Dedicated
To
My parents
&
Teachers

CERTIFICATE

I am pleased to certify that Md. Emon Shaikh, Roll no: 1418050 and Registration no:1365, has performed a project work entitled **Methods of Resource Scheduling Based on Optimized in Fog Computing** under my supervision in the academic year 2014-15 for the fulfillment of partial requirement of B.Sc. degree. So far as I concern that is an original project work that the carried out for B.Sc. degree in the department of information and communication technology, Islamic university, Kushtia-7003, Bangladesh

I strongly declare that this dissertation has not been copied from any other project or submitted to elsewhere prior submission to this department

[Dr. Md. Sipon Miah]

Associate professor

Department of information and communication technology

Islamic university, kushtia-7003, Bangladesh

ABSTRACT

Cloud computing and fog computing are the upcoming Information Technology (IT) computing models. These groundbreaking paradigms are leading IT to a new set of rules that aims to change computing resources delivery and exploitation model, thus creating a novel business market, saving money or time that is exponentially growing and attracting more and more investments from both providers and end users that are looking forward to make profits from these innovative models of computing. In the same context, researchers and investigators are wrestling time in order to develop, test and optimize cloud computing platforms, whereas several studies are ongoing to determine and enhance the essential aspects of these computing models especially compute resources allocation and resource optimization. The processing power scheduling is crucial when it comes to cloud computing and fog computing because of the data growth management data scheduling and delivery design proposed by these new computing models which require faster system and device responses from platforms and applications. Hence originates the importance of developing high efficient scheduling algorithms that are compliant with these computing models platforms and infrastructures requirement.

Keywords: Cloud computing; fog computing; optimize base resource scheduling algorithms; Cloud computing and fog computing simulation; Round Robin; First Come First Served and Priority scheduling.

CONTENTS

List of the figure

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

With the rapid development of technology related to the Internet of Things (IoT) [1], an increasing amount of data is being transmitted through the internet. Cloud computing, as a distributed computing model, can store and process the massive data generated by the IoT and provide terminal users with reliable services [2,3]. However, smart devices consume a large amount of network bandwidth and aggravate the burden of cloud data centers [4,5], such that some delay-sensitive services in the IoT cannot be responded to and processed quickly. In addition, many requirements of edge devices, such as real-timeliness, mobility, location awareness, etc cannot be met. A new computing paradigm called fog computing was proposed by Cisco in 2012, which extends the traditional cloud computing paradigm to the edge of the network [6]. Cloud computing introduces a new Information Technology (IT) service delivery model that aims to reshape the world of computing. In fact, a cloud user needs only a light terminal to access any desirable IT resource. The cloud proposed resources ranges from documents processing, email or storage to complicated data treatment and management system such as enterprise resource planning, database management systems, integrated development environment or application hosting, and all of that is presented as a service and charged in a pay-as-you-use, pay-as-you-go or even by subscription. Recently, we cannot talk about cloud computing without tackling the subject of data growth management, in other terms data scheduling. This newly born computing model is on an everyday development in order to cope with the changes going on the IT world thus adapting to the world cloud orientation. The central focus on these computing models are to raise profits for both providers and users and this is by reducing the costs relative to IT infrastructures, platforms management and scaling up as well as promoting accuracy and auditability.

With the cloud computing outstanding business growth, the investigations on cloud computing resources allocation and scheduling algorithms are becoming a major issue because of the cloud delivery model and whereas countless researches are carried out each year all over the world. Under the same theme, this paper presents an ongoing exploration of Central Processing Unit

(CPU) allocation and task scheduling on the cloud in order to elaborate an optimal design that guarantee data treatment celerity and efficiency while respecting data fundamentals

1.2 OBJECTIVE

- Resource optimization process of fog computing.
- Use different types of scheduling algorithms.
- Response Time comparison for multiple users of fog layer and cloud layer.

Future industrial automation architectures and applications based on Fog Computing, deterministic virtualization and execution, deterministic wired and wireless communication, resource provisioning and resource scheduling, Service-Oriented Architecture (SOA), real-time data analytics and security. The result will be an open architecture built on open source and open standards, e.g., Time-Sensitive Networking (TSN), Time saving, OPC Unified Architecture (UA).

CHAPTER 2
RELATED WORK

2.1 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. We also discuss about 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 responses 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. The fog layer functions like a cloud layer [19]. 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. 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, theactual 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. When tasks are of similar nature, the grouping of such manner can help to determine the cost of the overall execution more accurately [20]. 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 device. 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 [21]. 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. 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 [22]. 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 [23]. 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 [24]. 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) [25]. The scheduling is divided into two parts, one selects a storage which optimizes the storage of the work to the best level and the second selects computing resources which optimizes the quality of the work.

CHAPTER 3
FOG COMPUTING

3.1 FOG COMPUTING

Fog computing refers to a decentralized computing structure, where resources, including the data and applications, get placed in logical locations between the data source and the cloud; it also is known by the terms ‘fogging’ and ‘fog networking’. Fog computing is a paradigm with limited capabilities such as computing, storing and networking services in a distributed manner between different end devices and classic cloud computing. It provides a good solution for IoT applications that are latency-sensitive [9]. The goal of this is to bring basic analytic services to the network edge, improving performance by positioning computing resources closer to where they are needed, thereby reducing the distance that data needs to be transported on the network, improving overall network efficiency and performance. Fog computing can also be deployed for security reasons, as it has the ability to segment bandwidth traffic, and introduce additional firewalls to a network for higher security. Fog computing has its origins as an extension of cloud computing, which is the paradigm to have the data, storage and applications on a distant server, and not hosted locally. With the cloud computing model, the client can purchase the services from a provider, which delivers not only the service, but also the maintenance and upgrades, with the plus that they can be accessed anywhere, and facilitating work by teams.

