Spring 2018

PRIORITIZED TASK SCHEDULING IN FOG COMPUTING

TEJASWINI CHOUDHARI
San Jose State University

Writing Project

PRIORITIZED TASK SCHEDULING IN FOG COMPUTING

Final Report

Author

Tejaswini Choudhari

CS 298

February 2018

Advisor

Dr. Melody Moh

A Writing Project Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment of the Requirements for the Degree: Master of Science

The Designated Committee Approves the Master's Project Titled

PRIORITIZED TASK SCHEDULING IN FOG COMPUTING

By

TEJASWINI CHOUDHARI

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

FEBRUARY 2018

Dr. Melody Moh Signature:_________________________
Department of Computer Science

Dr. Robert Chun Signature:_________________________
Department of Computer Science

Dr. Teng Moh Signature:_________________________
Department of Computer Science

Acknowledgements

I would like to thank my advisor, **Dr. Melody Moh** for her continuous guidance and support throughout the course of this project. Her experienced insight and encouragement gave me motivation to keep working. The knowledge and wisdom provided by Dr. Moh has pushed me to achieve results efficiently.

I would also like to thank the committee members, **Dr. Robert Chun** and **Dr. Teng Moh** for the knowledge they have imparted during their coursework as it helped me to reach a solution with the help of this knowledge. Also, I want to thank them for their continuous support and guidance.

Lastly, I would like to thank my family members and friends for their moral and emotional support throughout the course of the project.

ABSTRACT

Cloud computing is an environment where virtual resources are shared among the many users over network. A user of Cloud services is billed according to pay-per-use model associated with this environment. To keep this bill to a minimum, efficient resource allocation is of great importance. To handle the many requests sent to Cloud by the clients, the tasks need to be processed according to the SLAs defined by the client. The increase in the usage of Cloud services on a daily basis has introduced delays in the transmission of requests. These delays can cause clients to wait for the response of the tasks beyond the deadline assigned. To overcome these concerns, Fog Computing is helpful as it is physically placed closer to the clients. This layer is placed between the client and the Cloud layer, and it reduces the delay in the transmission of the requests, processing and the response sent back to the client greatly. This paper discusses an algorithm which schedules tasks by calculating the priority of a task in the Fog layer. The tasks with higher priority are processed first so that the deadline is met, which makes the algorithm practical and efficient.

TABLE OF CONTENTS

List of Tables

List of Figures

1. INTRODUCTION

Cloud computing has a strong foothold in the technology world as it provides with various useful services to the end users. Cloud computing provides with many features like, huge amounts of processing power, great storage provision, massive network between the processing nodes and the data generators, pay-per-use model. Cloud provides the users with a set of deployment models which can be used to take advantage of the Cloud features. The models include, public Cloud, private Cloud, hybrid Cloud and community Cloud. The public Cloud model caters to the needs of users over the Internet. The private Cloud model provides services to a particular closed network, which is not open to general public. Private Cloud is secured against the outside world. Hybrid Cloud model is a combination of public and private Cloud. An organization can have private Cloud for their working teams/groups and it can have a public Cloud for marketing purposes. A community Cloud can be shared among multiple organizations which have similar objectives.

Cloud also provides with the famous service models, Platform-as-a-service(PaaS), Software-as-a-Service(SaaS) and Infrastructure-as-a-Service(IaaS). PaaS is used when an organization wants to build their own software and deliver it to the users over a platform. Here, the hardware and the management of resources is provided to the end user. SaaS is used when applications and services are made available to the user over the Internet. The end user does not have to worry about how the software to be developed. IaaS is a model where an organization is provided with virtual resources like, CPU, servers, storage, memory, operating systems to develop their application. Cloud computing is a powerful technology providing these models and many features like flexibility, lower cost, scalability and easy implementation of software.

But in-spite of these services, there are some drawbacks of Cloud computing. Some of the drawbacks are, the client and Cloud layer can be physically at a greater distance, which can increase the delay in transmission, there can be a shortage of the resources for the execution of the tasks, many resources could remain idle even though tasks need to be processed immediately, etc. To resolve these problems, virtualized Fog computing technology is used[1]. This is a layer between the end users and the Cloud data centers within the Internet. Main areas where Fog computing can be of help is to execute the applications that need low levels of latency, real time response depending on the location of the data generator. This layer can have many virtual servers to process the incoming requests.

According to Agarwal, Yadav and Yadav, “Resource allocation is the systematic approach of allocating available resources to the needed Cloud clients over the Internet” [2]. The time and order of the resources being assigned is crucial to get the maximum benefits of using a virtual server as highest throughput of the system can be obtained and the clients will not be charged exorbitantly. The availability of resources should be such that the high priority tasks are not kept waiting till the very end of the task queue. This could lead to suboptimal use of the virtual servers and possible loss of business. Hence, assigning resources in a prioritized fashion to gain maximum profit is of great importance and a good area of research.

This paper proposes a system where Fog layer is used in-between client and Cloud layer to cater to the applications which have low tolerance for latency. A task is said to have higher priority when the deadline of the task to be processed is closest than any other task. The higher priority tasks are processed first in the Fog layer. If all the data centers in the Fog layer are busy, then the high priority task is propagated to the Cloud layer for timely execution. The data centers in the Fog layer in a region can communicate with each other to check availability of servers and for load balancing. The paper is organized in the following order, section 2 focuses on related work to the research topic. Section 3 describes the proposed architecture and the design of the system. In Section 4 the algorithm of the system is discussed. Section 5 describes the simulation and the setup of the system. Section 6 will discuss in detail the results from the simulation and the comparison. And lastly, the paper concludes with summary in section 7.

2. Related Work

Here we will discuss the research done by other authors related to resource allocation. The various work done by them are based on resource allocation in the Cloud computing environment. But in this paper, we propose a system which has a Fog layer between the Cloud and the end user. Fog layer increases the efficiency of the system in terms of cost, bandwidth usage, energy consumption and the overall response time.

Virtualization is a means of splitting a server into multiple virtual machines. These machines can be used for processing massive operations which are sent to the Cloud. Each machine is provided with a processing unit and an operating system for the execution of the various requests. As these machines can be virtually multiplied, Cloud is immensely elastic in nature. This feature is used in Cloud to optimize resource allocation and the efficiency of the system. This can also decrease the response time of a request greatly as the waiting time is reduced. Fog layer is one such virtualization between the client and the Cloud. The paper by Agarwal, Yadav and Yadav simulates the model where a Fog layer is placed between the client and the Cloud layer [2]. The Fog layer functions like a Cloud layer. A part of this model has been used in our paper as well to reduce the response time of results after a request has been processed.

Resource allocation is of great importance in Cloud computing. To efficiently use Cloud the resources which are assigned to process a request have to be allocated in a particular order to reduce wait time. The response time of a request is crucial for many applications like real time traffic update, artificial intelligence, disaster updates, etc. To provide users with real time results, efficient ordering of resource allocation is very important. Dakshayini & Guruprasad have introduced a priority algorithm which orders the resources in the Cloud layer depending on the importance of the request. In our paper, we have combined the Fog layer [2] and the priority algorithm [3] to use virtualization with ordering of resources to efficiently obtain the results in reduced response time.

