Original Research Article

AN ENHANCED LOAD BALANCING ALGORITHM FOR CLOUD ENTERPRISE RESOURCE PLANNING (ERP) DATA IN A MULTI-CLOUD ENVIRONMENT

ABSTRACT: Businesses and individuals have seen the need to adopt the cloud and mulit-cloud environment for their businesses and storage of data. The load balancing concerns especially in the multi-cloud environment was investigated and a new algorithm proposed. In this research, a proposed new load balancing algorithm is presented and compared with the Round Robin (RR) and Weighted Round Robin (WRR) algorithms. The proposed scheduling algorithm considered several Cloud ERP Data chunks to analyse the data transmission rate or throughput, the transmission delay, data loss and the Cloud ERP Data drop ratio.

KEYWORDS: Load Balancing, Cloud Computing, ERP, LBaaS

INTRODUCTION

After decades of research, the Cloud Computing concept saw the light of day using and harnessing the potentials of existing technologies like grid computing, peer-to-peer technology, parallel computing, distributing computing and virtualization (Joshi and Kumari, 2017).

Information Technology and the internet has connected individuals and businesses across the world and thereby eliminating geographical barriers in communication and business transactions. The information Technology industry has seen so much technological improvements and innovation since the inception of the World Wide Web (www). The World Wide Web (www) is a system that houses documents and other web resources usually interlinked and accessed through a Uniform Resource Locator (URL) and a browser. The World Wide Web (WWW) was invented by a British scientist called Tim Berners-Lee in the year 1989 and it was made available to the public on April 30th, 1993 according to literature.

The web contents are stored in a Web Server and accessible through the browser. A Web Server is a computer or software that hosts the websites and provide services to users of the world wide web or the internet. The Web server usually respond to a request by the client by sending the file requested by the client. It does so by revoking a script and communicating with a database.

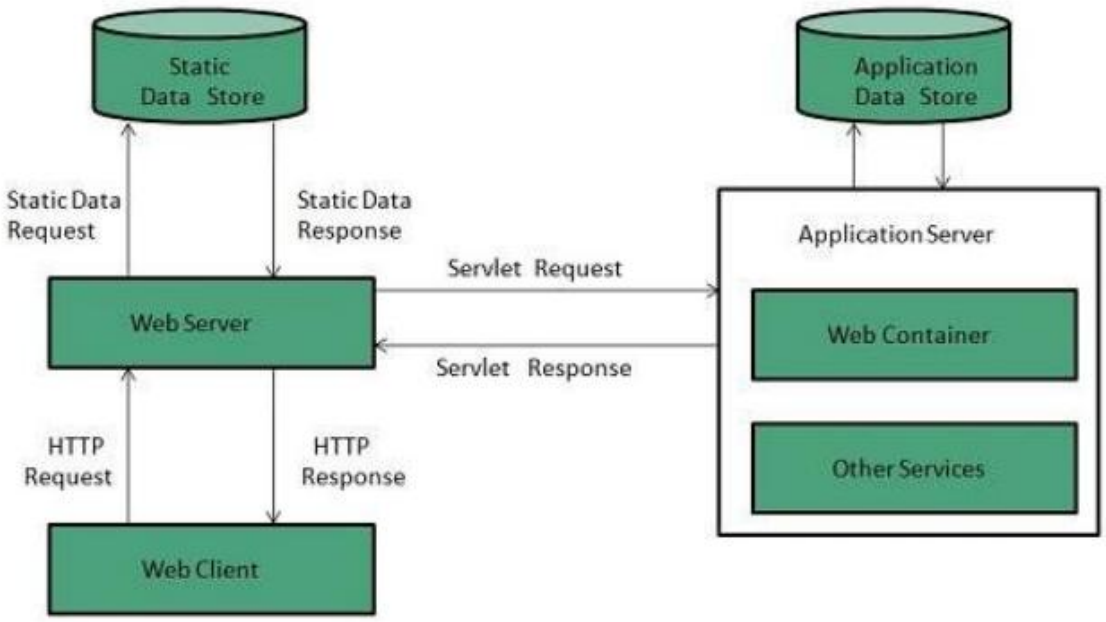

Figure 1: Web Server

The web server in the world wide web (www) contributed immensely towards the development and rise of cloud computing. Even though web servers like Apache and NGINX still play significant roles in assisting cloud service providers to deliver services on demand, the cloud services are actually being handled by Special computing systems called Cloud Servers.