3.2 FOG COMPUTING ARCHITECTURE

Fog computing is a new computing mode. As a derivative of cloud computing, fog computing can solve the problems of high latency, overloaded center server and overloaded bandwidth of network. There are many kinds of existing fog computing architectures, which lacks flexibility in handling of tasks and interaction of networks. In this paper, based on the advantages and problems of the existing fog computing architectures, we propose a distributed fog computing architecture that supports multiple migrating mode, which include center-edge, edge-edge migrating. The main research contents and contributions of this paper are as follows: (1) Propose a distributed architecture that contains user management node, network management node and service node on center and edge network. (2) Design a scalable distributed computing node model in edge network. (3) Carry out a user registration method based on distributed storage. (4) Design a service migration model based on the load of network, which included the center-edge and edge-edge migrating support. Cloud computing is centralized and composed of powerful servers, which can be extended to the edge of the network using fog computing. Fog computing consists of some weaker and more dispersed servers, which are closer to terminal users. In addition, fog computing can provide flexible resources to completely heterogeneous computing and storage requests in the IoT. Currently, the general fog computing architecture can be regarded as a three-tiered network structure, as shown in Figure 1.

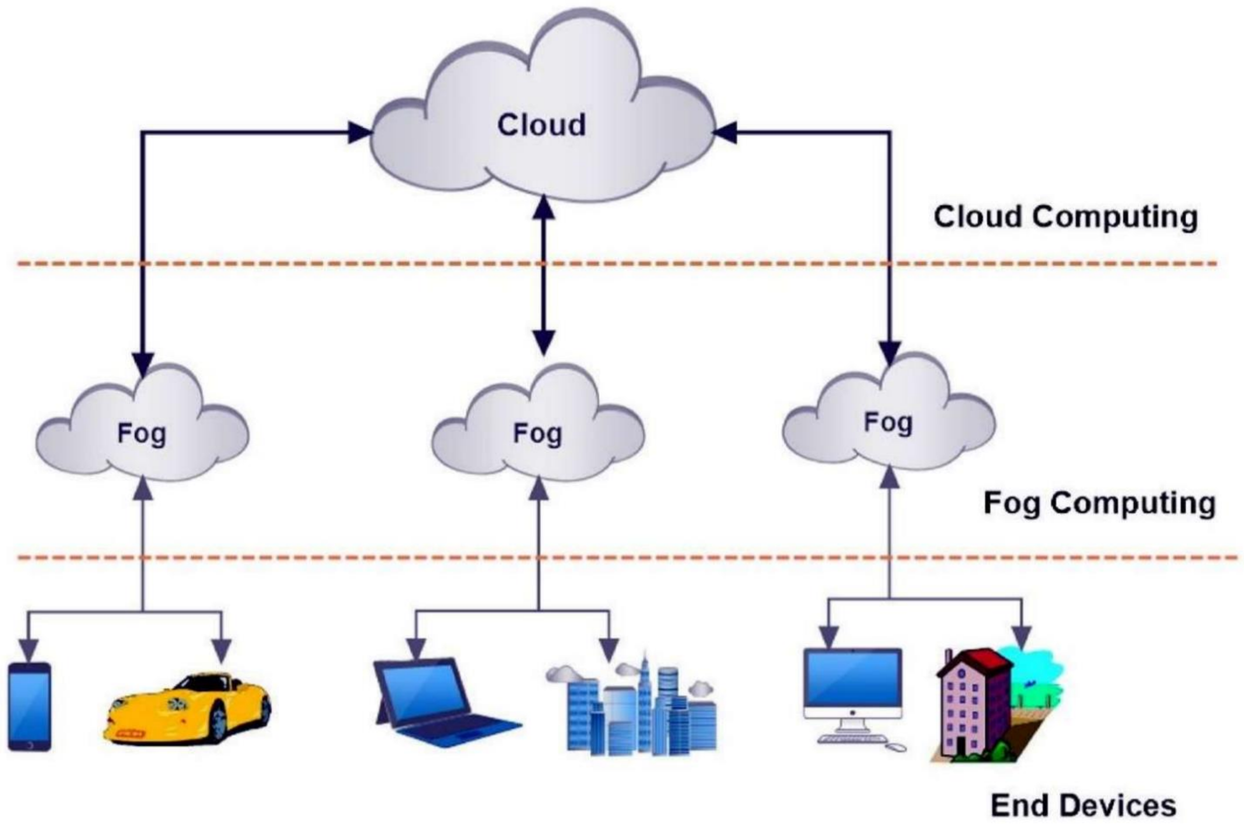

Figure 1. Fog computing architecture.

The IoT device layer includes various types of devices, such as smart phones, tablet computers, smart vehicles, and different smart home devices. The IoT can sense the surrounding environment and collect data continuously through some sensor devices. In addition, this layer communicates with the fog computing layer through 3G, 4G, WiFi, and WiBro technologies, and then uploads the collected sensor data. The fog computing layer consists of many edge servers, including some heterogeneous devices, such as routers, switches, gateways, and access points. The fog computing layer can be seen as Fog Instances (FIs) [10]. Each fog instance contains a Fog Smart Gateway (FSG) [11]. The FSG can implement heterogeneous network conversion and determine the type of data and timings to be processed in the fog. This layer can collect data and perform data preprocessing to filter out redundant or abnormal data and then upload them to the cloud. Moreover, fog computing is suitable for low-latency applications such as video streaming, augmented reality, and online games [12]. The cloud computing layer includes many powerful high-end servers; the cloud data center processes and stores large amounts of data, and enables efficient resource management by deploying virtualization technologies [13]. This level supports the long-term analysis of the uploaded data by the fog computing layer.