Cloud computing systems are used for large scale data processing. Massive files are transferred to the Cloud and processing of the files takes place by dividing the file into smaller blocks. These blocks are further divided into small tasks and are allocated to the servers for further computation. According to Ingole, Chavan and Pawde, the actual cost of Cloud resources cannot be measured as all the tasks differ from each other. Hence, the authors proposed an optimized algorithm for task scheduling based on Activity Based Costing. Tasks are categorized based on the resources on the server. The groups are namely, available and partial. Tasks in available group can be processed on a single data center. In this group, a task can be dependent or independent on other tasks. On the other hand, tasks in partial group cannot be fully processed on a single data center and they need information from other data centers. All the tasks which need data from a particular data center are grouped together. Hence, cat1 will be a list of tasks which need data from data_center1, cat2 will be a list of tasks which need data from data_center2 and so on. When tasks are of similar nature, the grouping of such manner can help to determine the cost of the overall execution more accurately [4]. The proposed priority scheduling algorithm in Fog splits the execution of the tasks between Cloud and Fog layer. It prioritizes the tasks depending on the deadline of the task. This makes it more efficient and cheaper.

Large amounts of data is being generated from smart applications and devices like, real-time location sensitive games and devices which can be worn. Such applications require low-latency computing as they are time-sensitive. Fog computing provides with a layer which provides with resources and computation to reduce the latency between the Internet of Things (IoT) devices and computing infrastructure. Zenith, a model used for allocating computing resources in edge computing is proposed by Xu, Palanisamy, Ludwig and Wang. The proposed system uses a layered architecture. The bottom layer has the IoT devices, the field layer has the computing services. A service management handles activities like, placing containers to meet latency requirements, amount of tasks in the workload to be scheduled to a container and increasing the number of containers to support the workload[5]. The Fog layer proposed in the priority scheduling algorithm decreases the latency of communication between the IoT devices and the servers. This latency decrease also decreases the execution time and the cost of the services greatly.

There are many task scheduling algorithms in Cloud computing and paper written by Singh, Paul and Kumar gives a review of these algorithms. Task scheduling is assigning of resources to the tasks in such a way that the given tasks are processed in the best way and the desired results are obtained by the user. There are two types of scheduling, static and dynamic. In static scheduling the information of the entire system and the resource mapping is known before the execution of the tasks. In dynamic scheduling, the assignment of the resources is dependent on the current state of the system and the computer machines including the submitted tasks in order to make a scheduling decision[6]. Some of the existing task scheduling algorithms covered by the article are, Deadline and Budget Distribution based Cost-Time Optimization Algorithm, Improved cost-based algorithm for task scheduling in Cloud Computing Environment, PSO-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments, etc.

Load balancing is a concept where tasks are divided amongst resources like computers, hard drives and network. Paper written by Verma, Bhardwaj & Yadav explains in detail one such algorithm for load balancing between the layers of client, Fog and Cloud. It tries to solve the issues related to deadlines, execution time and resource allocation. The model of the system first balances the load between the client layer and the Fog layer. In the second phase, if there are no resources available in the Fog layer, the request is propagated to the Cloud layer. The Fog layer is assigned a threshold to control the number of tasks that can be executed in this layer. Once this threshold is surpassed, the task requests are forwarded to the Cloud layer [7].

Resource provisioning based on the satisfaction of the customers can be fulfilled by considering the Service Level Agreements (SLA). By using SLAs and priority task scheduling, the optimal usage of the resources can be obtained. SLA parameters include, memory utilization, network bandwidth and CPU cycles. The article written by Pawar and Wagh describes a priority task scheduling in a dynamic way. The requests from the client are sent to a scheduler. The scheduler divides the request into smaller tasks and checks for the available resources. The resources are assigned to the task depending on the priority of the task and the SLA terms. A task's priority depends on its deadline. The task which has the least deadline is assigned a higher priority as it needs to be executed earlier than a task which has a later deadline. Assigning resources depending on the SLA terms simply means that the machine which will be assigned should have the minimum capability to execute the task at hand [8].

Two scheduling and resource allocation limitations are discussed in the article written by Elghoneimy, Bouhali and Alnuweiri. First limitation is Hadoop MapReduce and its schedulers and second is scheduling of virtual machines to resources in Cloud. MapReduce is the distributed programming model to execute huge number of tasks and data at a time. The available schedulers are, fair scheduler and capacity scheduler. Resource allocation has been one of the top research topics to study as it is the main area in Cloud which requires improvement. This paper proposes an algorithm where MapReduce is used to analyze and check the allocation of virtual machines [9].

To tackle the resource allocation limitation, an advanced version of FCFS has been put forth in article written by Marphatia, Muhnot, Sachdeva, Shukla and Kurup. This algorithm has a client-server model with a resource allocator which manages the allocation of the resources to the client. In this optimized FCFS, the resources are allocated in parts or the request is pushed into a wait queue. The next incoming request is processed if the current task cannot be processed at that moment. The user can login through a GUI and is billed as per the "pay-per use" model. This algorithm improves the performance and the memory usage of server and client [10]. In the priority scheduling in Fog, Cloud Analyst simulator is used to get the results of Fog & Cloud computing environments. The simulator provides with a wide range of parameters to modify and emulate. There are 6 regions in the simulator and each can have as many clients and servers as possible.

There are a number of service broker policies defined for Cloud environment. Some of the policies which are defined in research paper written by Wickremasinghe are, closest data center policy, optimal response time policy and dynamic reconfiguration with load balancing policy. The service broker policy decides the data center to which the request is forwarded to from the user. In the closest data center policy, the request is forwarded to the data center which is geographically closer to the user base. In optimal response time policy, the request is forwarded to the data center which can process the request in the most efficient time. In the dynamic reconfiguration policy, the load of the tasks is balanced by the data center by creating or destroying the virtual machines dynamically [11].

The paper by Bousselmi, Brahmi and Gammoudi focuses on scheduling of the tasks by considering data placement dependencies observed by the Cloud users to process data intensive tasks. The scheduling factors are: Resource properties: CPU, Memory, Network Capacities, Execution time, Data transfer time, Quality of storage resources: Capacity, Availability, Cost. The algorithm for scheduling is based on

Parallel Cat Swarm Optimization (PCSO). The scheduling is divided into two parts, one selects a storage which optimizes the storage of the workflow to the best level and the second selects computing resources which optimizes the quality of the Workflow [12].

3. FOG COMPUTING ARCHITECTURE

To gain maximum economic profit, proper resource allocation is necessary. Resource allocation improves the overall efficiency of the system with the customer satisfaction. As mentioned before, server virtualization is the core part of resource allocation and it improves the overall response time and the cost of the system. The architecture is based on Cloud computing environment with Fog computing technology as can be seen from Fig.1. It is good to note here that Fog layer is not used as replacement of the Cloud environment. But instead Fog layer helps in reducing the draw backs of the Cloud layer, by providing with low latency and geographical distribution. After analyzing the different existing algorithms for resource allocation, we have come up with the proposed framework of the solution.

3.1. DESIGN MODEL

This model is designed in a Cloud-Fog environment. The layers used in the model are, client layer, Fog layer and Cloud layer. First, we implement the algorithm in client and Fog layer to fulfill the requirements of resources for clients. If no resource is available in the Fog layer then move the request to the Cloud layer [2]. The requests in the Fog layer are assigned to the resources depending on the priority given to the request. There are many data centers in the Fog layer as well as the Cloud layer. Each node in the Fog layer has a Fog server (FS).