CLOUD SERVER

Cloud server is a virtual server in the cloud computing environment that helps in providing cloud computing related services to clients on demand.

The Cloud server has a device data logic, a data storage service, database engine, a system to do data analysis as well as a system for data visualization. It has sensor and node device management feature that connects and communicate with a web server through a dedicated Uniform Resource Locator (URL). Among others, the cloud server helps in managing the load generated by clients of the cloud computing service providers. Web servers like Apache and NGINX being part of the Cloud server architecture has the ability to handle the cloud load balancing requirement in the cloud environment.

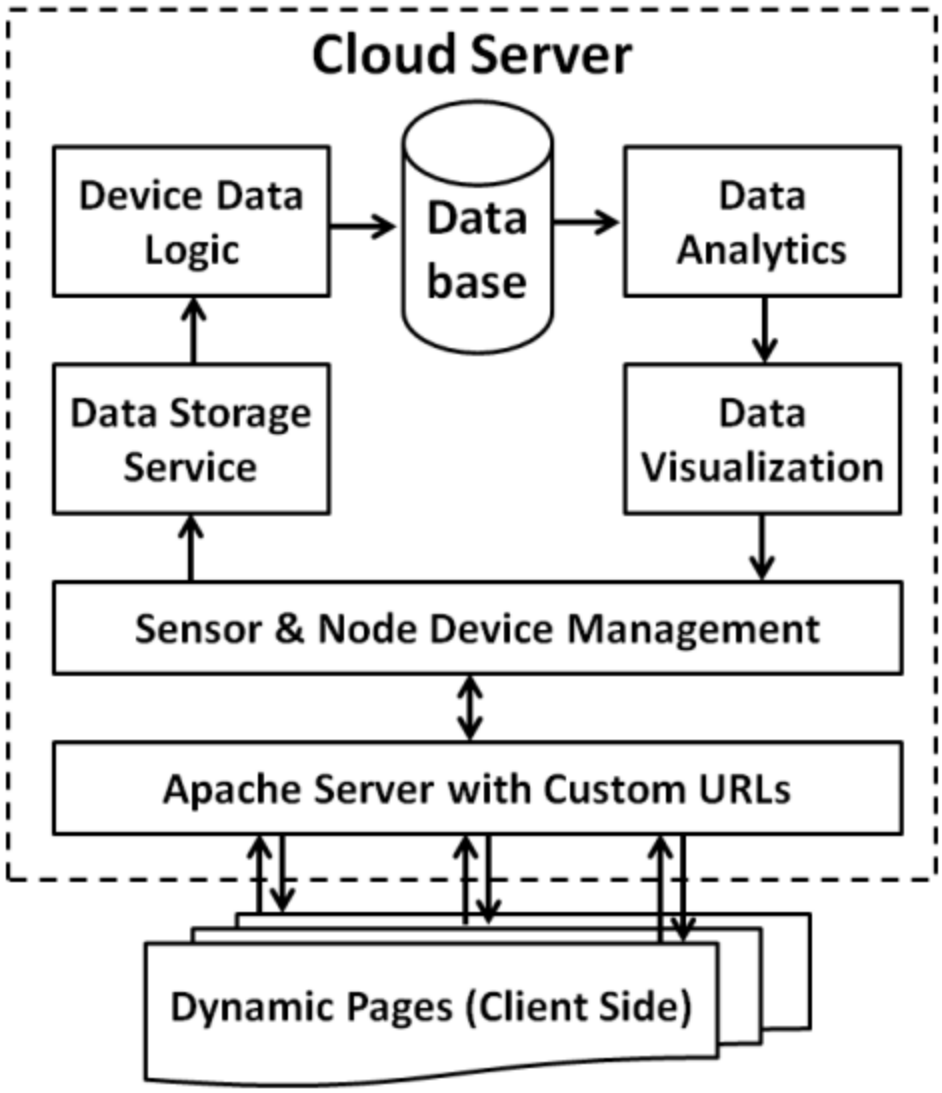

Figure 2: Cloud Server Architecture

LOAD BALANCING IN CLOUD

In the cloud computing environment, resource allocation and provisioning is very essential towards delivering quality services. Over the years, internet usage has increased tremendously since it has become a key and major tool in communication and business transactions. These significant increase in internet usage has resulted in congestions in the networks, web server overloading and delayed response which often results in crashing of servers leading to poor services in some cases. Most business systems across the world are using a Cluster-Based Web Servers (CBWS) where several number of web servers are clustered. The Cluster-Based Web Server (CBWS) is a powerful virtual server with a challenge of how to distribute clients requests properly. The concept of load balancing is crucial to help distribute the load from clients among the servers and other network devices in order to effectively manage resources and network congestion as well as prevent overloading of any given server or a network

device. An effective mechanism of managing issues regarding cloud-based solutions is load balancing since it focuses on resources provisioning and resource allocation in the cloud environment.

RESOURCE ALLOCATION IN CLOUD

Patel (2013) defines as the process of assigning cloud computing resources that are available to cloud applications or services used by clients. Poor management of cloud computing resources could result in unsatisfactory services to the clients and businesses using the cloud.

The resource allocation in the cloud is usually done using a Resource Allocation Strategy (RAS). The resource allocation strategy (RAS) is a guideline responsible for integrating cloud computing activities to utilise and allocate resources in the cloud computing environment. The Resource Allocation Strategy (RAS) is very essential in managing the cloud's resources by avoiding resource contention, avoiding resource fragmentation as well as avoiding scarcity of resources. According to Minarolli and Freisleben (2011), these are key areas needed to effectively manage resources in the cloud computing environment. Over provisioning and under provisioning of resources in order to ensure availability of resources at all times are also considered under the Resource Allocation Strategy (RAS) (Pradhan et al, 2016).