Fog computing has a number of advantages. By adding the capability to process data closer to where it is created, fog computing seeks to create a network with lower latency, and with less data to upload, it increases the efficiency at which it can be processed. There is also the benefit that data can still be processed with fog computing in a situation of no bandwidth availability. Fog computing provides an intermediary between these IoT devices and the cloud computing infrastructure that they connect to, as it is able to analyze and process data closer to where it is coming from, filtering what gets uploaded up to the cloud. A downside of cloud computing is that all this computing over the network relies heavily on data transport. While broadband internet access has generally improved over the last decade, there are still challenges with accessibility, peak congestion, lower speeds on mobile 3G and 4G cellular networks, as well as occasions of limited internet availability whether underground, off the grid or on an airplane. This lack of consistent access leads to situations where data is being created at a rate that exceeds how fast the network can move it for analysis. This also leads to concerns over the security of this data created, which is becoming increasingly common as Internet of Things (IoT) devices become more commonplace. Physically, this extra computing power closer to the data creation site in a fog computing configuration gets located at a fog node, which is considered a crucial ingredient in a cloud-fog-thing network. The fog node, which is located in a smart router or gateway device, allows for data to be processed on this smart device, so that only the necessary data gets further transmitted to the cloud, and decreases the bandwidth used.

CHAPTER 4
RESOURCE SCHEDULING PROBLEM IN FOG COMPUTING

4.1 DESCRIPTION OF RESOURCE SCHEDULING PROBLEM IN FOG COMPUTING

In most cases, resource scheduling can be seen as a task scheduling problem that allocates resources to users tasks. If the resource scheduling is performed in the cloud, large scale data transmission may consume a large amount of network bandwidth and add the burden of cloud data centers. In addition, resource scheduling in fog computing depends on the kind of user service request. Service requests include delay tolerance and delay sensitivity. The cloud computing is far away from the end users, which will generate a large transmission delay, but fog computing is close to the end users, which has the advantages of low delay and location awareness. Therefore, the delay sensitive user requests are usually processed in the fog computing layer, and the delay tolerant user requests are scheduled in the cloud computing layer. Therefore, this paper focuses on resource scheduling in fog computing. The specific differences between cloud-based scheduling and fog-based scheduling are shown in Table 1. Due to the diversity of user requests, fog computing will allocate resources according to the user needs. The goal of resource scheduling is to find the best matching resources for users to achieve the optimal scheduling goals, such as reducing the processing delay and improving resource utilization and Quality of Service (QoS).

Table 1**.** Show Differences between cloud-based scheduling and fog-based scheduling.

Cloud-Based Scheduling	**Fog Based Scheduling**
1. Strong computation power. 2**.** Apply to processing delay- tolerant user requests. 3. Consume a lot of bandwidth. 4. A large transmission delay.	1. Low delay and location awareness**.** 2. Apply to processing delay-sensitive user requests. 3. Weaker computation power. 4. Not applicable to processing large-scale requests.

4.2 CPU ALLOCATION & SCHEDULING

The allocation and scheduling of compute resources (CPU) is a major cloud computing factor that will affect massively the most significant characteristics of Quality of Service (QoS) related to applications and platforms response time, thus comes the importance of choosing high-enhanced CPU allocation algorithms that suits the cloud infrastructure and meets the customer requirements. A cloud provider main interest is to increase profits by achieving high levels of users satisfaction, which comes with providing the end user with the best experience. Hence, comes the importance of choosing the finest scheduling algorithms for resources allocation and task scheduling. The key purpose of scheduling algorithms is the appropriate allocation of task or a job to the appropriate resource. Therefore, it goes back always to the period necessary to carry out the execution of a specific task in order to evaluate the quality and performance of the scheduling algorithms [14,15]. The major compute resources scheduling algorithms are the following :

4.3 FIRST COME FIRST SERVED SCHEDULING

One of the basic scheduling algorithms and the foundation of the most used scheduling algorithms. The algorithm has a simple design of processing tasks based on their arrival order. Given n processes with their burst times, the task is to find average waiting time and average turnaround time using FCFS scheduling algorithm. First In, First Out (FIFO), also known as First Come, First Served (FCFS) [18], is the simplest scheduling algorithm. FIFO simply queues processes in the order that they arrive in the ready queue. In this, the process that comes first will be executed first and next process starts only after the previous gets fully executed

4.4 FIRST COME FIRST SERVE ALGORITHM

```
Start
Initialization of process
Input the process
Frist process that come need not wt
Em: wt [0] = 0
Find waiting time for process i->
Wt [i] =bt[i-1] + wt [i-1]
Turnaround time=waiting time +burst time
For all process
Average waiting time = total waiting time /no of processes
average turnaround time= total turnaround time/no of process
similarly those all process.
```

Table 1: shown a FIFO scheduling algorithm for different types of process. Here P1,P2, P3,P4, P5, P6 are the process.

Table 1: FIFO scheduling

Process	Burst time in second	Average time in second
P1	5	3
P2	3	1
P3	7	4
P4	1	2
P5	8	5
P6	2	1

This algorithm is one version of the previous algorithm, which adds a new rule that arranges tasks in an ascending way based on their burst time and then queuing them for execution.

