AN IMPROVED REVAMPED MEAN ROUND ROBIN ALGORITHM BASED ON DYNAMIC TIME QUANTUM FOR IMPROVING RESOURCE ALLOCATION IN CLOUD COMPUTING

[1]Bamaiyi Sule [2]Etemi Joshua Garba [3]Yusuf Musa Malgwi
[1,2,3] Department of Computer Sciences, Modibbo Adama University Yola State, Nigeria
Email: [1]bams4ann@gmail.com; [2]e.j.garba@mau.edu.ng; [3]yumalgwi@mau.edu.ng

ABSTRACT

Revamped Mean Round Robin (RMRR) CPU scheduling algorithm is an impartial scheduling algorithm that gives same time quantum to all processes. However RMRR algorithm was not build to be implemented in cloud computing environment; In addition, it context switching is not at it minimal, due to the way it preempted and time quantum selected, even though this two aforementioned processes are (preempted and the time quantum) are very critical as it affects the algorithm's performance. In this research, a new algorithm that improved the RMRR CPU scheduling algorithm was presented, and tailored towards efficient resource allocation in cloud computing environment. The New RMRR algorithm was implemented and compared with revamped mean round robin, New Improved Round Robin, Improved Round robin with Varying Quantum-Time and the developed algorithm, considering Average Waiting Time (AWT), Average Turnaround Time (ATAT), and Number of Context Switches (NCS) as performance measurement parameters. The result of the new RMRR technique reported a minimal AWT, improves on ATAT and better than the existing systems. Also modification was made on the improved algorithm to enable implementation on cloud environment. Built on these results, recommendations are made for the adoption of the new RMRR CPU algorithm over other scheduling algorithms for quality of service optimization in cloud computing environment.

KEYWORDS: Round Robin, Revamped mean Round Robin (RMRR), Scheduling Algorithm, Cloud Computing, Waiting Time, Turnaround Time, Response Time, average waiting time (AWT), average turnaround time (ATAT), number of context switches (NCS).

1. INTRODUCTION

In recent time, the concept of resource allocation is paramount to address in the cloud computing environment. Cloud computing is an emerging and trending technology (Pradhan, Prafulla. Behera, Ray, 2016). It aims to provide reliable, customized and Quality of Service (QoS) guaranteed computing dynamic environments for end-users. It is platform independent, hence users have no need to install any piece of software on their local PC and the resources provided by the cloud developers are used on a rented basis in a pay as you go manner (Gokilavani, Selvi, and Udhayakumar, 2013; Bhavani and Guruprasad, 2014).

Currently Cloud Computing is an emerging computing technology which is the big step in development and deployment of an increasing number of distributed applications (Pradhan, et al., 2016). Cloud Computing is defined as the computing model that operates based on Clouds. In turn, the Cloud is defined as a conceptual layer11 that operates above an infrastructure to provide services in a timely manner (Prakash et al., 2014). The emergence of cloud Computing is aim at providing reliable, customized and Quality of Service (QoS) guaranteed computing dynamic environments for end users (Pradhan et al., 2016). Distributed processing, parallel processing and grid computing together emerged as cloud computing. The basic principle of cloud computing is that user data is not stored locally but is stored in the data center of internet (Gokilavani et al., 2013). According to the National Institute of Standards and Technology (NIST) definition, Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction (Mell and Grance, 2011; Biwas, Samsuddoha, Asif, and Ahmed, 2023).

There are numerous advantages of cloud computing the most basic ones being lower costs, re-provisioning of resources and remote accessibility. Cloud computing lowers cost by avoiding the capital expenditure by the company in renting the physical infrastructure from a third party provider (Pradhan, et al., 2016; Biwas et al., 2023). Due to the flexible nature of cloud computing, we can quickly access more resources from cloud providers when we need to expand our business. The remote accessibility enables us to access the cloud services from anywhere at any time. To gain the maximum degree of the above mentioned benefits, the services offered in terms of resources should be allocated optimally to the applications running in the cloud. Cloud computing, at its simplest, is a collection of computing software and services available from a decentralized network of servers (Yaashuwanth, and Ramesh, 2015; Igbal, Ullah, Khan, Aslam, Shaheer, Humayon, Salahudin, and Adeel, 2023).

1.1 Statement of the problem

Resource allocation becomes a challenging issue to the cloud service providers when the numbers of available resources are limited and the demand for the resources is high (Dash et al., 2018; Biwas et al., 2023). Resource allocation is very necessary in cloud computing for effective sharing and utilization of minimal resources (Yaashuwanth, and Ramesh, 2015; Igbal et al., 2023). The traditional Round Robin Scheduling tend to proffer solution to this by introducing the quantum time. The time quantum is fixed and then processes are scheduled such that no process gets CPU time more than one-time quantum at a go. If time quantum is too large, the response time of the processes will be too much which may degrade performance in an interactive environment (Gokilavani et al., 2013; Sakshi, Sharma, Sharma, Kautish, Alsallami, Khalil, and Mohamed, 2022). If time quantum is too small, it causes unnecessarily frequent context switch which lead to more overheads resulting to less throughput and if it is too large is works as First Come First Serve algorithm (Khatri, (2016); Pradhan et al., 2016; Omotehinwa, Azeez, and Olafunyi, 2019).

2. Problem Definition

Cloud computing is an emerging technology in the field of information technology (IT). It is an infrastructure that provides accesses to various resources such as server, network, operating system and storage space for use by user payable on demand (Kathuria, et al., 2016). Cloud computing gives room for an organization and individuals to make use of the resource available in the cloud to save them the cost of buying and maintaining these resources (Yaashuwanth, and Ramesh, 2015; Igbal et al., 2023).

2.1 Cloud Computing Models

Cloud computing has two types of models as described by (Prahdhan, 2016; Biwas et al., 2023) which are namely; deployment model and service model.

2.1.1 Cloud computing deployment model

The way or manner at which the cloud can be located is called Cloud Computing deployment model. There are four types of deployment model: Public Cloud, Private Cloud, Hybrid Cloud and Community Cloud (Pradhan et al. 2016).