LOAD BALANCING ALGORITHMS

Since the concept of Cloud Computing became very popular among businesses and researchers across the globe, there has been alot research interest in how cloud computing resources are provisioned to ensure the effective use of resources. How the cloud handle resource allocation or how load balancing is handled in the cloud computing environment has seen several load balancing algorithms.

Individuals and businesses do not want to subscribe to a cloud computing service that has challenges regarding overloads and delay in service delivery.

ROUND ROBIN ALGORITHM

The most widely used and oldest algorithm for load balancing in the cloud computing environment according to Pradhan et al (2016) is the Round Robin Algorithm (RR). The Round Robin Algorithm is designed especially for time-sharing systems like the cloud computing environment.

One of the key and essential parameters needed for the application of the Round Robin Algorithm is the

Time-Slice also known as Quantum. For the purposes of our research, we will refer the Time-Slice or Quantum as *Q*. The value for *Q* is a small unit of time that will be defined in order to execute the Round Robin Algorithm.

In the round Robin Algorithm concept, the CPU time is shared among all tasks that are scheduled on the ready queue. Each task that is submitted for execution is allocated a time-slice or Quantum Q which is a very decisive characteristic in the concept of Round Robin Algorithm. Despite the importance and decisive nature of the time Quantum Q, several researchers have proposed Round Robin algorithms that

have static time Quantum Q. A static time Quantum according to Tani and Amrani (2017) does not offer the best of solutions always. Tani and Amrani (2017) proposed a dynamic time Quantum as a more better alternative such that the dynamic Quantum will adapt the CPU time slices as and when executions are done in the ready queue. For the purposes of this research we will represent the Dynamic time Quantum as Q_d. The Dynamic time Quantum Q_d is calculated as follows:

$$Q_d = {}^{C_t}/_{N_j} \qquad (1)$$

Where C_t is the total CPU Time and N_j is the total number of jobs. In the round robin scheduling algorithm, the resource usually in the form of CPU is assigned to the process on First-Come-First-Served (FCFS) basis for a fixed amount of time called the Quantum Q. After the Q expires, the process that is running is preempted and sent to the ready queue. The CPU is then assigned to the next process always in preemptive manner. The round robin algorithm leads to starvation of some processes in a situation where the burst time is larger and requires several repetitions before completing the cycle. The performance of this algorithm is largely dependent on the Quantum Q and do not have option to set priorities for any of the processes. When the Q is increasing or it's large, the Round Robin algorithm turns to be FCFS algorithm.