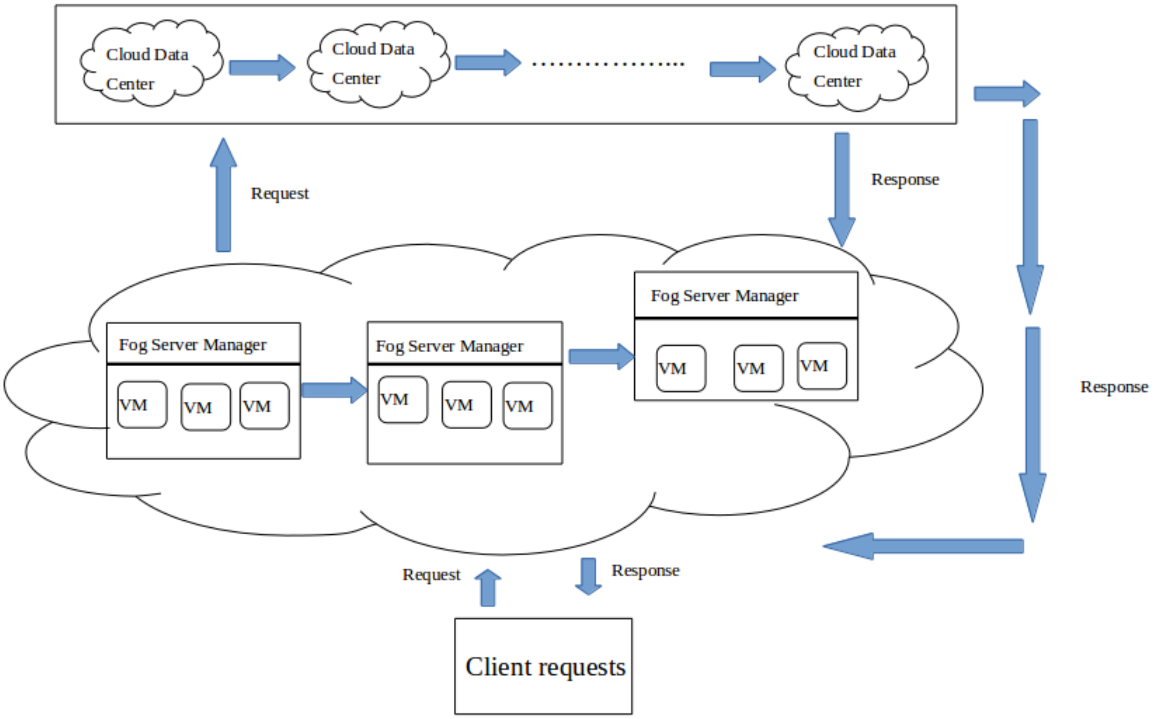
The model may be described as follows, referring to Figure 1:

1. On the top is the Cloud layer, consisting of Cloud data centers.

2. In the middle is the Fog layer. It has a number of Fog node, or Fog Server (FS), which consists of micro datacenters and VMs. Within each FS is also a Fog Server Manager (FSM), which manages the resource within the FS, including processors and VMs.
3. In the bottom is the client layer, where each client sends requests to its nearby FS.
4. Upon receiving a client request, the FSM executes the following step (detailed algorithm presented in Section 4):
    1. If the request cannot be served by its deadline, the request is rejected.
    2. Otherwise the FSM determines the priority queue for the request according to its priority level, its deadline, and the available resources (detailed priority-assignment algorithm presented in Section 3.2).
5. The FSM serves the requests in the priority queues in the order of the priority level. For each request:
    1. If all the resources required by the request are available on the assigned FS, then the request is processed and the result is sent back to the client.
    2. Else if the resources required may be satisfied by the Fog layer, then the requested task (which may be divided into sub tasks) is sent to one or more of the remaining FS in the Fog layer.
    3. Else (no sufficient resource available in the Fog layer) the request is sent to the Cloud layer for processing.

Fig. 1. Architecture of the resource allocation in fog and cloud layer

3.2. QUEUING SYSTEM MODEL



The many client requests which are received may have various deadlines which need to be followed so that the task is completed in time. Based on this requirement, we propose a priority scheduling of the tasks in the fog layer. According to the table in [3], following table is defined.

TABLE I. PARAMETERS CONSIDERED FOR THE QUEUING SYSTEM MODEL

Parameter	**Definition**
Q_H, Q_M, Q_L	High, Medium, Low Priority Queues
T_1, T_2	Threshold for deadline at levels 1 and 2
req_i	The i^{th} user request
λ, λ_e	Mean and Effective request arrival rates
μ_i	Total service time of req_i
$delay_i^T$	Maximum allowed (tolerated) delay of req_i
$ST_{est,i}$	Estimated service time required by req_i
ST_{est}^T	Estimated service time required by all tasks

W_i^Q	Total time user req$_i$ spends in the queue
C	Total number of servers

Assumptions of the priority scheduling is presented below,

1. There are three priority levels, or termed as subscription catalogues (SB $_{CAT}$): 1 = High, 2 = Medium, and 3 = Low.
2. The service time of each of the request from the user i (req$_i$), is calculated as follows:

   $delay_i^T = (DL_i^T - C_i^T)$ ------------- (1)

   where

   $delay_i^T =$ Maximum tolerable time of req$_i$

   C_i^T = Current time

   $DL_i^T =$ Deadline given by req$_i$

   The total service time, ST_{est}^T , required by all the tasks in the 3 priority queues is then:

   $ST_{est}^T = \sum_{i=1}^{Y} ST_{est,i}$ ------------- (2)

   where,

   $Y = |Q_H| + |Q_M| + |Q_L|$

Depending on the delay and the priority level of the request, it will be placed in one of the 3 queues. Total time req$_i$ spends in the fog layer is,

$W_i = W_i^Q + \mu_i$ ------------- (3)

where,

$W_i^Q =$ is the total time req$_i$ spends in the queue

$\mu_i =$ is the total service time taken by the req$_i$

The maximum delay, delay$_i^T$, a request is allowed is specified in SLA. To meet this QoS requirement for the req$_i$,

$W_i < delay_i^T$ ------------- (4)

4. Proposed Algorithm

CS_i : Request i processed by Cloud server.

FS_i : Request i processed by Fog server.

FSM: each FS has a FSM

Req $_i$: Request from user i is processed

SB $_{CAT}$: Subscription category

delay $^T{}_i$: Allowed delay for the request i

T_i : Threshold value for each request i_f .

T: Total task

t $_1$, t $_2$, t $_3$ .....: Sub task of total task.

Prioritized Task Scheduling Algorithm

Step A. Assign Each Request in FS and in Priority Queue

```
For each $req_i$
1. $req_i$ is sent to nearest FS.
2. FS calculates delay $_i^T$ using equation (1)
3. /* Check deadline */
   if ( (($ST_{est}^T$ /C ) +  $ST_{est,i}$) > delay $_i^T$ ),
        Reject $req_i$
4. else /* First attempt to serve in fog layer */
.    /* Place in priority queue using Priority Assignment Module*/
        ($Pr_i$ ) = Priority ($delay_i^T$ , $ST_{est}^T$)
            if ($Pr_i$ == H)
                Place $req_i$ in $Q_H$
            else if ($Pr_i$ == M)
                Place $req_i$ in $Q_M$
            else if ($Pr_i$ == L)
                Place $req_i$ in $Q_L$
```

Step B. Process Request in FS or in Cloud

```
/*Requests served according to priority and $SUB_{CAT}$ */
1.        for x = H, M, and L,
2.            while ($Q_x \neq$ NULL)
3.                for each request in $Q_x$
4.                    if sufficient resources in current FS /* Eq (4) holds */
```

```
                    Serve the request in the current FS
5.             else if sufficient resources in fog layer
                    Allocate resources from the remaining FS, following Step A,
          divide              task to sub-tasks if necessary.
6.             else /* no sufficient resource in fog cluster
                    Send the request to the cloud.
```

<u>Priority assignment Algorithm</u>

```
Priority(delay $^{T}{}_{i}$ , $ST_{est}$, $SB_{CAT}$)
1. /* maximum allowed delay no greater than estimated service time */
      if delay $_i^T$ == $ST_{est,i}$
            Return (H)
2. /* maximum allowed delay between threshold $T_1$ and $T_2$ */
   else if $T_1$ < delay $_i^T$ <= $T_2$
        if $SB_{CAT}$ ==1
            Return (H)
      else if $SB_{CAT}$ ==2
            Return (M)
      else if $SB_{CAT}$ ==3
            Return (L)
3. /* maximum allowed delay greater than threshold $T_2$ */
   else /* delay $_i^T$ > $T_2$ */
      if $SB_{CAT}$ ==1
            if $Q_H$ is not full
                Return (H)
            else
                Return (M)
      else if $SB_{CAT}$ ==2
            If $Q_M$ is not full
                Return (M)
            else
                Return (L)
      else if $SB_{CAT}$ ==3
            Return (L)
```

The variables T_1 and T_2 in the above algorithm signify different periods of time. These variables are used to order the requests in the system according to their deadline. While deciding in which queue a request should be added to, as T_1 is less than T_2, all the requests which have least deadline will be added to the high priority queue as shown in point 2. And all the requests which have a later deadline, will be added to the other queues, so that all the high priority requests are processed first. The use of these variables is important so that the high priority queue is not filled with the requests which could be processed later on.