2.1.1.1 Public Cloud

Public Cloud as the name implies, it is public and open therefore it is very easy to access. It is less secured and very economical because multiple users share the same resources provided by a single provider hence the cost of the resources is shared among the multiple users causing each user to use the resources at a very cheap cost. Examples of a public cloud include Microsoft Azure, Google App Engine etc. Public clouds help in realization of characteristics like:

a. Flexible and Elastic Environment
b. Freedom of Self-Service
c. pay for what you use
d. Availability and Reliability.

2.1.1.2 Private Cloud

Private Cloud: As the name implies is private and close therefore it very difficult for the general public to access it because it is restricted to function only within an organization and only the people within the organization can access it. It is more secured and very expensive. It is smaller than public cloud. An example of a private cloud is the Eucalyptus Systems (Prahdhan, 2016). Private clouds help in realization of characteristics like:

a. Enhanced Security Measures
b. Dedicated Resources
c. Greater customization

2.1.1.3 Community Cloud

Community Cloud as the name implies it is a cloud that is within a community i.e. this cloud is made up of a group of organizations such as banking or educational sector with common interest. It is smaller than public cloud and bigger than private cloud. This cloud is accessible to only members of the community. Community cloud is not as cheap as public cloud and may exist locally or remotely. An example of a Community Cloud includes Facebook, Twitter, Instagram etc.

Community Cloud helps in realization of characteristics like:

a. Grid Computing

b. Digital Ecosystems
c. Green Computing

2.1.1.4 Hybrid Cloud

Hybrid Cloud This model combines of two different clouds, i.e. public and private cloud. Data/applications can be transferred between the two clouds through their interfaces. Hybrid cloud enables cloud providers to manage sensitive data/information within the organization while spreading less sensitive data/information to the general public. The users of hybrid cloud enjoy the advantage of two different clouds. An example of a Hybrid Cloud includes Amazon Web Services (AWS).
Hybrid Clouds realize the characteristics:

a. Optimal Utilization
b. Data Centre Consolidation

2.1.2 Cloud computing service model

Cloud Computing Service Model provides services based on three models. These models are arranged in form of a layer from the top to the bottom.

2.1.2.1 Infrastructure as a Service (IaaS)

Infrastructure as a Service (IaaS) is a cloud computing offering in which a vendor provides users access to computing resources such as servers, storage and networking. Organizations use their own platforms and applications within a service provider’s infrastructure.
The following are key features of Infrastructure as a Service:

a. Instead of purchasing hardware outright, users pay for IaaS on demand.
b. Infrastructure is scalable depending on processing and storage needs.
c. Saves enterprises the costs of buying and maintaining their own hardware.
d. Because data is on the cloud, there can be no single point of failure.
e. Enables the virtualization of administrative tasks, freeing up time for other work.

This is also known as the bottom layer. It is the most fundamental and essential cloud service model that offers infrastructure as a service. It combines both the internal and external resources and does not depend on any platform or operating system. Virtualization and multiprocessing are made possible through the hypervisor (Mell and Grance, 2011; Igbal et al., 2023).

2.1.2.2 Platform as a Service (PaaS)

Platform as a Service (PaaS): is a cloud computing offering that provides users with a cloud environment in which they can develop, manage and deliver applications (Biwas et al., 2023). In addition to storage and other computing resources, users are able to use a suite of prebuilt tools to develop, customize and test their own applications, Key features are:

a. PaaS provides a platform with tools to test, develop and host applications in the same environment.

b. Enables organizations to focus on development without having to worry about underlying infrastructure.
c. Providers manage security, operating systems, server software and backups.
d. Facilitates collaborative work even if teams work remotely.

This is also known as the middle layer. It provides run time environment or platform like operating system, programming language execution environment, database, and web server as a service. This platform enhances the development and running of application on a cloud. It saves the developers the stress of buying or managing underlying hardware and hardware.

2.1.2.3 Software as a Service

Software as a Service (SaaS): is a cloud computing offering that provides users with access to a vendor's cloud-based software. Users do not install applications on their local devices. Instead, the applications reside on a remote cloud network accessed through the web or an API. Through the application, users can store and analyze data and collaborate on projects.
The following are key features of Software as a Service:

a. SaaS vendors provide users with software and applications via a subscription model.
b. Users do not have to manage, install or upgrade software; SaaS providers manage this.
c. Data is secure in the cloud; equipment failure does not result in loss of data.
d. Use of resources can be scaled depending on service needs.
e. Applications are accessible from almost any internet-connected device, from virtually anywhere in the world.

This is also known as the top layer. It provides software as a service which Application layer or Software-as-a-Service replaces the applications that are running on the computer. SaaS provides application to users at a minimal cost which enhance the users to have no need to install and run software on their computer (Nayak, Malla, and Debadarshini, 2012; Fiad, et al., 2020).

2.2 Resource Scheduling Methods in Cloud Computing

In cloud computing environment, scheduling methods can be classified into three groups.

a. Resource scheduling: it performs mapping of virtual resources among physical machines
b. Workflow scheduling: it schedules workflows constituting an entire job in a suitable order
c. Task scheduling: this method can be centralized or distributed. It can be performed in homogeneous or heterogeneous environment on dependent or independent tasks. Centralized scheduling uses a single scheduler to perform all mappings. It is easy to implement but it losses scalability and fault tolerance as it always has a bottleneck of single point of failure. In distributed scheduling, scheduling is partitioned among different schedulers. It has high implementation complexity but processor cycles are saved as the work load is distributed to partner nodes. There are two types of distributed scheduling. i.e. heuristic and hybrid

2.3 Classification of Resource Scheduling Algorithm

The existing task scheduling algorithm can be divided into two different categories.

a. Heuristics: Heuristics is a logic procedure for getting solution, heuristic approaches can be either static or dynamic.
b. Hybrid: These algorithms are novel or are developed on the top of some existing methods incorporating more scheduling parameters to improve the performance.

This research focused on the heuristics algorithms and as a result, some heuristics algorithms are reviewed below.

2.3.1 Heuristics

Heuristics is a logic procedure for getting solution (Dash et al., 2015; Neha 2018; Sakshi et al., 2022). It provides an optimal solution in which it uses the knowledge bases for taking the scheduling decisions. Heuristic approaches can be either static or dynamic.