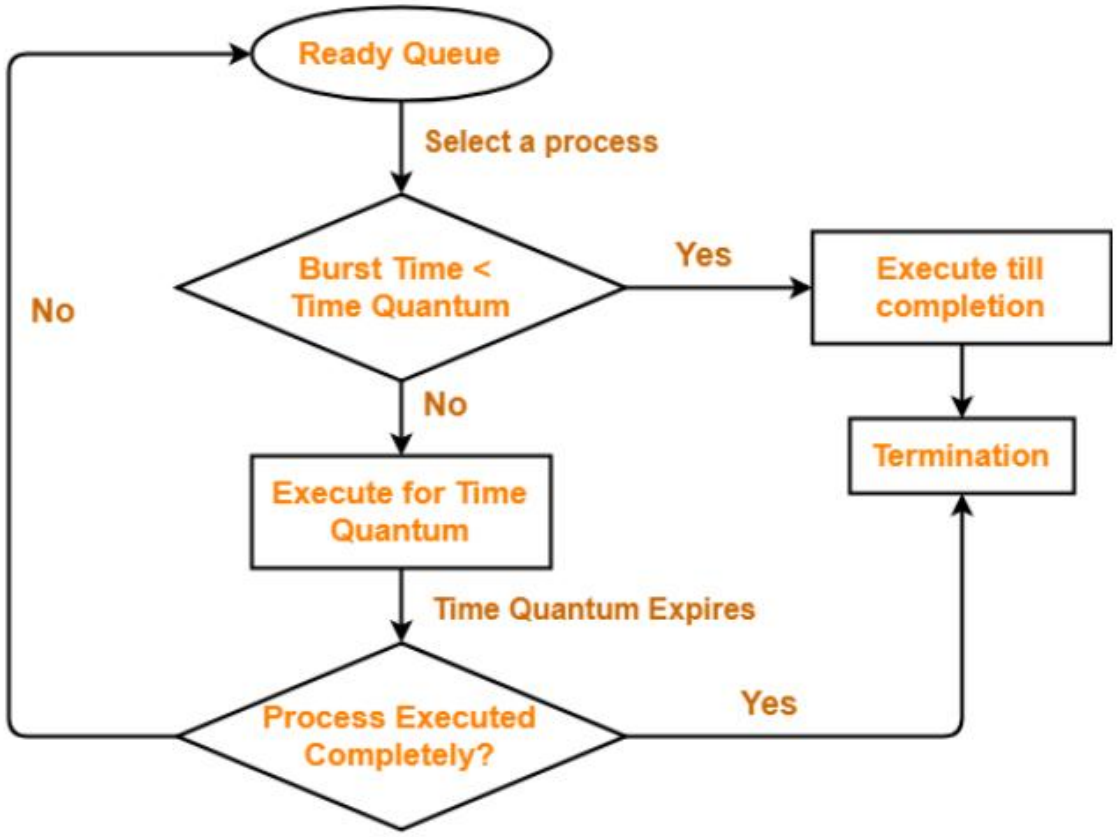

Figure 3: Round Robbin Scheduling

WEIGHTED ROUND ROBIN ALGORITHM

The Weighted Round Robin (WRR) Algorithm is one of the most used algorithm for scheduling largely due to it's ability to handle computational overheads and it's simplicity (Saidu et al, 2014). Despite efficiently handling computational overheads, the Weighted Round Robin Algorithm, according to Saidu et al (2014) is affected by bursty traffic leading to performance degradation. They explained that the bursty traffic in the Weighted Round Robin Algorithm is due to it's static weights used to determine the transmission of packets. The Weighted Round Robin (WRR) algorithm is reported to be using static weights to differentiate Quality of Service (QoS) requirements for classes within the various services in the scheduling processes (Saidu et al, 2014). Each process in a a queue is assigned a fixed weight. The fixed weights indicates the number of packets to be executed in cycle.

