	3d 6h	23h 25m	3d 6h	23h 15m	3d 7h	23h 30m
Best Fit Medium	12 h	3h 45m	12 h	3h 45m	15h 43m	4h 40m
Best Fit High	0	0	0	0	0	0
First Fit Low	3d 7h	23h 30m	3d 7h	23h 30m	3d 8h	1d
First Fit Medium	13 h	4h	13 h	4h	16 h	4h 45m
First Fit High	0	0	0	0	0	0

Table 3 illustrates how long it takes for services to travel from fog devices to the cloud using various techniques. Standard, Priority, and Sort were the algorithms employed. The findings are reported in terms of the amount of time required to distribute services using mobile and broadband networks. The data clearly shows that the time required to distribute services to the cloud is often longer for the mobile network than for the broadband network. This is most likely due to the fact that the mobile network has more constraints and requirements for service distribution than the broadband network. In terms of the various algorithms utilized, we can observe that the Best Fit algorithm outperformed the others for both mobile and broadband networks. For the mobile network, the Worst Fit algorithm performed the worst, but it was comparable with the other methods for the broadband network. The First Fit method performed effectively in the broadband network but not so well on the mobile network. It is clear that the high configuration setup of fog devices has no cost over the network as all the services have been handled locally. Overall, the table findings indicate that the Best Fit algorithm may be the most effective method for distributing services from fog to the cloud. However, the particular method used may be determined by the network's specific requirements and constraints.

6.1. Total services allocated to the cloud

In this experiment, we allocated all services to the cloud in order to compare our techniques against allocation without a strategy and to investigate architecture-based allocation. Table 4 clearly shows that "No of Services" signifies 0 services out of 800 services assigned to fog devices and 800 services out of 800 services given to cloud computing. In other words, the fog devices receive 0% of the services, while the cloud devices receive 100% of the services. The column "Allocated Services in GB" in the table indicates that 0 services size in GB and 0 GB are allocated to the fog and 383 GB are allocated to the cloud devices. The time of service allocation is then represented as "Time of service allocation (mobile)," which indicates that the amount of time it takes for a mobile network to allocate the services to fog is obviously 0 because there are no costs associated with data communication over

 doi:10.20944/preprints202305.1523.v1

the network because it is located locally, but the duration it takes to allocate the services (383 GB) from fog to cloud is 4 days, 11 hours, and 0 minutes using the mobile network. However, "Time of service allocation (broadband)" means that the period of time it takes to allocate services from fog to the cloud using the fixed broadband network is 1 day and an hour, and 40 minutes.

Table 4. Total services allocation

Total services in GB	To Fog	To Cloud	Total
No of Services	0	800	800
Percentage	0	100%	100%
allocated Services in GB	0	383 GB	383 GB
Time of allocating services (mobile)	0	4d 11h	4d 11h
Time of allocating services (broadband)	0	1d 8h	1d 8h

We have conducted three experiments using methods, namely best fit, first fit, and worst fit. Based on the results, the worst fit results were lower than the best fit and first fit, it is clear that the data communication over the network is reduced with all strategies when compared to total service allocation to the cloud, and most of the services are allocated to the fog devices. Since reducing data transfer across the network was our primary goal, it fits with our proposed technique. Furthermore, the second aim was to use the priority aspect to allocate the services as much as possible to the fog devices, and based on the results, this aim was achieved as 90% or more of the services have been allocated to the fog devices.

We ran a total of twenty-seven experiments utilizing the best fit, first fit, and worst fit methodologies. According to the results, the worst fit and first fit results were lower than the best fit, indicating that data traffic via the network is decreased with all methods when compared to the entire service allocation to the cloud, with the fog devices receiving the majority of the services. It fits with our proposed technique because reducing data transfer across the network was our primary goal. Furthermore, the second goal was to use the priority aspect to allocate as many high-priority services as possible to fog devices, and based on the results, this goal was achieved because 90% or more of the priority services were allocated.