2.3.1.1 Static Heuristic Methods

Static scheduling algorithms consider that all tasks arrive at the same instant of time and they are independent of the system resource's states and their availability (Mythili, et al., 2017; Abubakar, Yusuf, Obiniyi, and Mohammed, 2023). The static heuristics include algorithms like FCFS, RR, OLB, MET, MCT, Min-Min, Max-Min, GA, SA, etc.

2.3.1.1.1 First come first serve (FCFS) method: Collects the tasks and queues them until resources are available and once they become available the tasks are assigned to them based on their arrival time. It is less complex in nature but does not consider any other criteria for scheduling the tasks to machines.

2.3.1.1.2 Round robin (RR) method: Operates using FIFO technique. It allots a resource to each task for a particular time quantum (Amar, Sandipta and Sanjay, 2015; Mythili, et al., 2017).After that the task is pre-empted and queued until its next chance for execution.

2.3.1.1.3 Opportunistic load balancing (OLB) method this heuristic method tries to schedule the tasks to the next available machines based on their expected completion time. Although it tries to utilize the resources equally by making all machines busy at the same time but it has poor make-Span.

2.3.1.1.4 Minimum execution time (MET) method: this heuristic strategy assigns tasks on the machines based on which machine it takes less execution time. It selects the best machine for execution but do not consider the availability of resources at the time of scheduling so load imbalance will occur (Neha, 2018).

2.3.1.1.5 Minimum completion time (MCT) method: this heuristic method selects machines for scheduling the tasks based on the expected minimum completion time of tasks among all the machines available. It considers the load of the machine also before scheduling the task on that machine. The task may not have minimum execution time on the same machine (Nayak et al., 2012).

2.3.1.1.6 Min-min method: this heuristic method selects the smallest task first from all the available tasks and assigns it to a machine which gives the minimum completion time (fastest machine) for that task. It increases the total completion time of all the tasks and hence increases the make-span but it does not consider load of the machines before scheduling as simply assigning smaller tasks on faster machines. The expected completion time and execution time for a task are considered to be almost same values or close values. The long tasks have to wait for completing the execution of smaller ones. But the method improves the system's overall throughput (Arjun, 2017).

2.3.1.1.7 Max-min method: this is similar to min-min except that it selects the longest task(with maximum completion time) first to schedule on the best machine available based on the minimum completion time of that particular task on all available machines. Here the smaller tasks have to starve and load balancing is also not considered. Anyway it increases the make-span and system throughput than the min-min strategy since the longest task determines the make span of all the available tasks in the system. Hence in max-min the longer tasks can be executed first in faster machines as well as smaller tasks can be executed in parallel on other possible machines which results in better make span and balanced load than the previous method (Dash et al., 2015).

2.3.1.1.8 Genetic algorithm (GA): is a heuristic approach which is used to perform near optimal scheduling. It is based on survival of the fittest and there are four different operations in GA i.e. evaluation, selection, cross over and mutation. The initial population represents the possible mappings of the given task list on the available machines. Each job is represented as a vector in which each position of that vector represents a task in the task list. The value in each position represents the machine to which the task is mapped. Each job represents a chromosome. Every chromosome has a fitness value indicating the overall execution time of all the tasks (make-Span) which is formed from the mapping of tasks to resources constituting that chromosome and it is selected such that it reduces make-Span. This method uses past results with present results to get better possible mappings (Kochut et al., 2015).

2.3.1.1.9 Simulated annealing (SA) algorithm: is a heuristic approach which is used to perform near optimal scheduling. It is an iterative method which is similar to genetic algorithm. The algorithm starts with a single solution (mapping) selected from a random distribution. The initial version of SA is evaluated to get a better version. After mutation the new make-span is analyzed. If it is lower than the previous one, then replace the old one with the new make-span. Simulated Annealing finds poorer solutions than Genetic Algorithm. The features of genetic algorithm and simulated annealing can be combined to get a better scheduling solution (Bhavani, and Guruprasad, 2014).

2.3.1.2 Dynamic Heuristic Scheduling Methods

In dynamic scheduling methods tasks are dynamic in nature (Yaashuwanth and Ramesh 2015; Olofintuyi, Omotehinwa, Oyekanmi, and Olajubu, 2019).Here tasks arrive at different points of time and it is dependent on the system machine's state. Dynamic scheduling algorithms are classified into two categories:

2.3.1.2.1 Online mode: tasks are assigned instantly once they arrive in the system. MCT, MET, OLB are example of online mode and they work similar to static algorithms. Switching algorithm which switches between MET and MCT as per the load of the system also belong to online mode.

2.3.1.2.2 Batch mode: tasks are collected as a group and scheduled at predefined times. Min-min, max-min, round robin and suffrage heuristic are some examples for batch mode. In suffrage heuristic, tasks are scheduled based on a suffrage value. It is calculated from the first and second earliest completion times of a task. The suffrage values are compared for different tasks and the task with higher suffrage is selected for scheduling on a same resource (Yaashuwanth and Ramesh 2015).

2.4 Related works on variant scheduling Algorithm

The author (Abdulrahim, Abdullahi, and Sahalu, 2014), proposed to improved the work of (Manish and Abdulkadir, 2012), titled "New Improved Round Robin CPU (NIRR) scheduling algorithm". The proposed algorithm was implemented and benchmarked against five other algorithms. From their experiment, the proposed algorithm produces minimal average waiting time, turnaround time and number of context switches more than the other algorithms in their literature, but has a major drawback on response time.

Yaashuwanth and Ramesh, (2015) and Abubakar et al., (2023), propose a Modified Round Robin Algorithm (MRR) where Time Slice (TS) is calculated and allocates based on (range× total no of process (N)) divided by (priority (pr) × total no of process (p)). Range is calculated by (maximum burst time + minimum burst time) divided by the scheduling process. From their results MRR outperform the Simple RR for all parameters. Samal et al. (2013) also proposed another algorithm called Time Slice Priority Based Round Robin (TSPBRR)**.** TSPBRR focuses on priority and is compared to other variants. From their experimental result, TSPBRR focuses on priority and as compared to other variants. It is also observed that the ATAT and AWT in TSPBRR are superior to that in MRR algorithm. Let 'TQ_i' is the time quantum in round i. The number of rounds i varies from 1 to n, where value of i increments by 1 after every round till ready queue is not equal to NULL.