4.5 ROUND ROBIN SCHEDULING

The Round Robin Algorithm [16,17] was designed based on the distribution of CPU time among the scheduled tasks. On the same context, all the tasks get on a queue list whereas each task get a small unit of CPU time (Quantum, usually 10-100 milliseconds). Round Robin scheduling is the preemptive scheduling in which every process get executed in a cyclic way, i.e. in this a particular time slice is allotted to each process which is known as time quantum. Every process, which is present in the queue for processing, CPU is assigned to that process for that time quantum. Now, if the execution of the process gets completed in that time quantum, then the process will get terminate otherwise the process will again go to the ready queue, and the previous process will wait for the turn to complete its execution.

4.6 ROUND ROBIN SCHEDULING ALGORITHM

```
1. Start
2. Declare the array size
3. Read the number of processes to be inserted
4. Read the burst times of the processes
5. Read the Time Quantum
6. if (BT>TQ)
then
TQ-BT
Else
Assign the burst time to time quantum.
7.Calculate the average waiting time and turn around time of the processes.
8.Display the values
9. Stop
```

Table 1: shown a Round Robin scheduling algorithm for different types of process. Here P1, P2, P3, P4, P5, P6 are the process.

Table 2: Round Robin Scheduling

Process	Burst time in second
P1	4
P2	1
P3	2
P4	3
P5	5
P6	8

4.7 PRIORITY SCHEDULING

The Priority Scheduling algorithm assigns a fixed priority (score) to every task, and the scheduler arranges the tasks in the ready queue based on their priority (score) and executes the ones with the higher priority and so on. Priority scheduling is a method of scheduling processes based on priority. In this method, the scheduler chooses the tasks to work as per the priority, which is different from other types of scheduling, for example, a simple Round Robin. Priority scheduling involves priority assignment to every process, and processes with higher priorities are carried out first, whereas tasks with equal priorities are carried out on a First-Come, First Served (FCFS) or Round Robin basis

4.8 PRIORITY ASSIGNMENT ALGORITHM

```
Priority (delay Ti, STest, SBCAT)
1.  maximum allowed delay no greater than estimated service time
if delay i
T == STest, i
Return (H)
2.  maximum allowed delay between threshold T1 and T2
else if T1 < delay i
T <= T2
if SBCAT ==1
Return (H)
else if SBCAT ==2
Return (M)
else if SBCAT ==3
Return (L)
3. maximum allowed delay greater than threshold T2
else delay i
T > T2
if SBCAT ==1
if QH is not full
Return (H)
else
Return (M)
else if SBCAT ==2
If QM is not full
Return (M)
else
Return (L)
else if SBCAT ==3
Return (L)
```

Table 3**:** shown a priority scheduling algorithm for different types of process. Here P1, P2, P3, P4, P5, P6 are the process.

Table 3: Priority scheduling algorithm

Process	Burst time in Sc.	Average time in Sc.	Turn around time in Sc.	Waiting time in Sc.
P1	3	2	4	1
P2	1	2	6	7
P3	5	3	1	8
P4	1	6	2	3
P5	5	4	7	1
P6	8	2	3	6

4.9 ROUND ROBIN CPU SCHEDULING ALGORITHM

To the best of our knowledge, the Round Robin scheduling algorithm is one of the simplest and most used scheduling algorithm up to this moment. The concept of this algorithm is to share the CPU time among all scheduled tasks on a ready queue. The most important aspect of the Round Robin algorithm is the time slice (Time Quantum) that will be allocated to each task submitted for execution. While the time quantum is a decisive characteristic on the Round Robin algorithm, several proposed Round Robin based algorithms are suggesting static time quantum that segments the CPU time among all submitted tasks, nevertheless, a static time quantum is not always the best solution. A more viable alternative is the use of dynamic time quantum that adapts the CPU time slices to the tasks changes happening on the ready queue for execution. Under the same topic, the Round Robin based algorithm proposed on this paper uses a dynamic time quantum and adds a smarter layer to the existing algorithm in order to adjust the CPU time to different situations.

CHAPTER 5
SIMULATION RESULT AND DISCUSSION

5.1 SIMULATION RESULT AND DISCUSSION

In the FIFO scheduling, there are 6 processes with process ID P1, P2, P3, P4, P5 and P6. P1 burst time 5, P2 burst time 3, P3 burst time 7, P4 burst time 1, P5 burst time 2 and P6 burst time 4 in the ready queue. Also P1 average time 1, P2 average time 4, P3 average time 2, P4 average time 5, P5 average time 2 and P6 average time 1. The processes and their respective burst time and average time are shown in the following Figure 2. It is Non Pre-emptive algorithm, which means the process priority doesn't matter. If a process with very least priority is being executed, more like daily routine backup process, which takes more time, and all of a sudden some other high priority process arrives, like interrupt to avoid system crash, the high priority process will have to wait, and hence in this case, the system will crash, just because of improper process scheduling. Not optimal Average Waiting Time. Resources utilization in parallel is not possible, which leads to convoy effect, and hence poor resource (CPU) utilization.