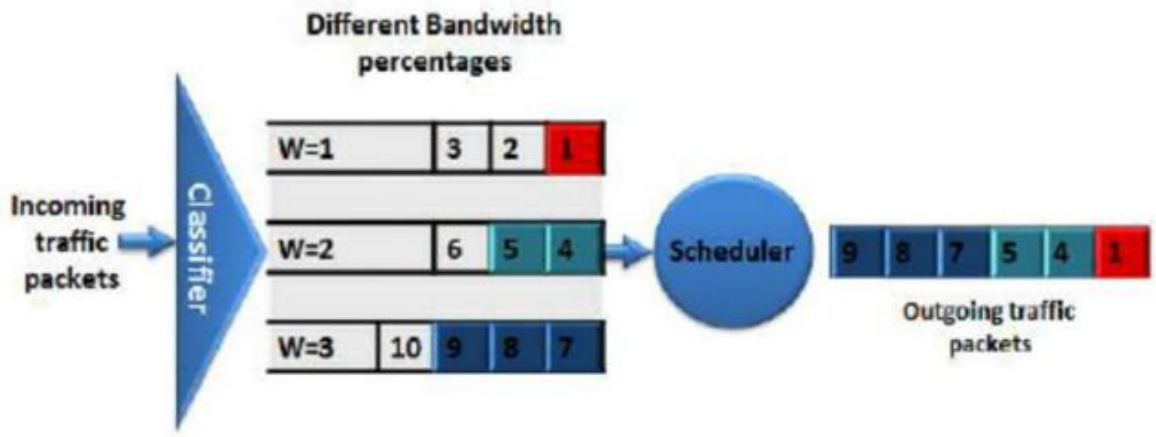

Figure 4: Weighted Round Robin Scheduler

MODIFIED WEIGHTED ROUND ROBIN ALGORITHM

The Weighted Round Robin Algorithm discussed above has the weight of each queue calculated based on the Quality of Service required for each process or server and are largely based on traffic priority (Khan et al, 2010). The Weighted Round Robin algorithm is a fair and ideal algorithm in situations there the weights assigned to the servers or queues are equal as well as requires equal packets for transmission. However, the static weights based on priority levels often lead growth of the number of queues and packet delays and at times packets loss (Khan et al, 2010). The lower priority classes or servers suffers imposed delays.

The delay in the lower priority servers or classes lead to Mardini and Alfool (2011) proposing a Modified

Weighted Round Robin (MWRR) Algorithm. In this algorithm, the weight for the queues are calculated based on priority coupled with the number of noempty queues and thus allows a certain number of packets to be transmitted in a single transmission cycle.

To increase the transmission cycle in order to accommodate all queues, the weight counter is multiplied by an integer. The Modified Weighted Round Robin (MWRR) reduces the average delay and increases the throughput through the increasing of the transmission cycle. However, the challenge is that the multiplier used is static. And this when not properly chosen may lead to decrease in throughput and increase delays.

LOAD BALANCING FOR MULTI-CLOUD

Businesses in recent times across the globe are gradually adopting the multi-cloud environment to develop their business applications. Developing applications or having business solutions that runs on multi-cloud comes with it's advantage and eliminate the posibility of vendor lock-in as well as improve security especially when the clients has absolute control of their own data or cloud solution.

Businesses are gradually changing how they develop their applications in support of their business transformation, digital transformation and the growth of the information technology industry. Lushasz et al (2017) states that load balancing is an integral part of a software that serve requests of multiple and concurrent computing resources with the aim to maximise the usage of resources and minimise response time. There are several benefits in adopting the multi-cloud load balancing for applications that can share data across multiple cloud service providers in the case of this research work. Notable among some of the benefits are:

1) Failover management: The multi-cloud environment offers businesses to use the other clouds as a backup services in case one cloud goes down. In the multi-cloud environment, a set of identical interfaces and replicated data is provided to all the cloud service providers in the multi-cloud environment. If one cloud goes down, a request from a client is routed to another cloud in the multi-cloud environment.

2) Effective Application Migration: In the multi-cloud environment, the application is provisioned on all clouds in the multi-cloud environment. This makes it easier to migrate data or the application to another cloud service provider especially when you want to adopt one cloud as the primary cloud in the multi-cloud environment. This can also be achieved when a weight is placed on each cloud n the multi-cloud environment.