The author (Pradhan et al. 2016), proposes an algorithm that modifies the RR algorithm with a dynamic time quantum. The algorithm calculates time quantum by maintain two different queues namely PRQ and RQ. The time quantum is obtained by taking the mean of the burst time of processes in the RQ. When it allocates CPU to first process present in RQ if the remaining CPU burst time of the currently running process is less than time quantum then the CPU will still allocate the currently running process for remaining CPU burst time and after the process complete its execution, it will be removed from the ready queue and allocate CPU to next process. This mechanism helps in reducing the number of contexts switching and waiting time, but has a major drawback on response time and turnaround time.

In the work of (Gupta, Priya, Carlos, garg, and Dinesh, 2022), Gupta Elaborate an optimized his earlier work of Priority Round Robin for better CPU Scheduling. The time-specific applications are assigned to Central Processing Unit (CPU) of the system and one of the most promising functions of the time-sharing operating systems is to schedule the process in such a way that it gets executed in minimal time. At present, the Round Robin Scheduling Algorithm (RRSA) is the most widely used technique in a timesharing operating system because it gives better performance than other scheduling techniques, namely, First Come First Serve (FCFS), Shortest Job First (SJF), and Priority scheduling. The major challenge in RRSA is the static value of Time Quantum (TQ) which have plays a pivotal to decrease or increase the performance of the system. In existing literature, many statistical techniques are used for identifying efficient time quantum for RRSA. However, there is limited exposure in existing literature on generating a learning model for identifying optimized TQ. In this research work, a new research direction is given for identifying Optimized TQ by training a learning model and predicting optimum TQ value.

(Khatri (2016), proposed an Improved Dynamic Round Robin (DRR) model in which the median of the set of processes in the ready queue is considered as the optimal time quantum and if the median is less than 25 then its value must be considered as 25 to avoid the overhead of context switch. The first process in the ready queue is allocated to the CPU for a time interval of up to 1 time quantum. If the remaining burst time of the currently running process is less than or equal to 1 time quantum, the processor again allocated to the same process. Else the process is preempted and place at the rear of the ready queue otherwise the process will be halted. Experimental result shows that the proposed algorithm IDRR, when compared with various variants of Round Robin algorithm it produces minimal average waiting time and average turnaround time but has a major drawback on response time.

Nayak, et al. (2012), proposed an algorithm which employs the median method to find out optimal time quantum. All processes in ready queue are arranged in ascending order based on their burst time. Median and the highest burst time is calculated and set as the time quantum. Their experimentally results performs better than the RR algorithm, by reducing context switching, average waiting and average turnaround time

The Researchers (Manish and Faizur, 2014), propose an improve “Round Robin with varying time quantum (IRRVQ)” employing the features of Shortest Job First and Round Robin scheduling with varying time quantum to improve on Round Robin CPU scheduling. IRRVQ arrange all processes found in the ready queue in ascending order according to their burst time and set time quantum equal to burst time of the first process in the ready queue. Their simulation results show that the waiting time and turnaround time have been reduced in the proposed algorithm compared to traditional RR.

Pradhan (2016) proposed a "Modified Round Robin Algorithm for Resource Allocation in Cloud Computing". In their work they modify the traditional round robin that has the drawback of static time quantum and high waiting and turnaround time. They address this issue by employing the use of dynamic time quantum to minimize average waiting and turnaround time. Their experiment was performed using Matlab. Although their result drastically reduced the high average waiting and turnaround time found in the traditional round robin. The problem with their work is that, jobs with smaller remaining burst time are always kept in the queue which lead to higher response time and increase in context switches.

The work of (Dash et al., 2015), presented a "Dynamic Average Burst Round Robin (DABRR) using dynamic time quantum". The processes are arranged in an ascending order in the ready queue base on their burst time. The average of the active processes in the ready queue are calculated and set as the time quantum after each iteration. Their results improved performance in terms of average waiting time and average turnaround time.

3. Methodology

3.1 The Proposed Improved Revamped Mean Round Robin Algorithm

The Round Robin (RR) CPU scheduling algorithm is an impartial scheduling algorithm that gives same time quantum to all processes. The selection of the time quantum is very critical as it affects the algorithm's performance. This new algorithm improved on the Revamped Mean Round Robin (RMRR). The proposed algorithm was implemented and benchmarked against other algorithms available in the literature. The proposed algorithm compared with the other algorithms, produces minimal average waiting time (AWT), average turnaround time (ATAT), and number of context switches (NCS). It algorithm uses neural fuzzy to find the best allotted time base on the historical data from the previous handled task. The algorithm maintain two registers.

a. SR: stores the sum of the remaining burst time in the ready queue
b. AR: stores average of the burst times by dividing the value found in the SR

by the count of processes found in the ready queue. AR is set as the time quantum. The algorithm will always check if the remaining burst time of a currently running process is less or equal to the time quantum. If remaining burst time of a currently running process is less or equal to the time quantum, then the current running process will not be preempted otherwise it will be preempted and stored at the rear of the ready queue. The SR will be updated by subtracting the time consumed by this process and AR will be updated according to the new data.

3.1.1 Assumptions

a. Each operation cannot be interrupted during its performance (Non-preemptive).

b. Each machine can perform at most one operation at a giving time.
c. Each machine becomes available to another operation once the operation which are assigned is completed
d. All machine are available at time t = 0.
e. All process can be started at time t = 0 if the remaining Burst time of the running process is > TQ.
f. If the remaining time of a running process is less than the TQ resource is allocated to the process to finish.
g. Setting up time of machines and move time between operations are negligible.
h. Machines are independent from each other.
i. There are no precedence constraints among operations of different jobs.
j. Release time or due dates are not specified (Arash, et al, 2010).

3.2 Evaluation Metrics

The performance of the proposed algorithms will be evaluated using the following performance metrics:

Pre-Ready Queue: stores the newly arrived process before moving it to ready queue.

Ready Queue: stores the process ready to be executed.