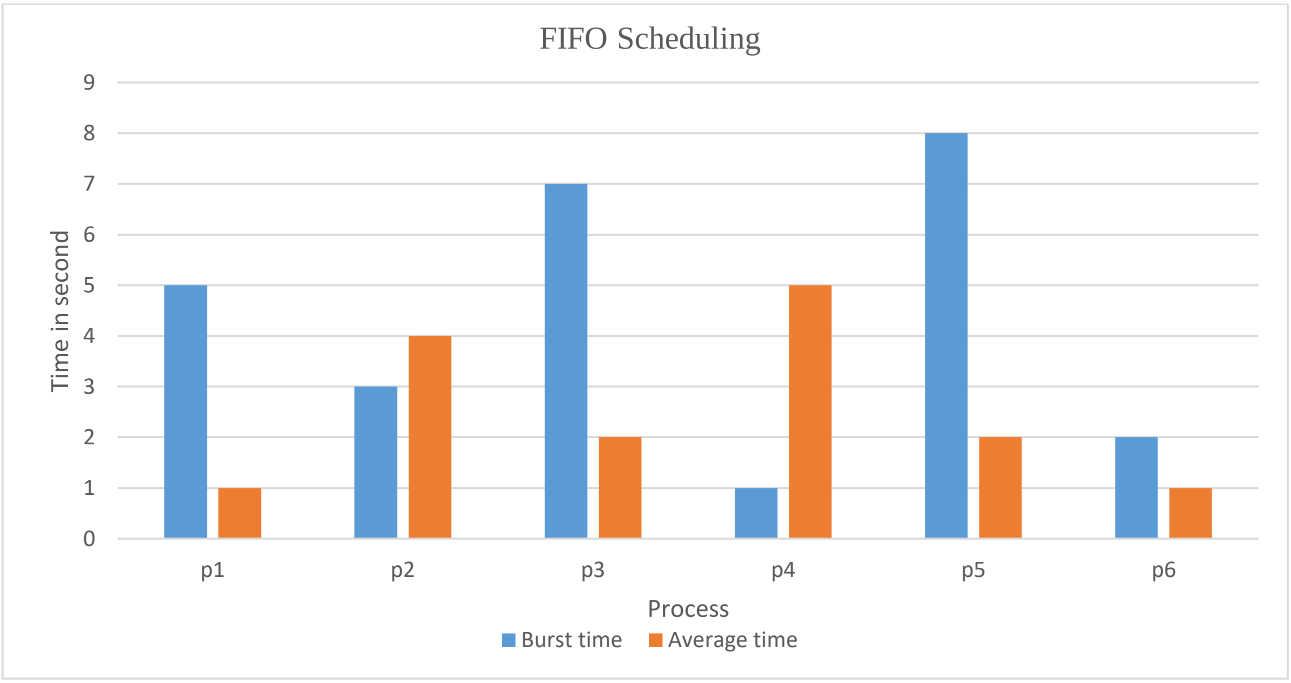

Figure 2 : FIFO scheduling algorithm

In the Round Robin scheduling, there are 6 processes with process ID P1, P2, P3, P4, P5 and P6. P1 burst time 4, P2 burst time 1, P3 burst time 2, P4 burst time 3, P5 burst time 5 and P6 burst time 8 in the ready queue. The processes and their respective burst time and are shown in the following Figure 3. Round Robin is the preemptive process scheduling algorithm. Each process is provided a fix time to execute, it is called a quantum. Once a process is executed for a given time period, it is preempted and other process executes for a given time period. Context switching is used to save states of preempted processes.