5. SIMULATION

CloudAnalyst simulator is chosen to execute and analyze the proposed algorithm [11]. This simulator is built on top of CloudSim tool. CloudAnalyst simulates a geographically distributed environment which is used to evaluate the performance of the computing servers when various algorithms are used. This simulator is built for large-scale Cloud applications to study the results of such applications by modifying the many configurations. CloudAnalyst provides the user with a graphical representation of the Cloud environment on a world map. In this simulator, one can select the location of the user bases and the servers, the algorithm which will determine the server to which the task will be forwarded to. The simulation results include a the response time of a user base and server, and the cost associated with it.

The various parameters are set to evaluate the proposed algorithm with the existing algorithms in the simulator. The simulator periodically generates bursts of request from the client devices to create traffic. Parameters like peak hours during which requests will be sent, average number of peak users during a particular moment and off peak average users can be specified using the GUI. The number of users online at a particular time is calculated using these parameters, namely, peakHours, peakAvgUsers and offPeakAvgUsers and also by using a Poisson distribution to randomly vary the number in a realistic manner. For our simulation, the arrivale rate during peak hours is 60,000 requests/hour and during non-peak hours the arrival rate is 6000 requests/hour. For the execution of proposed algorithm, two phases are used to simulate the three layers, client layer, Fog layer and Cloud layer. Duration of the simulation is set to 60 minutes. Every data center has 5 VMs with specifications, Linux Operating system, Xen VMM, cost of VM $/Hr is 0.1 and data transfer cost $/Gb is set to 0.1. The first phase is set for the communication between the client layer and the Fog layer as can be seen in Fig. 2. There are 3 user bases - UB1, UB2, UB3, in the client layer set in the regions 4, 2, 1 respectively. There are 3 data centers – DC1, DC2, DC3, in the Fog layer set in the regions 2, 1, 4 respectively.

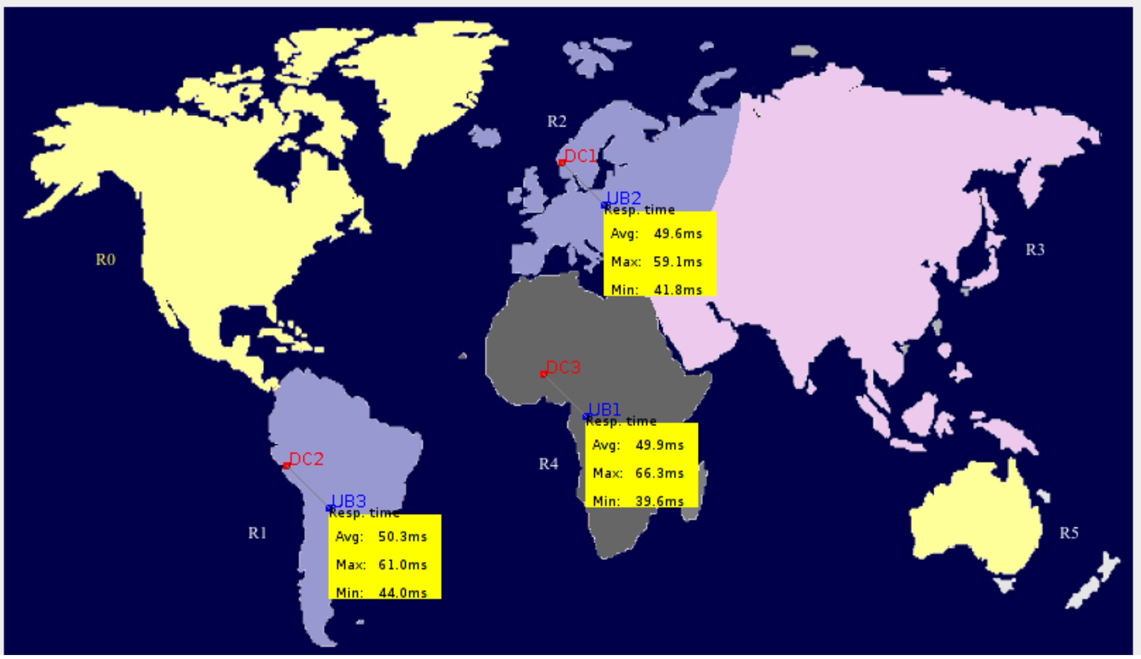


Fig. 2. First phase of the proposed algorithm

For the second phase of the proposed algorithm, we set the parameters for the communication between the Fog layer and the Cloud layer. But here, the Fog layer becomes the client and the Cloud layer becomes the data center to provide the computation. In this phase, we set only one Fog data center as the user base (UB1) as only one Fog data center is selected for each user base in the previous phase. We set one data center (DC1) in the Cloud layer in this phase as the Fog will choose the nearest data center when there are no more processes in the Fog layer. The UB1 is set in the region 1 and the DC1 is set in the region 0. For the simulation of the existing algorithms 3 user bases in the client layer and 3 data centers in the Cloud layer are considered. The user bases, UB1, UB2, UB3 are set in the region, 4, 2, 1 respectively. The data centers, DC1, DC2, DC3 are set in the region, 0, 3, 5 respectively.