We are going to defined it Mathemat ically as:

i. Let TT_n denote Turnaround time of the nth process
ii. Let CT_n denote completion time of the nth process
iii. Let AT_n denote arrival time of the nth process
iv. Let WT_n denote waiting time of the nth process
v. Let NCS denote Number of Context Switching
vi. γ represents the number of process in the set under consideration.
vii. $\beta t[\mathrm{i}]$ denotes the burst time of a given ith process in the queue.
viii. Let ω represent the number of times pre-emption takes place in the CPU.

Turnaround Time of the nth process is determined thus: $TT\mathrm{n} = CT\mathrm{n} - AT\mathrm{n}$ (3.1)

Waiting Time of the nth process is determined thus: $\mathrm{W}T\mathrm{n} = TT\mathrm{n} - \beta t[\mathrm{i}]$ (3.2)

Hence, the Average Waiting Time (AWT) is determined thus:

$$\mathrm{AWT} = \frac{1}{\gamma}\sum_{\mathrm{n}=1}^{\gamma} \mathrm{WTn} \quad (3.3)$$

The Average Turnaround Time (ATT) is determined thus:

$$\mathrm{ATT} = \frac{1}{\gamma}\sum_{\mathrm{n}=1}^{\gamma} \mathrm{TTn} \quad (3.4)$$

The Number of Context Switching is determined thus:

$$\mathrm{NCS} = \sum_{\mathrm{n}=1}^{\gamma} (\omega\mathrm{n} - 1) \quad (3.5)$$

3.3 Research Framework

3.3.1 Improved Revamped Mean Round Robin Algorithm Analysis

In the review of literatures so many approaches were applied to solve Round Robin algorithm resource allocation problem. Some of the researcher's uses single parameter to address the problem while others use multi objectives parameters approaches but all provide solution based on efficient resource allocation. This work tries to imbibe the idea of Revamped Mean Round Robing with improving multi-parameters objectives to address the resource allocation problems which in turn reduce wastage of resources..

3.3.2 Improved Revamped Mean Round Robin Algorithm Design

An Improved Revamped Mean Round Robin Algorithm is a Design Approach that tense to efficient use available resource and allocate it to processes with various time quantum and the efficient use of resource of both the ready queue and the pre-ready queue. This research uses Dynamic Time Quantum for Improving Resource Allocation in Cloud Computing approach not just to provide good result but also to provide a better solution to resource allocation problems.

3.3.3 Design Tools

For the sake of this research work, we used java netbeans to simulate the scheduling problem on intel core CPU, 4GB Ram and 2.20GHz. The utilization of the machine was used to determine which of the machine is more loaded with jobs and the machine that execute fewer jobs.

4. Results and Discussion

4.1 Experimental Data

The experimental data used in carrying out the work is thus, the system considers seven different processes with randomly generated burst time for each processes and their corresponding arrival time randomly generated by the java scheduler. As displayed below in table 4.1 with their correspounding burst time. Process ID: this means process identification number. Each process is tagged with an identification number. The ID is represented by Process 1,Process 2,Process 3,…,Process n. with their corresponding Arrival Time and Burst Time as shown in below.

Table 1: Random Scheduler Generated Processes, Arrival Time and Bust Time

Process	Arrival time	Bust time
P_0	0	90
P_1	10	54
P_2	3	43
P_3	14	14
P_4	19	9
P_5	14	81

4.2 The Various Algorithms Considered

The following algorithm will be run and analyzed and the result will be compared with each other and the performance will be evaluated against the standard evaluation metrics in other to determine the best performing algorithm, the algorithm are:

a. The Revamped Mean Round Robin (RMRR)
b. New Improved Round Robin (NIRR)

c. Improve Round Robin with Varying Quantum-Time (IRRVQ)
d. Improve Dynamic Round Robin (IDRR)
e. The Developed Algorithm

4.3 Results of Various Considered Algorithms

In this session seven (7) processes were used in demonstrating the behavior of the algorithms. The five algorithms above uses the same processes but gave different solutions. These solutions occur as a result of the nature of the algorithms. The five algorithms are revamped mean round robin, New Improved Round Robin, Improved Round robin with Varying Quantum-Time and the developed algorithm. Waiting time, turnaround time, response time and context switch are used as metrics to evaluate the performance of the various algorithms as shown in Table 4.1 below.

Table 2: Comparative Result Obtain from Various Algroithm Considered

Metrics	RMRR	New improved RR	Improved RR with varying quantum-time	Improved dynamic RR	Developed Algorithm
Average Waiting Time (s)	172	164	166	164	160
Average Turnaround Time (s)	209	201	203	201	197
Number of Context Switch (s)	14	7	7	7	7
Response Time (s)	6	48.71	22.73	22.73	22.73