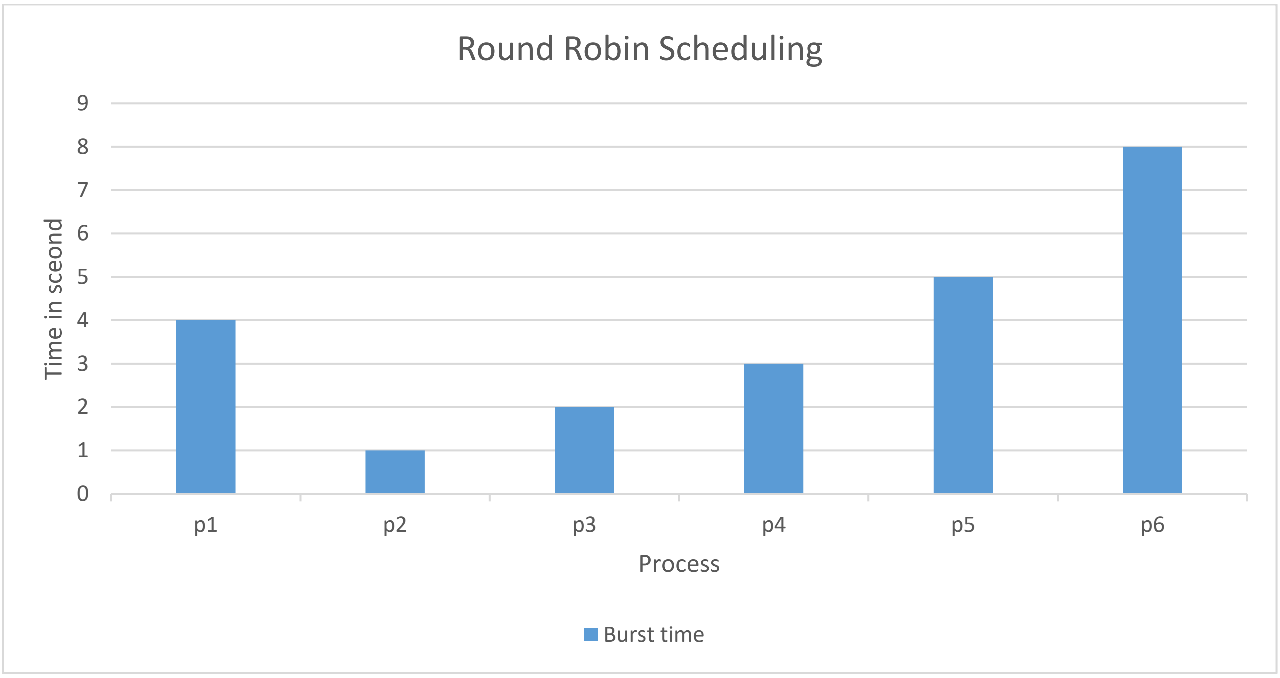

Figure 3 : Round Robin scheduling algorithm

In the Priority scheduling, there are 6 processes with process ID P1, P2, P3, P4, P5 and P6. P1 burst time 3, P2 burst time 1, P3 burst time 5, P4 burst time 1, P5 burst time 5 and P6 burst time 8 in the ready queue. Also P1 average time 2, P2 average time 2, P3 average time 3, P4 average time 6, P5 average time 4 and P6 average time 2. P1 turn around time 4, P2 turn around time 6, P3 turn around time 1, P4 turn around time 2, P5 turn around time 7 and P6 turn around time 3 in the. P1 waiting time 1, P2 waiting time 7, P3 waiting time 8, P4 waiting time 3, P5 waiting time 1 and P6 waiting time 6. The processes and their respective burst time, average time, turn around time and waiting time are shown in the following Figure 4. In Priority scheduling algorithm, the chances of indefinite blocking or starvation. A process is considered blocked when it is ready to run but has to wait for the CPU as some other process is running currently. But in case of Priority scheduling if new higher priority processes keeps coming in the ready queue then the processes waiting in the ready queue with lower priority may have to wait for long durations before getting the CPU for execution.

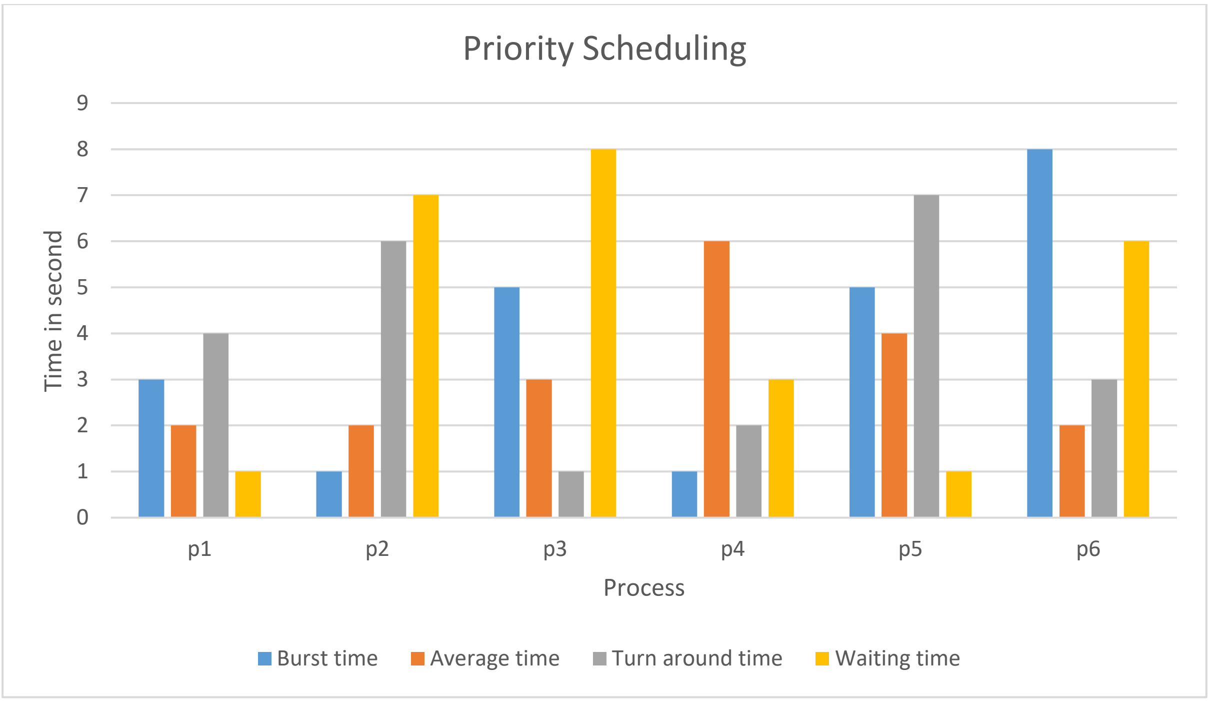

Figure 4 : Priority scheduling algorithm

In Figure 5 shown a combination of FIFO or First Come First Serve, Round Robin and Priority scheduling algorithm. To implementing the “Methods of Resource Scheduling Based on Optimized in Fog Computing” we use three different scheduling techniques namely First Come, First Services (FCFS), Round Robin and Priority scheduling respectively. After applying those techniques we found that, the Round Robin Scheduling technique is more fruitful and effective than others. The reasons for choosing Round Robin scheduling are, Firstly each process is served by the CPU for a fixed time quantum, so all processes are given the same priority. Secondly starvation doesn't occur because for each Round Robin cycle, every process is given a fixed time to execute.