3) Effective handling of Cloud Bursting: The multi-cloud environment gives room for the dynamic addition and removal of either a private cloud or a public cloud. This feature is done by the mulit-cloud load balancer profile and a proportion of all the requests going forward goes to the newly provisioned cloud in the case of addition of a cloud. Resources are relocated and provisioned to other clouds based on their profiles in the case of removal of a cloud.

4) Internet Scalability: In the multi-cloud environment, the clouds are likely to reside in different Geographical locations with requests coming different geographical locations or distances as well. The multi-cloud environment based on the load balancing algorithm can direct incoming requests or traffic to the closest cloud in order to achieve the lowest latency for the client. The routing is done typically with the shortest geographic distance for the clouds in the multi-cloud environment. This makes the multi-cloud environment very effective in terms of internet scalability.

5) Fault Tolerance: The load balancing in multi-cloud has the ability to detect and redirect failed clouds or components to another cloud. This is done until the failed cloud or component is restored back to service.

THE PROPOSED LOAD BALANCING SCHEME

In this research work, we are proposing a modification of the Weighted Round Robin (WRR) Algorithm that will be applicable in a multi-cloud environment. The Weighted Round Robin algorithm is an improved modification of the famous and widely used Round Robin (RR) algorithm.

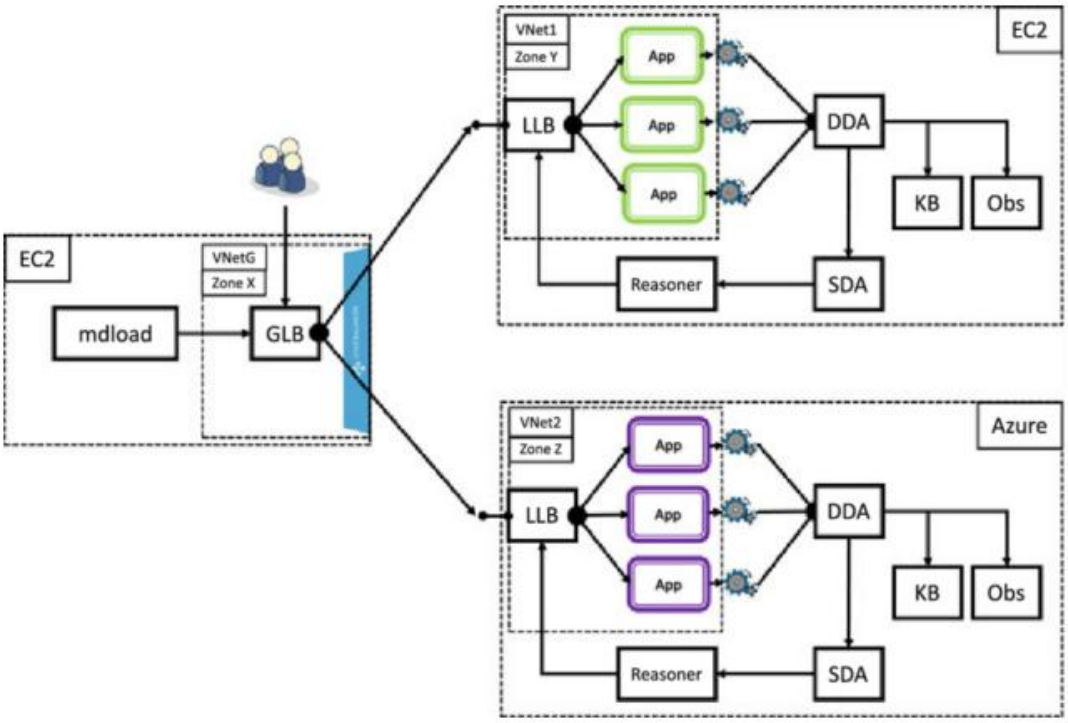

Figure 5: Multi-Cloud Load Balancer

The modification is largely on the static weights being assigned to each process in the queue which are determined based on their priorities and minimum bandwidth required.

In our quest to propose a modified version of the Weighted Round Robin algorithm that is applicable to a multi-cloud environment, we introduce the following parameters:

CT_i = The *ith* Cloud Traffic Minimum Reserved Rate

W_i = Static Weight of the *ith*-Cloud.