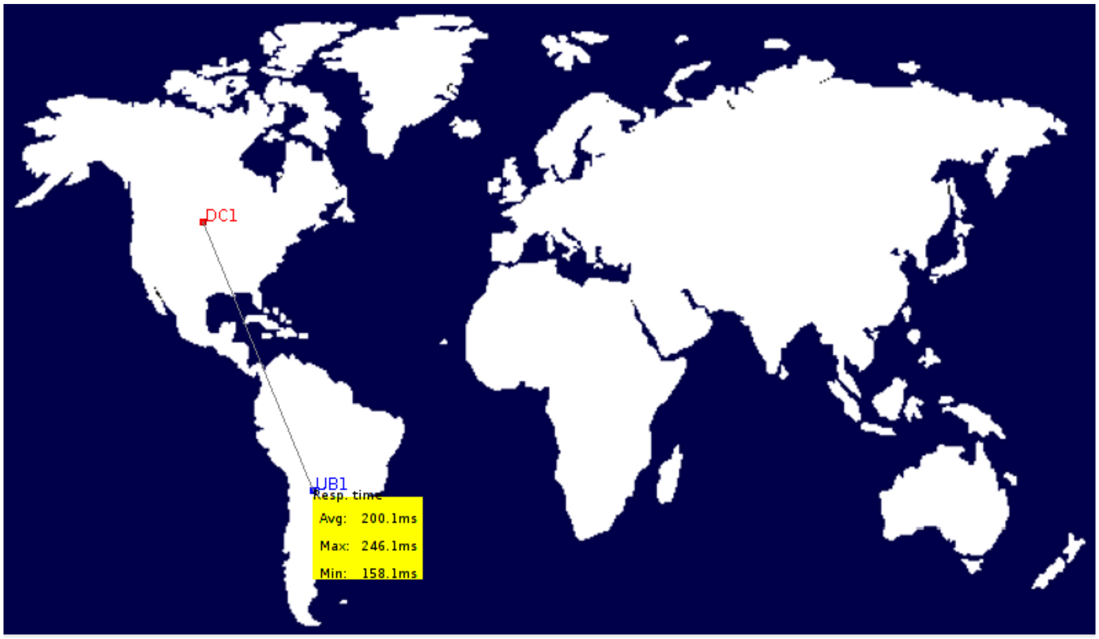
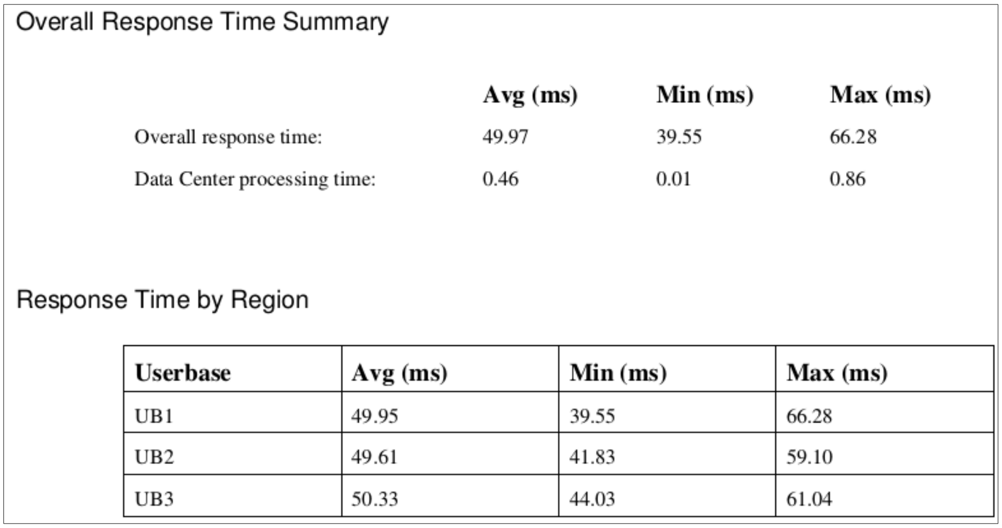


Fig. 3. Second phase of the proposed algorithm



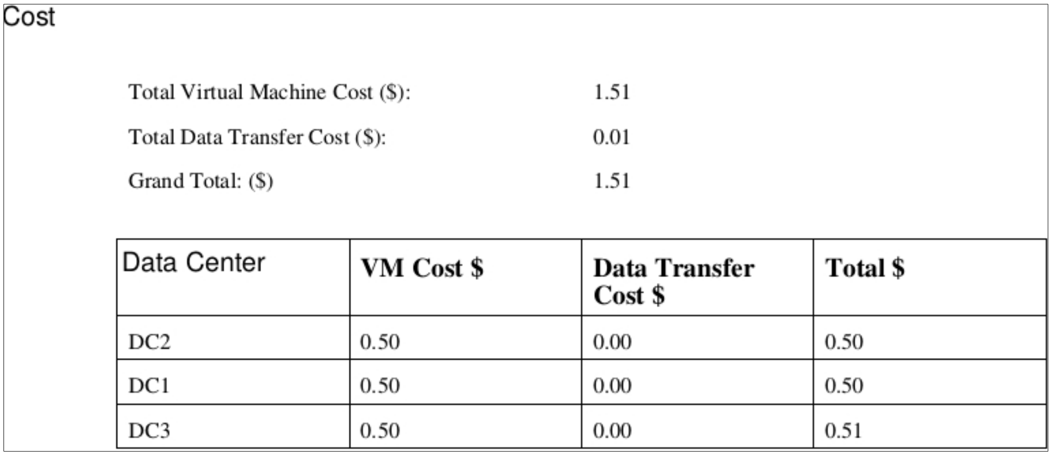
Overall Response Time Summary

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	49.97	39.55	66.28
Data Center processing time:	0.46	0.01	0.86

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	49.95	39.55	66.28
UB2	49.61	41.83	59.10
UB3	50.33	44.03	61.04

Fig. 4. Response Time of the First phase of the proposed algorithm



Cost

Total Virtual Machine Cost ($):	1.51
Total Data Transfer Cost ($):	0.01
Grand Total: ($)	1.51

Data Center	VM Cost $	Data Transfer Cost $	Total $
DC2	0.50	0.00	0.50
DC1	0.50	0.00	0.50
DC3	0.50	0.00	0.51

Fig. 5. Cost of the First phase of the proposed algorithm

6. SIMULATION OUTPUT

The client, Fog servers and Cloud servers are geographically placed throughout the globe and they are spread out over 6 regions. The proposed architecture is implemented in two phases as mentioned above and the output can be seen in the figures below. For the propose algorithm, the closest data center policy has been selected on the screen. For the existing algorithms, we have used Optimize Response Time policy and Reconfigure Dynamically with Load Balancing policy.

6.1. Cloud-only layer results

The results include overall response time, response time of data centers and the cost associated with the processing of the tasks. The below table represents the result for the communication between the client and the Cloud layer only. The results are a comparison between two version of the same algorithm, with and without using priority scheduling.

Table 2. simulation results for Cloud layer

Algorithm	ORT (Optimize Response Time)	ORT with Priority Scheduling	Performance Difference

Overall avg. response time (ms)	333.83	345.56	3.5%
Total cost ($)	2.59	1.41	-45%
Algorithm	RD (Reconfigure Dynamically)	RD with priority	Performance Difference
Overall avg. response time (ms)	335.40	376.28	12.1%
Total cost ($)	4.27	3.88	-9.13%

The table shows priority scheduling combined with two existing algorithms in the simulator. From the results it can be observed that the priority scheduled version of both the existing algorithms cost lesser than their original versions.