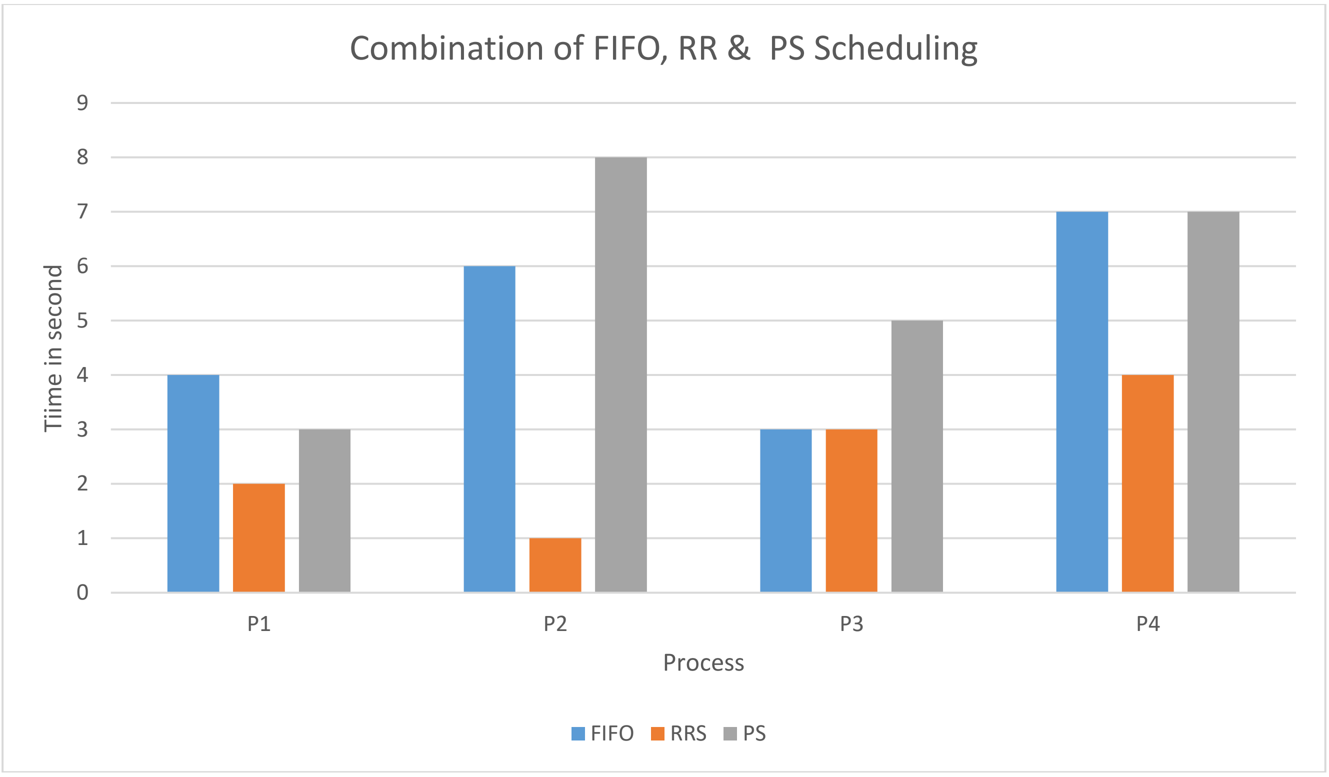

Figure 4 : Combination of FIFO, RRS and PS Algorithm

A big advantage of Round Robin scheduling over non-preemptive schedulers is that it dramatically improves average response times. By limiting each task to a certain amount of time, the operating system can ensure that it can cycle through all ready tasks, giving each one a chance to run. With Round Robin scheduling, interactive performance depends on the length of the quantum and the number of processes in the run queue. A very long quantum makes the algorithm behave very much like First Come, First Served scheduling since it's very likely that a process with block or complete before the time slice is up. A small quantum lets the system cycle through processes quickly. This is wonderful for interactive processes. In Round Robin based algorithm, which calculates a dynamic time quantum based on the average burst time of tasks on a queue list and then assign the amount of time to each task in the queue. The algorithm introduces a test to verify if the time quantum was enough for task termination and reallocate the necessary time to complete the task execution in the opposite case. Round Robin scheduling, time quantum calculated from the average burst time of tasks on the ready queue and it is dynamically adjustable to those tasks that require greater time than the allotted time quantum. Round Robin algorithm that calculates the time quantum dynamically as the average burst time of tasks on the ready queue without any task arrangement and assigns the time quantum based on the tasks arrival time to the ready queue. Round Robin based algorithm that calculates a dynamic time quantum based on priority otherwise the algorithm adapts the tasks execution to the shorter job first algorithm. Round Robin scheduling, which compares the remaining burst time of a task to the algorithm static time quantum after the first allocation. If the remaining time is less than one time quantum the CPU will reallocate the time quantum to the task otherwise it is sent to a waiting queue.