4.4 Graphical Comparison of Various Algorithms Considered in table 2

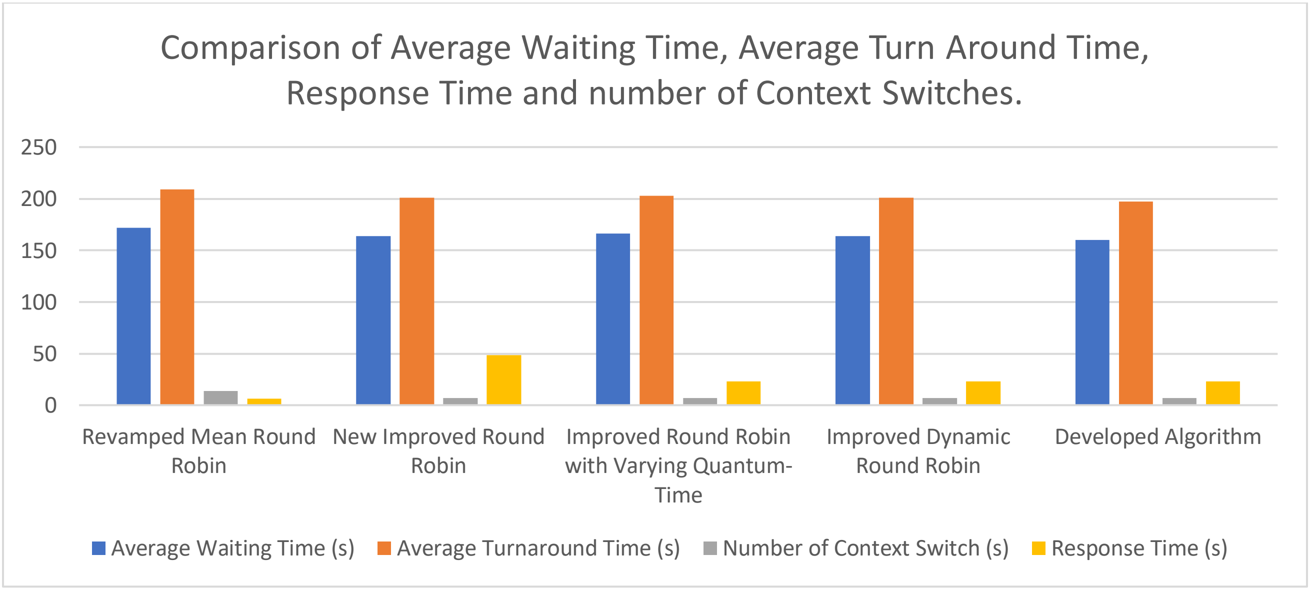

Figure 4.1: Comparative analysis of selected algorithms

4.5 Discussion of Result

From session 4.1 above, seven processes were used in demonstrating the behavior of the algorithms, and the result and graphical representation is seen in session 4.3 and 4.4 respectively.

The five algorithms above uses the same processes but gave different solutions. These solutions occur as a result of the nature of the algorithms. The five algorithms are revamped mean round robin, New Improved Round Robin, Improved Round robin with Varying Quantum-Time and the developed algorithm. Waiting time, turnaround time, response time and context switch are used to evaluate the performance of the various algorithms as shown in Table 2 above.

4.5.1 Evaluation of Average Waiting Time

The revamped mean round robin, when compared side by side with the developed algorithm shows an improve reduction in terms of average waiting time from 172 to 160 giving the difference of 12, the new improved round robin when compared side by side with the developed algorithm shows an improvement in terms of average waiting time from 164 to 160 giving the difference of 4, the improved round robin with varying quantum-time when compared side by side with the developed algorithm shows an improvement in terms of average waiting time from 166 to 160 giving the difference of 6, also the improved dynamic round robin when compared side by side with the developed algorithm shows an improvement in terms of average waiting time from 164 to 160 giving the difference of 4 . Based on the result in Table 2 derived by using equation 3.3 above, it is observed that revamped mean round robin is the highest in terms of waiting time followed by improved round robin varying quantum-time, improved dynamic round robin, new improved round robin and lastly the developed algorithm with the minimal average waiting time. Hence this implies that the developed algorithm has better performance in terms of waiting time.

4.5.2 Evaluation of Average Turn-Around Time

The revamped mean round robin, when compared side by side with the developed algorithm shows an improve reduction in terms of average turnaround time from 209 to 197 giving the difference of 12, the new improved round robin when compared side by side with the developed algorithm shows an improvement in terms of average turnaround time from 201 to 197 giving the difference of 4, the improved round robin with varying quantum-time when compared side by side with the developed algorithm shows an improvement in terms of average waiting time from 203 to 197 giving the difference of 6, also the improved dynamic round robin when compared side by side with the developed algorithm shows an improvement in terms of average waiting time from 201 to 197 giving the difference of 4. Based on the result in Table 2derived by using equation 3.4 above, it is observed that revamped mean round robin is the highest in terms of waiting time followed by improved round robin varying quantum-time, improved dynamic round robin, new improved round robin and lastly the developed algorithm with the minimal average turnaround time. Hence this implies that the developed algorithm has better performance in terms of turnaround time.

4.5\.4 Evaluation of Number Context Switches

The revamped mean round robin, when compared side by side with the developed algorithm shows an improve reduction in terms of number of context switches from 14 to 7 giving the difference of 7, while the new improved round robin, improved round robin with varying quantum-time and the improved dynamic round robin when compared side by side with the developed algorithm shows little or no improvement in terms of number of context switches in few processes, but shows significant improvement when considering larger number of processes. Hence, result in Table 2derived by using equation 3.5 above, this implies that the developed algorithm has better performance in terms of number of context switches.

5. Conclusion

The experimental result was compared with the revamped mean round robin, new improved round robin, improved dynamic round robin and improved round robin with varying quantum-time. Based on the following scheduling criteria namely, waiting time, turnaround time, response time and context switch. From the results obtained, it is observed that the developed algorithm outperforms the revamped mean round robin, new improved round robin, improved dynamic round robin and improved round robin with varying quantum-time because it produces minimal average waiting time, average turnaround time and number of context switches. In conclusion, the simulation result shows that the aim and objectives of this work is achieved.

6. REFERENCE

Abdulrahim, A., Abdullahi, E. S.and Sahalu, B. J. (2014). A New Improved Round Robin (NIRR) CPU Scheduling Algorithm.International Journal of Computer Applications, (0975 – 8887).

Abu, T. and Zamani, N. (2014). The Impact of the Cloud Based M-Learning in Higher Education,International Journal of Advanced Research in Computer Science Engineering 4,pp 569-574.

Abubakar, S.E., Yusuf, S.A., Obiniyi A. and Mohammed A. (2023) Modified Round Robin with Highest Response Ratio Next CPU Scheduling Algorithm Using Dynamic Time Quantum. SuleLamido University Journal of Science & Technology Vol. 6 No. 1&2 (March 2023), pp. 87-99 https://doi,org/10.56471/slujst.v6i.363

Ajit, S., Priyanka, G. and Sahil, B. (2010). An Optimized Round Robin Scheduling Algorithm for CPU Scheduling, International Journal on Computer Science and Engineering (IJCSE), Vol. 02; No. 07, 2383-2385, pp 2382-2385.

Al-Tamimi, A.K., Jain R.and So-In, C. (2010). Dynamic resource allocation based on online traffic prediction for video streams, Internet Multimedia Services Architecture and Applications (IMSAA); IEEE 4th International Conference on, pp. 1-6 2010.

Amar, R. D., Sandipta, K. S. and Sanjay, K. S. (2015). An Optimised Round Robin CPU Scheduling Algorithm with dynamic time quantum, International Journal of Computer Science, Engineering and Information Technology (IJCSEIT), Vol. 5,No.1, February 2015.

Amit, N., Sanjay C. and Gaurav S. (2011). Policy based resource allocation in IaaS Cloud in ELSEVIER- Future Generation Computer Systems 28,94–103.