We have compared the performance of the algorithms namely best fit, worst fit, and first fit while allocating services to fog devices in the variable capability of fog devices and variable service requirements. The focus was on the number of services allocated to fog, resource usage, and data communication over the network. The First Fit Algorithm allocates services to the first fog device which can handle them. It may, however, produce some wastage, lowering fog device usage. The use of fog devices was approximately 90%. The best Fit Algorithm allocates services to the smallest fog device that can handle them. This algorithm tries to reduce waste and maximize fog device use. The wastage was found to be the lowest. The use of fog devices in the priority strategy was 100% and in other strategies was more than 90%. The worst Fit Algorithm allocates services to the largest available fog device that can handle them. However, this algorithm leads to increased wastage. The wastage was found to be the highest. Additionally, the fog device usage was better than the first fit, and better than the best fit in some of the experiments, but generally worse than the best fit. In terms of waste, the Best Fit Algorithm outperforms the first fit and worst fit, attaining the lowest wastage. The Worst Fit Algorithm generated the most wastage. As a result, the Best Fit Algorithm is regarded as the most advantageous of the three for the variable capability of fog devices and variable services requirements scheme since it reduces waste and maximizes fog device use. Based on our knowledge that the worst fit can lead to high wastage (fragmentation) and the best fit can be best in terms of wastage. However, in some cases, the worst fit can increase the usage of resources [38].

 doi:10.20944/preprints202305.1523.v1

7. Discussion and Evaluation:

We used two commonly used evaluation measures to assess the effectiveness of our model: the allocation success rate (ASR) and the average resource usage (ARU). The ASR calculates the number of services successfully assigned to fog devices/cloud devices. The ARU calculates the percentage of RAM used by all fog devices/cloud servers. To illustrate the robustness of our model, we did sensitivity analysis by adjusting various factors such as dataset size, technological requirement distribution, and the number of fog devices/cloud servers. The findings indicate that our model is not overfitting to a specific dataset or set of parameters. We compared our strategies including standard, priority, and sort performance while considering commonly used algorithms in the literature: the best-fit, first-fit and worst-fit algorithms. Our model performs well in terms of ASR and ARU, according to the results. We used various data groups randomly generated with different distributions and sizes to demonstrate the validity of our model. The findings reveal that our proposal performs well with the given setups and distributions. Our methodology proposes a realistic and efficient solution to the service allocation problem in fog computing applications.

Based on the results, we observed several observations as follows

The number of services allocated to fog devices

As previously said, our primary aim in fog computing is to allocate services as close to the data as possible while simultaneously maximizing resource consumption, network data transfer, and balancing service allocation. The amount of services allocated is influenced by a variety of factors, including strategy, capabilities, and service requirements. According to the findings, the allocation method and algorithm have a considerable influence on the number of services assigned to the fog and cloud environments. The Priority approach allocates more high-priority services to the fog environment and more low-priority services to the cloud environment. The allocation algorithm utilized also influences service allocation, with the Worst Fit algorithm allocating can allocate more services to the fog environment and the First Fit algorithm allocating fewer services to the fog environment, but the worst fit can result in more wastage. The results also demonstrate that allocating services without regard for priority or employing a sorting technique without regard for priority is inefficient. According to the study, a thorough evaluation of the allocation method and algorithm is required to maximize service allocation in fog and cloud situations.

Resources usage

The results shown in Figure 5 indicate that the strategy used for distributing services to fog devices can have a considerable influence on the amount of unused RAM in the fog. The Best Fit method produces the least waste in all circumstances, whereas the Worst Fit strategy produces the most waste, especially when priority is provided. In terms of waste, the First Fit technique lies in between the other two. The chart also demonstrates that high-config devices did not waste any resources because they could handle all of the services assigned to them. These findings imply that careful study of the allocation approach and algorithm can aid in optimizing the use of fog resources and minimizing wastage.

Data communication over the network

The results clearly indicate that most of our experiments could result in low data communication over the network compared to total services allocated to the cloud without strategies 6.1 which is the traditional way of allocating services without considering the power of fog and strategies. The variations of fog device capabilities used in our research can help us choose a combination that depends on a number of factors, such as the technical requirements of services as well as the RAM requirements of fog nodes. To decide which allocation technique is optimal for allocating services to fog or cloud devices, we may consider the service requirements and device capabilities. The results show the amount

 doi:10.20944/preprints202305.1523.v1

of time taken to allocate mobile and broadband services to different devices using three different algorithms: Worst Fit, Best Fit, and First Fit. The values in the table represent the time taken in hours and minutes for each algorithm and device configuration: low, medium, and high capability. The findings imply that Best Fit is the most efficient algorithm, as it takes the least amount of time to allocate services for all device types. The Worst Fit algorithm results in the longest allocation time for low and medium-capacity devices, while the First Fit algorithm falls somewhere in between the other two algorithms. Additionally, the high-capacity devices did not have any allocation time as all services were allocated to them. It is clear that the data communication over the network is reduced with all strategies, and most of the services are allocated to fog devices. Since reducing data transfer across the network was our primary goal, it fits with our proposed strategy. Furthermore, the second aim was to use the priority aspect to allocate the high-priority services as much as possible near the fog devices, and based on the results, this aim was achieved as 90% or more of the services have been allocated to the fog devices and the data communication is reduced by 82% compared to 6.1.