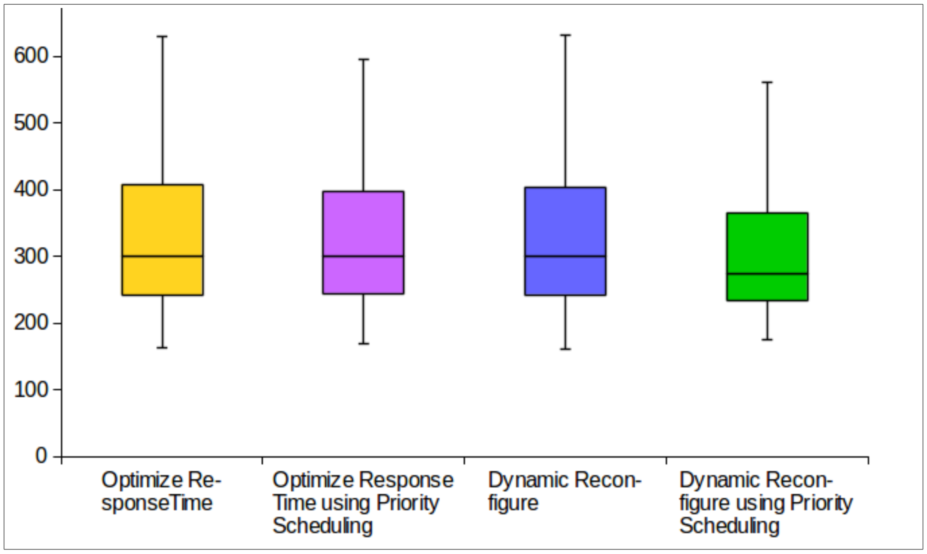


Fig. 6. Overall Response Time(ms) comparison in Cloud

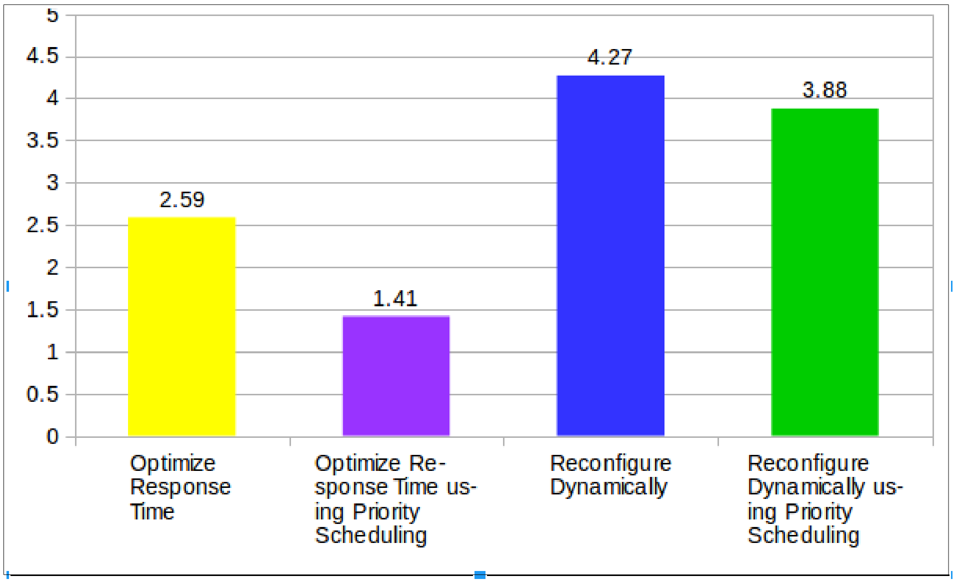


Fig. 7. Total Cost($) comparison in Cloud

6.2. Fog layer results for single user

TABLE 3. SIMULATION RESULTS FOR FOG AND CLOUD LAYERS

Client – Fog Layers			
Algorithm	ERA (Efficient Resource Allocation) [2]	ERA with Priority (Proposed)	**Performance Difference**
Overall avg. response time (ms)	50.24	49.97	**-0.53%**
Total cost ($)	1.70	1.51	**-11.12%**
Fog – Cloud Layers (no priority in cloud layer)			
Algorithm	ERA (Efficient Resource Allocation) [2]	ERA with Priority (Proposed)	Performance Difference
Overall avg. response time (ms)	200.07	200.07	0%
Total cost ($)	0.57	0.57	0%

3-Layer: Client – Fog – Cloud			
Algorithm	ERA (Efficient Resource Allocation) [2]	ERA with Priority (Proposed)	**Performance Difference**
Overall avg. response time (ms)	250.31	250.04	**-0.1%**
Total cost ($)	2.27	2.08	**-8.37%**

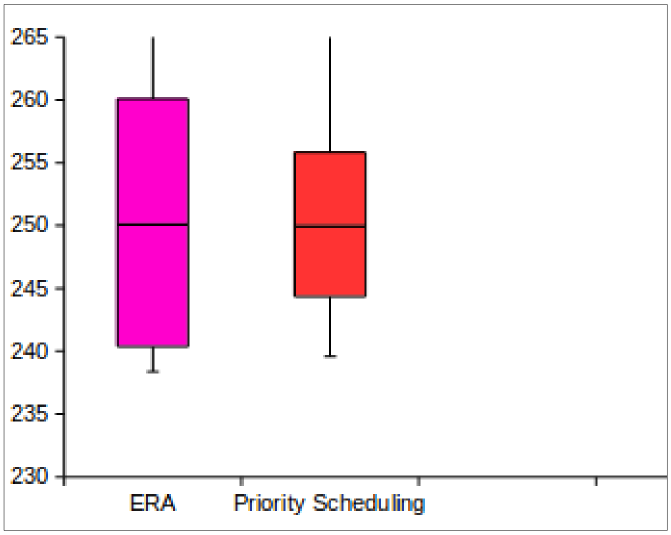


Fig. 8. Overall Response Time(ms) comparison of Fog layer

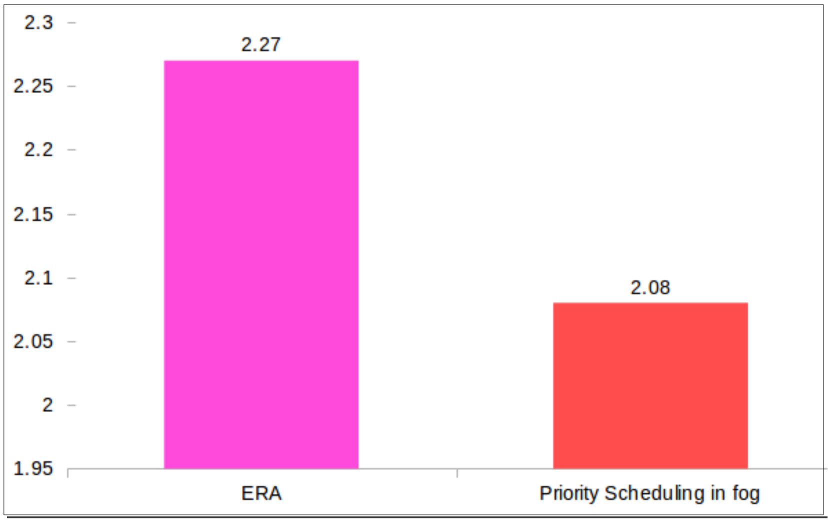


Fig. 9. Cost($) comparison of Fog layer

The simulation for this result is set for single user and multiple servers in one region only. The comparison of the proposed algorithm is done with the Efficient Resource Allocation algorithm. As can be seen from Table 3, the proposed algorithm performs better in terms of the response time as well as the cost associated with the simulation of the system when compared with the efficient resource allocation algorithm. It can be observed that the 'Optimize response time using priority scheduling' costs less than the 'Priority scheduling in Fog', but it trades off the execution time for the cost. In a practical setting, the system user can choose the algorithm according to the needs of the customer. The choice can be between quicker execution or lower cost.

6.3. Fog layer results for multiple entities in a region

In real world, there are multiple Fog centers and users in a particular region, hence multiple users and Fog centers are considered in a particular region. The load of all the requests is uniformly distributed among the located Fog centers in the region of the user base. There are 8 user bases and 4 Fog centers in region 1. In region 2 and 4, there is one of user base and Fog center each.

The Fig. 8 depicts a graph which is a comparison between the algorithms, priority scheduling in Fog and the existing algorithms, efficient resource allocation , optimize response time and reconfigure dynamically with load balance for the overall response time. As can be seen even for multiple user base and Fog data centers simulation, the proposed algorithm, priority scheduling performs better than the existing algorithms in CloudAnalyst. The execution time of priority scheduling algorithm and optimize response time is almost similar, but the cost of this execution is lower for the priority scheduling algorithm.