Arya, G.P., Nilay, K. and Prasad D. (2018) An Improved Round Robin CPU Scheduling Algorithm Based on Priority of Process. International Journal of Engineering &Technology, 7 (4.5) (2018)238-241.

Arjun, G. (2017). Balanced Path Selection in Cloud Computing using round robin Algorithm, |JSRCSEIT|Volume 2|Issue 4|ISSN2456-3307

Atsuo, I., Taiki, M., Minoru I. and MizanurRahman S.K. (2010). Proposal and Evaluation of Dynamin Resource Allocation Method Based on the Load Of VMs on IaaS, IEEE,,978-1-4244-8704-2/11.

Baomin, X., Chunyan, Z., Enzhao, H. and Bin H. (2011). Job scheduling algorithm based on Berger model in cloud environment in ELSEVIER - Advances in Engineering Software 42 419–425.

Behera, H.S. Rakesh, M., Jajnaseni, P., Dipanwita, T. and Subasini, S. (2011). Experimental Analysis of a New Fare-Share Scheduling Algorithm with Waited Time Slice for Real Time Systems, Journal of Global Research in Computer Science, Vol. 2, No. 2, pp. 54-60.

Bhavani, B. H., and Guruprasad, H. S. (2014). A Comparative Study on Resource Allocation Policies in Cloud Computing Environment. An international journal of advanced computer technology.

Biwas, D., Samsuddoha, A.L., Asif, R. and Ahmed M.(2023)Optimized Round Robin Scheduling Algorithm using Dynamic Time Quantum Approach in Cloud Computing Environment. I.J Intelligent Systems and Applications, 2023,1, 22-34 Published Online on February 8, 2023 by MECS Press (http://www.mecs-press.org/) DOI:10.5815/ijisa.2023.01.03.

Botterman, and Maarten. Policy Paper on IoT Future Technologies: Opening towards a New Reality.. http://www.smart-action.eu/fileadmin/smart action/publications/PolicyPaperonIoTFutureTechnologies.pdf Issue brief no. D5.2. 39

Chanjuan, L. (2013). Analysis on the Design Patterns of Cloud Computing in Mobile Learning Systems. Cloud Times, Mobile Cloud [Online] http://cloudtimes.org/mobile-cloud/.

Chenn-Jung, H., TaiGuan, C., Chen, H. M., Wang, Y.H., Chang, S.C., Ching-Yu, L. andWeng, C.H. (2013). An Adaptive Resource Management Scheme in Cloud Computing in ELSEVIER - Engineering Applications of Artificial Intelligence 26 382-389.

Daniel, O. and Eduardo, O. (2010). Article: Is Cloud Computing the Solution for Brazilian Researchers? International Journal of Computer Applications 6(8):19–23.

Dash, R. A., Sahu, k. S., and Samantra, K. S. (2015). An Optimized Round Robin Cpu Scheduling Algorithm With Dynamic Time Quantum. International Journal of Computer Science, Engineering and Information Technology (IJCSEIT).

Dogan, A. and Ozguner, F. (2002). Matching and scheduling algorithms for minimizing execution time and failure probability of applications in heterogeneous computing, IEEE Transactions on Parallel and Distributed Systems, pp. 308–323.

Dorian, M. and Bernd, F. (2011). Utility– based Resource Allocations for virtual machines in cloud computing, IEEE.

Fiad, A., Maaza, M. Z. and Bendoukha, H. (2020). Improved Version of Round Robin Scheduling Algorithm Based on Analytic Model, International Journal of Networked and Distributed Computing, Vol. **8**(**4**); December (2020), pp. 195–202. DOI: https://doi.org/10.2991/ijndc.k.200804.001; ISSN 2211-7938; eISSN 2211-7946 https://www.atlantis-press.com/journals/ijndc.

Gokilavani, M., Selvi S., and Udhayakumar, C. (2013). "A Survey on Resource Allocation and Task Scheduling Algorithms in Cloud Environment".

Guiyi, W., Athanasios, V., Zheng, Y. and Xiong, N. (2010). A game-theoretic method of resource Allocation n for cloud computing services, in Springer, J Supercomput 54: 252-269.

Gurdev, S., ShanuSood, and Amit, S. (2011). ―CM- Measurement Facets for Cloud Performance‖, IJCA, Lecturer, Computer science & Engineering, Eternal University, Baru Sahib (India), Volume 23 No.3.

Goudarzi, H. and Massoud, P. (2011). Maximizing Profit in Cloud Computing System Via Resource Allocation IEEE 31st International Conference on Distributed Computing Systems Workshops: pp,1- 6.

Hindman, B., Konwinski, A., Zaharia, M., Ghodsi, A. Joseph, D. Katz, R. Shenker, S. and Stoica.Mesos I, (2011). A platform for fine-grained resource sharing in the data center. In NSDI.

Igbal, M., Ullah, Z., Khan I.A., Aslam, S., Shaheer, H., Humayon, M., Salahudin, M. A. and Adeel, M. (2023). Optimizing Task Execution: The Impact of Dynamic Time Quantum and Priorities on Round Robin Scheduling. FutureInternet 2023, 15, 104. https//doi.org/10.3390/fi15030104

Ishwari, S. R. and Deepa, G. (2012). A Priority based Round Robin CPU Scheduling Algorithm for Real Time Systems, International Journal of Innovations in Engineering and Technology (IJIET), Vol. 1 Issue 3, pp 1-11.

Jiyin, L., Meikang, Q., Niu, J.W., Chen, Y. and Ming, C. (2010). Adaptive Resource Allocation for Premptable Jobs in Cloud Systems, IEEE International Conference on Intelligent Systems Design and Applications, pp. 31-36

Justin, Y. S., Moussa, T., and Abdallah K. (2011) Resource Planning for Parallel Processing in the Cloud, in IEEE International Conference on High Performance Computing and Communications, 978-0-7659-4538-7/11.

Kathuria, S., Singh, P. P., Tiwari, P. and Prashant, N. (2016). A Revamped Mean Round Robin (RMRR) CPU Scheduling Algorithm. International Journal of Innovative Research in Computer and Communication Engineering.