8. Conclusion

In conclusion, this paper developed an efficient service allocation strategy priority based service allocation (PSA) and sort based service allocation (SSA) with lower bandwidth consumption, faster response times, improved resource usage, and the identification of the best method for processing data at a large scale. As a result, we determined the capabilities of fog and IoT devices, as well as the technical needs of the services, in order to efficiently allocate the services to the devices. our proposed service allocation strategy was significant because providing services to devices in the IoT is a difficult process due to the many types of devices and their capabilities. Then, We proposed our allocation method, which utilizes a list of every fog device in the network. We looked at how fog computing may improve service quality while fulfilling the expectations of fog applications for optimal resource consumption. This included the necessity of allocating services to devices and optimizing resource use. Our findings demonstrated that our approach was suitable for the given setting and dataset. Our results showed that by distributing most services locally in fog, we reduced data transmission over the network by 82%, and we maximized the number of distributed services to fog devices by 90% while minimizing the wastage of fog resources.

9. Future Work

Our future work includes the following open challenges:

- Privacy: Privacy: Fog nodes acquire a considerable quantity of personal information from fog applications such as smart healthcare. Despite the fact that some researchers utilize privacy-preserving techniques on fog nodes [39], based on our knowledge that no standard authentication solution exists.
- Security is a serious concern because fog devices lack resources and are positioned in risky environments, leaving them open to attack. As a result, designing a lightweight, quick, and trustworthy safety algorithm remains a tough task. Only a few researchers are currently focusing on fog computing security challenges [39], and there are several outstanding issues such as dynamic authentication, access controls, external threats, and intrusion detection.
- Energy usage: As fog devices have limited battery capacity, energy awareness remains a problem that remains in fog computing. Some researchers are concerned with optimizing energy use, [39], while others are worried about the proper use of bandwidth in data transfer, battery waste, and battery-draining issues.

Security is a serious concern because fog devices lack resources and are positioned in risky environments, leaving them open to attack. As a result, designing a lightweight, quick, and trustworthy safety algorithm remains a tough task. Only a few researchers are currently focusing on fog computing security challenges [39], and there are several outstanding issues such as dynamic authentication, access controls, external threats, and intrusion detection.

 doi:10.20944/preprints202305.1523.v1

Supplementary Materials: Not applicable.

Author Contributions: Conceptualization, A.A.A. and R.A.A.; methodology, A.A.A., B.A., R.A. and R.A.A.; software, A.A.A. validation, M.Y.D.and B.A.; formal analysis, R.A.A.; investigation, B.A. and R.A.; resources, B.A.; data curation, A.A.A.; writing—original draft preparation, A.A.A. and R.A.A.; writing—review and editing, A.A.A, R.A.A., B.A.and M.Y.D.; visualization, R.A.A.; supervision, M.Y.D., and A.A.A.; project administration, A.A.A.; funding acquisition, R.A., A.A.A. and R.A.A.; All authors have read and agreed to the published version of the manuscript.