TABLE IV. SIMULATION RESULTS FOR MULTIPLE USERS IN SINGLE REGION

2-Layer: Client - Cloud			
Algorithm	ORT	ORT with Priority Scheduling	**Performance Difference**
Overall avg. response time (ms)	265.98	252.12	**-5.5**
Total cost ($)	5.01	4.43	**-13.1%**
Algorithm	RD	RD with priority	
Overall avg. response time (ms)	225.92	253.7	**12.3%**
Total cost ($)	5.14	4.63	**-10.1%**
3-Layer: Client – Fog – Cloud (no priority in cloud layer)			
Algorithm	ERA (Efficient Resource Allocation) [2]	ERA with Priority (Proposed)	Performance Difference
Overall avg. response time (ms)	250.13	250.11	-0.007%
Total cost ($)	4.14	3.53	-14.73%

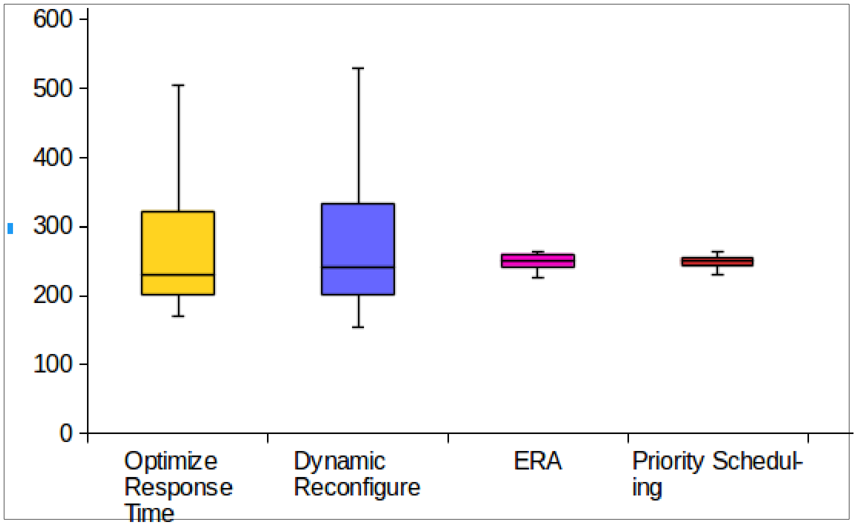


Fig. 9. Overall Response Time(ms) comparison for multiple users

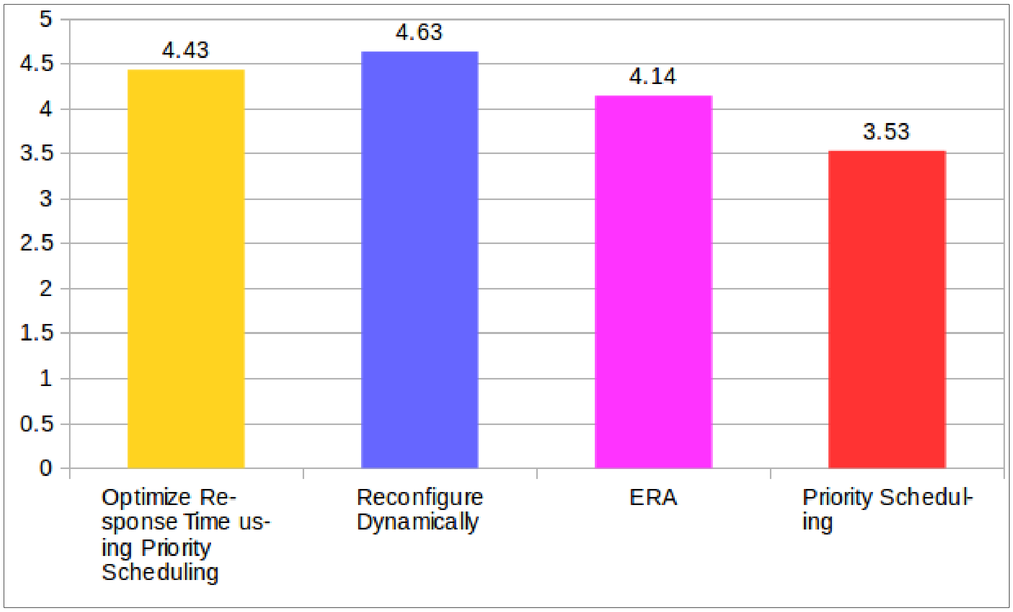


Fig. 10. Total Cost($) comparison for multiple users

In the above figures Optimize Response Time using Priority Scheduling & Reconfigure Dynamically algorithms are Cloud-only algorithms. ERA & Priority Scheduling are Fog algorithms. As can be seen from the results, the Fog algorithms perform better in terms of response time as well as the cost for the simulation.

6.4. Fog layer results for multiple entities in multiple regions

Another simulation which was carried out has multiple clients and multiple data centers in multiple regions. There are 8 clients and 4 Fog nodes in region 1, 4 clients and 2 Fog nodes in region 2 and, 4 clients and 2 Fog nodes in region 4. Fig 11 and 12 show the comparison between the algorithms in this simulation. And for the simulation between client and Cloud layer the setup includes, 5 clients in region 1, 1 data center in region 0, 4 clients in region 2, 1 data center in region 3, 4 clients in region 4, 1 data center in region 5.

TABLE V. SIMULATION RESULTS FOR MULTIPLE USERS IN SINGLE REGION

2-Layer: Client - Cloud			
Algorithm	ORT	ORT with Priority Scheduling	**Performance Difference**
Overall avg. response time (ms)	332.9	326.92	**-1.8%**
Total cost ($)	4.6	3.54	**-23.1%**
Algorithm	RD	RD with priority	
Overall avg. response time (ms)	293.8	326.46	**11.1%**
Total cost ($)	5.02	4.6	**-8.5%**
3-Layer: Client – Fog – Cloud (no priority in cloud layer)			
Algorithm	ERA (Efficient Resource Allocation) [2]	ERA with Priority (Proposed)	Performance Difference
Overall avg. response time (ms)	250.14	250.11	-0.008%
Total cost ($)	4.53	3.55	-21.63%

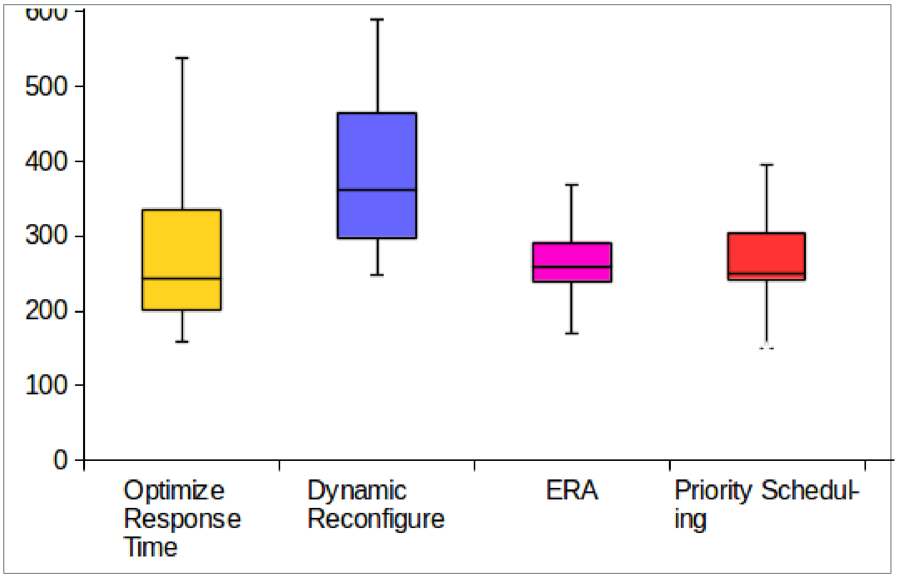


Fig. 11. Overall Response Time(ms) comparison for multiple users in multiple regions