Khatri, J. (2016). An Improved Dynamic Round Robin CPU Scheduling Algorithm Based on Variant Time Quantum,IOSR Journal of Computer Engineering (IOSR-JCE) , PP 35-40.

Kochut, T., York,M. and Miller, G. (2010). Desktop Workload Study with Implications for Desktop Cloud Resource Optimization, IEEE 978-1-4244-6534-7/10 57

Kuo-Chan, H., and Kuan-Po, L. (2010). Processor Allocation policies for Reducing Resource fragmentation in Multi cluster Grid and Cloud Environments,IEEE, pp.971-976.

Lalit, K. Rajendra, S. and Praveen, S. (2011). Optimized Scheduling Algorithm, International Journal of Computer Applications, pp 106-109.

Manish, K. M. and Faizur, R. (2014). An Improved Round Robin CPU Scheduling Algorithm with Varying Time Quantum, International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol.4, No.4.

Manish, K. M. and Abdulkadir, K. (2012). An Improved Round Robin CPU Scheduling Algorithm, Journal of Global Research in Computer Science, Volume 3, No. 6, pp 64-69. ISSN: 2229-371X

Mell, P. and Grance, T. (2011). The NIST definition of Cloud Computing. NIST, Special Publication 800–145, Gaithersburg, MD.

Mythili N.P., Korde P. and Dey P. (2017) An Advanced Approach to Traditional Round Robin CPU Scheduling Algorithm to Prioritize Processes with Residual Burst Time Nearest to the Specified Time Quantum. IOP Conference Series: Materials Science and Engineering, VIT University, Vellore-632014, Tamil Nadu, India.

Nayak, D., Malla, K. S. and Debadarshini, D. (2012). Improved Round Robin Scheduling using Dynamic Time Quantum, International Journal of Computer Applications, 0975 – 8887.

Neha, A. J. (2018). An Inproved Round Robin CPU Scheduling Algorithm, Iconic Research and Engineering Journals, Mar 2018|IRE Journals|Volume 1 Issue 9| ISSN: 2456-8880.

Olofintuyi S.S., Omotehinwa T.O., Oyekanmi E.O. and Olajubu E.A. (2019) An Improved Time Varying Quantum Round Robin CPU Scheduling Algorithm. Achievers Journal of Scientific Research Vol 2, Issue 2, December 2019 – p.31 -38.

Omotehinwa O. T., Azeez I .S., and Olafunyi S.S (2019) A Simplified Improved Dynamic Round Robin (SIDRR) CPU Scheduling Algorithm. Internationa Journal of Information Processing and Communication (IJIPC) vol.7 No.2 [December 2019], pp.122-140.

Onat, G. Y., Mathews, C., Neville, F. S., Guitouni, A., Gnat, S. and Coady, Y. (2010). Dynamic Resource Allocation in Computing Clouds using distributed Multiple Criteria. Decision Analysis. IEEE.

Oyetunji, E.O. and Oluleye A. E. (2009). Performance Assessment of Some CPU Scheduling Algorithms, Research Journal of Information Technology1(1): pp 22-26, 2009.

Pallab, B., Biresh, K. and Probal, B. (2017). Mixed Round Robin (MRR) Scheduling for Real Time System International Journal of Computer Trends and Technology (IJCTT) – Volume 49 Number 3July 2017.

Pandaba, P., Prafulla, K., Behera, K.P., and Ray B.N., (2016). Modified Round Robin Algorithm for Resource Allocation in Cloud Computing, in International Conference on Computational Modeling and Security(CMS) 878-890

Patel, R. and Patel, S. (2013). Survey on Resource Allocation Strategies in Cloud Computing International Journal of Engineering Research & Technology (IJERT)

Pradhan, P., Behera, K. P., and Ray, B. N. (2016). Modified Round Robin Algorithm for Resource Allocation in Cloud Computing. Elsevier.

Prakash, G. L., Manish, P., and Inder, S. (2014). Data Encryption and Decryption Algorithms using Key Rotations for Data Security in Cloud. International Journal of Engineering and Computer Science, 3(4), 5215-5223.

Rajput, S. I. and Gupta, D. (2012). A Priority based Round Robin CPU Scheduling Algorithm for Real Time Systems. International Journal of Innovations in Engineering and Technology (IJIET).

Ritu, A.(2018). Resource Provision and Resource allocation in Cloud. International Journal of Scientific Research in Computer Science, Engineer and Information Technology, 2018| JSRCSEIT|Volume 3|Issue 3|ISSN2456-3307

Ronak, P., Sanjay P, (2013). Survey on Resource Allocation Strategies in Cloud Computing. International Journal of Engineering Research & Technology (IJERT) Vol. 2 Issue 2, February- 2013 ISSN.

Sakshi, C., Sharma S., Kautish S., Alsallami S., Khalil E.M. and Mohamed A. (2022) A New Median-Average Round Robin Scheduling Algorithm: An Optimal Approach for Reducing Turn Around and Waiting Time, Alexandria Engineering Journal. 61,10527-10538.

Sheth, A. and Ranabahu, A. (2010). Semantic Modeling for Cloud Computing, ‖ IEEE Computer Society — Semantics & Services, 2010 Part I.

Singh,A., Goyal, P. and Batra, S. (2010). An Optimized Round Robin Scheduling Algorithm for CPU Scheduling, International Journal of Computer and Electrical Engineering, Vol. 2, No. 7, pp. 2383-2385.

Sundarraj, B. and Venkatesan K.G. (2011). A Stochastic Model to Investigate Data centers for Better Performance.

Vinothina, V., Sridaran R. and Padmavathi, G. (2012). Resource Allocation Strategies in Cloud Computing. International Journal of Advanced Computer Science and Applications [IJACSA], Vol. 3, No.6. ISSN: 2158-107X (Print), DOI: 10.14569/issn.2156-5570.

Yaashuwanth, C. and Ramesh. R. (2015). Design of Real Time Scheduler Simulator and Development of Modified Round Robin Architecture for Real Time System, International Journal of Computer and Electrical Engineering, Vol. 10, No. 3, 2010, pp. 43-47.

Zhang, Q., Cheng, L. and Boutaba, R. (2010). Cloud Computing: state of the-art and research challenges. Springer, pp. 7-18.