Funding: This project was funded by the Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah, under grant no. IFPIP: 1033-611-1443. The authors, therefore, acknowledge with thanks DSR for technical and financial support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Betty Jane, J.; Ganesh, E. Big data and internet of things for smart data analytics using machine learning techniques. In Proceedings of the International conference on computer networks, big data and IoT. Springer, 2020, pp. 213–223.
2. Wang, N.; Varghese, B.; Matthaiou, M.; Nikolopoulos, D.S. ENORM: A framework for edge node resource management. *IEEE transactions on services computing* **2017**, *13*, 1086–1099.
3. de Moura Costa, H.J.; da Costa, C.A.; da Rosa Righi, R.; Antunes, R.S. Fog computing in health: A systematic literature review. *Health and Technology* **2020**, *10*, 1025–1044.
4. Pisani, F.; de Oliveira, F.M.C.; Gama, E.S.; Immich, R.; Bittencourt, L.F.; Borin, E. Fog computing on constrained devices: Paving the way for the future iot. *Advances in Edge Computing: Massive Parallel Processing and Applications* **2020**, *35*, 22–60.
5. Hong, C.H.; Varghese, B. Resource management in fog/edge computing: a survey on architectures, infrastructure, and algorithms. *ACM Computing Surveys (CSUR)* **2019**, *52*, 1–37.
6. Ortiz, G.; Zouai, M.; Kazar, O.; Garcia-de Prado, A.; Boubeta-Puig, J. Atmosphere: Context and situational-aware collaborative IoT architecture for edge-fog-cloud computing. *Computer Standards & Interfaces* **2022**, *79*, 103550.
7. Alturki, B.; Reiff-Marganiec, S.; Perera, C.; De, S. Exploring the effectiveness of service decomposition in fog computing architecture for the Internet of Things. *IEEE Transactions on Sustainable Computing* **2019**.
8. Puliafito, C.; Mingozzi, E.; Longo, F.; Puliafito, A.; Rana, O. Fog computing for the internet of things: A survey. *ACM Transactions on Internet Technology (TOIT)* **2019**, *19*, 1–41.
9. Laroui, M.; Nour, B.; Moungla, H.; Cherif, M.A.; Afifi, H.; Guizani, M. Edge and fog computing for IoT: A survey on current research activities & future directions. *Computer Communications* **2021**, *180*, 210–231.
10. Mahmud, R.; Toosi, A.N.; Ramamohanarao, K.; Buyya, R. Context-aware placement of industry 4.0 applications in fog computing environments. *IEEE Transactions on Industrial Informatics* **2019**, *16*, 7004–7013.
11. Maia, A.M.; Ghamri-Doudane, Y.; Vieira, D.; de Castro, M.F. A multi-objective service placement and load distribution in edge computing. In Proceedings of the 2019 IEEE global communications conference (GLOBECOM). IEEE, 2019, pp. 1–7.
12. Behravesh, R.; Coronado, E.; Harutyunyan, D.; Riggio, R. Joint user association and VNF placement for latency sensitive applications in 5G networks. In Proceedings of the 2019 IEEE 8th International Conference on Cloud Networking (CloudNet). IEEE, 2019, pp. 1–7.
13. Lai, P.; He, Q.; Abdelrazek, M.; Chen, F.; Hosking, J.; Grundy, J.; Yang, Y. Optimal edge user allocation in edge computing with variable sized vector bin packing. In Proceedings of the Service-Oriented Computing: 16th International Conference, ICSOC 2018, Hangzhou, China, November 12-15, 2018, Proceedings 16. Springer, 2018, pp. 230–245.
14. Yu, N.; Xie, Q.; Wang, Q.; Du, H.; Huang, H.; Jia, X. Collaborative service placement for mobile edge computing applications. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM). IEEE, 2018, pp. 1–6.
15. Moubayed, A.; Shami, A.; Heidari, P.; Larabi, A.; Brunner, R. Edge-enabled V2X service placement for intelligent transportation systems. *IEEE Transactions on Mobile Computing* **2020**, *20*, 1380–1392.
16. Maiti, P.; Shukla, J.; Sahoo, B.; Turuk, A.K. QoS-aware fog nodes placement. In Proceedings of the 2018 4th International Conference on Recent Advances in Information Technology (RAIT). IEEE, 2018, pp. 1–6.
17. Huang, M.; Liang, W.; Shen, X.; Ma, Y.; Kan, H. Reliability-aware virtualized network function services provisioning in mobile edge computing. *IEEE Transactions on Mobile Computing* **2019**, *19*, 2699–2713.
18. Ma, H.; Zhou, Z.; Chen, X. Leveraging the power of prediction: Predictive service placement for latency-sensitive mobile edge computing. *IEEE Transactions on Wireless Communications* **2020**, *19*, 6454–6468.
19. Peng, Q.; Xia, Y.; Feng, Z.; Lee, J.; Wu, C.; Luo, X.; Zheng, W.; Pang, S.; Liu, H.; Qin, Y.; et al. Mobility-aware and migration-enabled online edge user allocation in mobile edge computing. In Proceedings of the 2019 IEEE International Conference on Web Services (ICWS). IEEE, 2019, pp. 91–98.
20. Toka, L.; Haja, D.; Kőrösi, A.; Sonkoly, B. Resource provisioning for highly reliable and ultra-responsive edge applications. In Proceedings of the 2019 IEEE 8th International conference on cloud networking (CloudNet). IEEE, 2019, pp. 1–6.