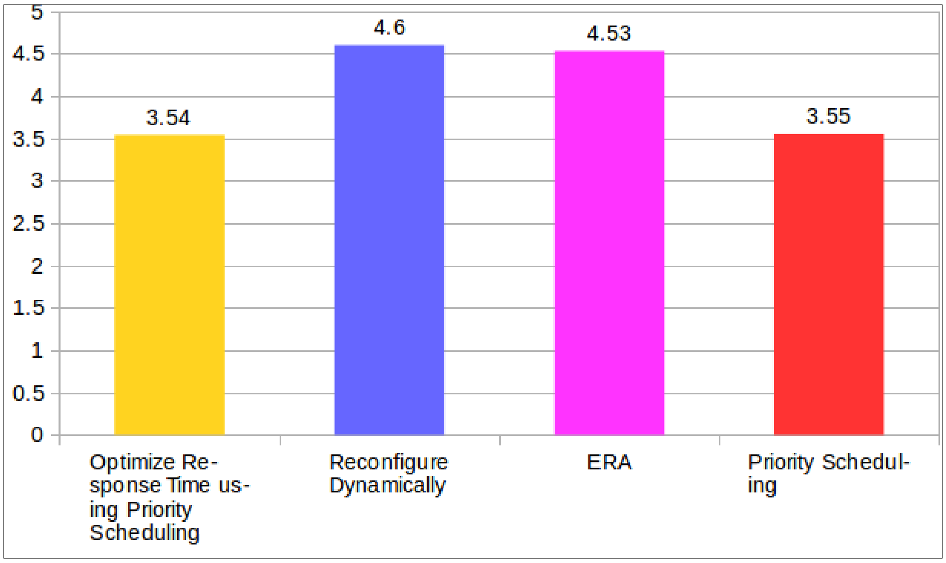


Fig. 12. Total Cost($) comparison for multiple users in multiple regions

As can be seen from the charts, the overall response time of Priority Scheduling algorithm (Fog) is almost as same as ERA(Fog) algorithm. Yet the cost of the proposed Priority Scheduling algorithm is significantly lower in comparison with ERA. The performance in cost reduction is even more substantial in this more complex fog computing setting. Overall, the proposed algorithm achieves the lowest response time and the lowest cost.

7. CONCLUSION

Many IoT devices produce a lot of data and requests need to be handled with as less delay as possible. Fog layer takes care of this as it is placed between client and the Cloud layer. Fog layer aids in processing the tasks which are deployed by the client by providing with the responses in less time. It brings the Cloud environment physically closer to the client. The requests can have different levels of priorities and the higher priority requests should be processed before other lower priority tasks. This paper discussed in detail the priority task scheduling in the Fog layer. The algorithm was implemented using CloudAnalyst, a simulator for Cloud environment. The algorithm was tested against existing algorithms present in the simulator and ERA. The simulation carried out was for the various types of environments, single client & single data center, multiple clients & data centers in single region, multiple clients and data centers in multiple regions. From the results, it was found out that some algorithms perform better for execution time ans some perform better cost-wise. Depending on the requirements of the system, one can choose priority scheduling for quicker execution or one can choose ERA algorithm for lesser cost.

As an enhancement to the proposed system, we can use a priority scheduling algorithm which is more suitable for the Cloud layer and analyze the response of the system. As the thresholds of the algorithm decide the tasks which are rejected, we can dynamically change T_1 & T_2 in a future algorithm.

REFERENCES

[1] Kamyab Khajehei, *"Role of virtualization in Cloud computing", International Journal of Advance Research in Computer Science and Management Studies,* Volume 2, Issue 4, April 2014.

[2] S. Agarwal, S. Yadav & A.Yadav, "*An Efficient Architecture and Algorithm for Resource Provisioning in Fog Computing"*, in MCEP, 2016, doi: 10.5815/ijieeb.2016.01.06

[3] Dr. M. Dakshayini & Dr. H. S. Guruprasad, "An Optimal Model for Priority based Service Scheduling Policy for Cloud Computing Environment", International Journal of Computer Applications (0975 – 8887), ISE, Bangalore, India, Volume 32– No.9, October 2011

[4] A. Ingole, S. Chavan and U. Pawde, "An Optimized Algorithm for Task Scheduling based on Activity based Costing in Cloud Computing", IJCA Proceedings on 2nd National Conference on Information and Communication Technology NCICT(3):34-37, Nov., 2011

[5] J. Xu, B. Palanisamy, H. Ludwig and Q. Wang, "Zenith: Utility-aware Resource Allocation for Edge Computing", IEEE International Conf., 2107, 10.1109/IEEE.EDGE.2017.15

[6] R. Singh, S. Paul, A. Kumar, "Task Scheduling in Cloud Computing: Review", (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (6) , 2014, 7940-7944

[7] M. Verma, N. Bhardwaj & A. Yadav, "*Real Time Efficient Scheduling Algorithm for Load Balancing in Fog Computing Environment*", in MCEP, 2016, doi: 10.5815/ijitcs.2016.04.01

[8] C. Pawar and R.Wagh, "Priority Based Dynamic resource allocation in Cloud Computing", Intelligent Systems and Signal Processing (ISSP), 2013 International Conference, 10 June 2013

[9] E. Elghoneimy, O. Bouhali, H. Alnuweiri, “Resource Allocation and scheduling in Cloud Computing”, Proc. Of the IEEE International Workshop on Computing, Networking and Communications, 2012, pp. 309 – 314.

[10] A. Marphatia, A. Muhnot, T. Sachdeva, E. Shukla, Prof. L. Kurup, “Optimization of FCFS Based Resource Provisioning Algorithm for Cloud Computing”, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 10, Issue 5 (Mar. - Apr. 2013), PP 01-05

[11] B. Wickremasinghe, “Cloud Analyst: A CloudSim Based Tool For Modeling And Analysis Of Large Scale Cloud Computing Environments”, MEDC Project Report 2009

[12] K. Bousselmi, Z. Brahmi and M. Gammoudi, “QoS-Aware Scheduling of Workflows in Cloud Computing Environments”, IEEE, 2016 , 10.1109/AINA.2016.72

[13] K. Khajehei,”Role of virtualization in Cloud computing”, International Journal of Advance Research in Computer Science and Management Studies Volume 2, Issue 4, April 2014.

[14] I. Stojmenovic, S. Wen, “The Fog Computing Paradigm: Scenarios and security issues” Proceedings of the IEEE International Fedrerated Conference on Computer Science and Information Systems, 2014, pp. 1-8.

[15] N. Rammohan and E. Baburaj, “Resource Allocation Techniques in Cloud computing-Research Challenges for Applications”, Proc. Of the IEEE Fourth International Conference on Computational Intelligence and Communication Networks, 2012, pp. 556-560.

[16] L. Luo,Wenjun Wu, D. Di, F. Zhang, Y. yan, Y. Mao,”A Resource scheduling algorithm of Cloud computing based on energy efficient optimization methods”, Proc. Of the IEEE International Green Computing Conference (IGCC), 2012, pp. 1 – 6.

[17] Aditya Marphatia, Aditi Muhnot, Tanveer Sachdeva, Esha Shukla, Prof. Lakshmi Kurup,” Optimization of FCFS Based Resource Provisioning Algorithm for Cloud Computing” , IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p- ISSN: 2278- 8727Volume 10, Issue 5 (Mar. - Apr. 2013), PP 01-05.

[18] Swati Agarwal, Shashank Yadav, Arun Kumar Yadav,”An architecture for elastic resource allocation in Fog computing”, International Journal of Computer Science and Communication, Vol. 6 Number 2 April - Sep 2015 pp. 201-207.

[19] R. Buyya, R. Ranjan, and R. N. Calheiros, “Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities,” Proc. of the 7th High Performance Computing and Simulation Conference (HPCS 09), IEEE Computer Society, June 2009.

[20] S. K. Garg, R. Buyya, and H. J. Siegel, “Time and cost trade off management for scheduling parallel applications on utility grids,” Future Generation. Computer System, 26(8):1344–1355, 2010.