CHAPTER 6
CONCLUSION & FUTURE WORK

6.1 CONCLUSION

In this paper, an enhanced version of the Round Robin algorithm, called Smarter Round Robin (SRR) has been introduced. The evaluations conducted to test this proposed algorithm showed great benefits of this version in comparison to the other existing scheduling algorithms especially those integrated with cloud sim simulation toolkit. The improvements we brought to the Round Robin algorithm aims to inject a smarter layer to the existing algorithm in order to adapt the algorithm to different situations that comes with the new delivery model of cloud computing as well as reducing the run-time of big data processing. While the shorter job first algorithm gave great results in regards to the average turn around time, average waiting time and number of context switches it still lacks the intelligence and the capability to run tasks in parallel mode or to detect situations where a user executes several instances of the same tasks. The First Come First Served algorithm shows poor performances because of its functioning mechanism that executes the tasks based on their logical arrival time. Consequently, once a time consuming task takes on the lead of the ready 8 journal of data mining and digital humanities. Queue list, the other submitted tasks will have to wait for this huge task to finish execution before having the chance to get to the processor thus a long waiting time. The previous proposed version of the Round Robin algorithm used a dynamic CPU quantum based on the average burst time, whereas the time quantum will be recalculated each time there are tasks on the waiting list. This new perception exhibits great benefits but was unable to detect the changes happening on the waiting list, therefore, even if there were only two tasks on the waiting list and their burst time is very low the algorithm instructs the CPU to recalculates the time quantum thus a high number of context switches and average waiting time. Finally, The Round Robin algorithm version integrated with the cloud sim simulation toolkit uses a static time quantum calculated based on the system resources, the static quantum is then allocated for each submitted task. This algorithm is deficient in respect to the average turnaround time, average waiting time and number of context switches, which were always very high; hence, it is not well suited for cloud computing nor Big Data.

6.2 FUTURE WORK

This paper studied the resource scheduling problem in fog computing our future work will be concentrated on implementing the proposed algorithm with real life cloud computing platforms as well as on bringing more intelligence to the proposed scheduling algorithms especially neural networks that can have a massive impact on compute resources allocation. We applied the First Come First Serve, Round Robin scheduling algorithm and Priority scheduling algorithm to cluster fog resources, which narrows the range of user requirements for matching resources. Finally, from the experimental analysis, the objective function value of the FIFO algorithm in this paper was shown to have a faster convergence speed than the RR and Priority scheduling algorithm. In addition, the proposed FIFO algorithm can match user requests with the appropriate resource categories quicker and improve user satisfaction. In future work, we will consider the dynamic changes of resources and propose a new scheduling strategy to improve the utilization of resources and ensure user satisfaction.

REFERENCES

1 Atzori, L.; Iera, A.; Morabito, G. The internet of things: A survey. Comput. Netw. 2010, 54, 2787–2805. [CrossRef]

2. Lei, Y.; Cai, Z. Dynamic scaling of virtual clusters with bandwidth guarantee in cloud datacenters. In Proceedings of the IEEE Infocom—The IEEE International Conference on Computer Communications, San Francisco, CA, USA, 10–14 April 2016.

3. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I. A view of cloud computing. Int. J. Comput. Technol. 2013, 4, 50–58. [CrossRef]

4. Lei, Y.; Shen, H.; Cai, Z.; Ling, L.; Pu, C.; Lei, Y.; Shen, H.; Cai, Z.; Ling, L.; Pu, C. Towards bandwidth guarantee for virtual clusters under demand uncertainty in multi-tenant clouds. IEEE Trans. Parallel Distrib. Syst. 2018, 29, 450–465.

5. Lei, Y.; Shen, H.; Sapra, K.; Lin, Y.; Cai, Z. CoRE: Cooperative end-to-end traffic redundancy elimination for reducing cloud bandwidth cost. IEEE Trans Parallel Distrib. Syst. 2017, 28, 446–461.

6. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; pp. 13–16

7.Savitha. P, J Geetha Reddy, A Review Work on Task Scheduling in Cloud Computing Using Genetic Algorithm, International Journal of Scientific & Technology Research, Volume 2, Issue 8, August 2013

8.N. Srilatha, M. Sravani, Y. Divya, "Optimal Round Robin CPU Scheduling Algorithm using Manhattan Distance", International Journal of Electrical and Computer Engineering, Vol. 7, No. 6, December 2017

9.Verma, M.; Bhardwaj, N.; Yadav, A.K. Real Time Efficient Scheduling Algorithm for Load Balancing in Fog Computing Environment. Int. J. Inf. Technol. Comput. Sci. 2016, 8, 1–10. [CrossRef]

10. Misra, S.; Sarkar, S. Theoretical modelling of fog computing: A green computing paradigm to support IoT applications. IET Netw. 2016, 5, 23–29.

11. Aazam, M.; Huh, E.N. Fog computing and smart gateway based communication for cloud of things. In Proceedings of the International Conference on Future Internet of Things & Cloud, Barcelona, Spain, 27–29 August 2014.

12. Stojmenovic, I.; Sheng, W. The Fog computing paradigm: Scenarios and security issues. In Proceedings of the Federated Conference on Computer Science & Information Systems, Warsaw, Poland, 7–10 September 2014.

13. Yu, L.; Chen, L.; Cai, Z.; Shen, H.; Liang, Y.; Pan, Y. Stochastic load balancing for virtual resource management in datacenters. IEEE Trans. Cloud Comput. 2016, PP, 1. [CrossRef]

14.V. Vinothina, et al., “A Survey on Resource Allocation Strategies in Cloud Computing,” International Journal of Advanced Computer Science and Applications, vol/issue: 3(6), 2012

15.V. Goyal, “Review: Layers Architecture of Cloud Computing,” International Journal of Computing & Business Research.

16.N. Zanoon and D. Rawshdeh, “STASR: A New Task Scheduling Algorithm For Cloud Environment,” Network Protocols and Algorithms, vol/issue: 7(2), 2015

17.A. Agarwal and S. Jain, “Efficient Optimal Algorithm of Task Scheduling in Cloud Computing Environment,” International Journal of Computer Trends and Technology, vol/issue: 9(7), 2014

18.Operating system concept by Abraham Silberschatz peter Baer Galvin Greg Gange,nine edition

19. 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

20. 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

21. 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

22. 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

23. 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

24. 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

25. 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