21. Mseddi, A.; Jaafar, W.; Elbiaze, H.; Ajib, W. Intelligent resource allocation in dynamic fog computing environments. In Proceedings of the 2019 IEEE 8th International Conference on Cloud Networking (CloudNet). IEEE, 2019, pp. 1–7.
22. Badri, H.; Bahreini, T.; Grosu, D.; Yang, K. Energy-aware application placement in mobile edge computing: A stochastic optimization approach. *IEEE Transactions on Parallel and Distributed Systems* **2019**, *31*, 909–922.
23. Haja, D.; Szalay, M.; Sonkoly, B.; Pongracz, G.; Toka, L. Sharpening kubernetes for the edge. In Proceedings of the Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos, 2019, pp. 136–137.
24. Mahmud, R.; Srirama, S.N.; Ramamohanarao, K.; Buyya, R. Profit-aware application placement for integrated fog–cloud computing environments. *Journal of Parallel and Distributed Computing* **2020**, *135*, 177–190.
25. Deng, R.; Lu, R.; Lai, C.; Luan, T.H. Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing. In Proceedings of the 2015 IEEE international conference on communications (ICC). IEEE, 2015, pp. 3909–3914.
26. Deng, R.; Lu, R.; Lai, C.; Luan, T.H.; Liang, H. Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption. *IEEE internet of things journal* **2016**, *3*, 1171–1181.
27. Gu, L.; Zeng, D.; Guo, S.; Barnawi, A.; Xiang, Y. Cost efficient resource management in fog computing supported medical cyber-physical system. *IEEE Transactions on Emerging Topics in Computing* **2015**, *5*, 108–119.
28. Kashani, M.H.; Ahmadzadeh, A.; Mahdipour, E. Load balancing mechanisms in fog computing: A systematic review. *arXiv preprint arXiv:2011.14706* **2020**.
29. Korte, B.H.; Vygen, J.; Korte, B.; Vygen, J. *Combinatorial optimization*; Vol. 1, Springer, 2011.
30. man Jr, E.C.; Garey, M.; Johnson, D. Approximation algorithms for bin packing: A survey.
31. Laabadi, S.; Naimi, M.; El Amri, H.; Achchab, B. A binary crow search algorithm for solving two-dimensional bin packing problem with fixed orientation. *Procedia Computer Science* **2020**, *167*, 809–818.
32. Anand, S.; Guericke, S. A bin packing problem with mixing constraints for containerizing items for logistics service providers. In Proceedings of the Computational Logistics: 11th International Conference, ICCL 2020, Enschede, The Netherlands, September 28–30, 2020, Proceedings. Springer, 2020, pp. 342–355.
33. Chraibi, A.; Alla, S.B.; Ezzati, A. An efficient cloudlet scheduling via bin packing in cloud computing. *International Journal of Electrical and Computer Engineering* **2022**, *12*, 3226.
34. Goyal, T.; Singh, A.; Agrawal, A. Cloudsim: simulator for cloud computing infrastructure and modeling. *Procedia Engineering* **2012**, *38*, 3566–3572.
35. Armant, V.; De Cauwer, M.; Brown, K.N.; O'Sullivan, B. Semi-online task assignment policies for workload consolidation in cloud computing systems. *Future Generation Computer Systems* **2018**, *82*, 89–103.
36. Aftab, A.; Ali, M.A.; Ghaffar, A.; Shah, A.U.R.; Ishfaq, H.M.; Shujaat, M. Review on Performance of Quick Sort Algorithm. *International Journal of Computer Science and Information Security (IJCSIS)* **2021**, *19*.
37. Taylor, P. Statista Average Global Broadband Download amp; Upload Speed 2022, 2023.
38. Mandal, A.K.; Baruah, D.K.; Medak, J.; Gogoi, N.; Gogoi, P. CRITICAL SCRUTINYOF MEMORY ALLOCATION ALGORITHMS: FIRST FIT, BEST FIT AND WORST FIT. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)* **2020**, *11*, 2185–2194.
39. Das, R.; Inuwa, M.M. A review on fog computing: issues, characteristics, challenges, and potential applications. *Telematics and Informatics Reports* **2023**, p. 100049.