C_i = The *ith* Cloud in the multi-cloud environment.

n = the number of clouds in the ,multi-cloud environment.

We calculate W_i as follows:

$$W_i = \frac{CT_i}{\sum_{i=1}^{n} CT_i} \quad (2)$$

In this research, we are proposing that the Wi should be calculated dynamically in order to address the shortfalls of the Weighted Round Robin (WRR) algorithm presented above. The traffic characteristics of each cloud is taken into consideration at the beginning of each reset counter. These are proposed to keep track of the variations in the clouds capabilities in order to reduce delays and loss of packets and thereby improving throughput.

In the proposed algorithm, the Cloud ERP data be shared in the multi-cloud environment has the data variance calculated in the following equation:

$$D_v = \frac{1}{n}\sum_{i}^{n}(D_{i,r} - D_r)^2 \quad (3)$$

Where D_r is the mean from round *r*. The value for D_r is calculated from the formula below:

$$D_r = \frac{1}{n}\sum_{i}^{n} D_{i,r} \quad (4)$$

In attempting to derive the dynamic coefficient of the *ith*-cloud, we calculated the root mean square R_{ms} at round *r* as:

$$R_{ms} = \sqrt{D_v} \quad (5)$$

The Dynamic Coefficient Dc is calculated as shown in the following equation:

$$D_c = \left\lceil \frac{D_{i,r}}{D_v + 1} \right\rceil \quad (6)$$

The dynamic weight W_d of each cloud is calculated using the dynamic coefficient at round *r* and the weight of the *ith*-cloud *Wi* as indicated in the equation below:

$$W_d = D_c + W_i \quad (7)$$

The following algorithm is used in calculating the Dynamic Weight of the various clouds in the multi-cloud environment:

Algorithm 1. Algorithm to calculate Dynamic Weight

Step 1: let *r* be the round at a given instance of the ERP Data D_i assigned to the *ith*-cloud where $1 \leq i \leq n$.

Step 2: Calculate cloud ERP Data variance D_v .

Step 3: Determine the root mean square R_{ms} errors at round r.

Step 4: Calculate the dynamic coefficient D_c of the *ith*-cloud at round *r*

Step 5: Calculate the Dynamic weight W_d of the *ith*-cloud in round *r*

From the above algorithm, it can be seen that the larger the value of Wd the higher the counter reset time and this will ensure all the data are transmitted with enough reset time. Considering the fact that it is possible to have large values for the weights, our algorithm is adopting a suitable constant proposed by Tasaka and Ishibashi (2002) to reduce the weights in order to avoid errors of quantization from large divisors.

HOW THE PROPOSED ALGORITHM SOLVES DATA LOSS AND DELAY PROBLEM

Let's consider the following:

$C_{i,r}$ = Clouds packets in the round *r*.

Let the weight $W_i = 2$, weight counters of the various clouds as computed above be:

Weight Counter of $C_{1,1} = 6$

Weight Counter of $C_{2,1} = 4$

Weight Counter of $C_{3,1} = 2$

Weight Counter of $C_{4,1} = 2$

Let the number of Cloud ERP Data chunk in each cloud in the multi-cloud environment be:

$C_{1,1}$ = 60,50,20,30,5,10

$C_{2,1}$ = 25,20,35,7,12

$C_{3,1}$ = 15,30,5,10

$C_{4,1}$ = 5,10

In the proposed algorithm we combined both the static weight Wi and our proposed Dynamic weight W_d to effectively schedule the Cloud ERP Data in the multi-cloud environment. With the various dynamic weights W_d assigned to each cloud based on their data, it means the cloud ERP

Data chunk will be transmitted accordingly as follows:

Cloud ERP Data before Transmission:

Weight Counters *WC* = 6 : 4 : 2 : 2

$C_{1,1}$ = 60,50,20,30,5,10

$C_{2,1}$ = 25, 20,35,7,12

$C_{3,1}$ = 15, 30,5,10

C_4 = 5, 10

After the first round the weight counter *WC* is decreased based on the transmitted data and the data transmission continues as follows with a static weight $W_i = 1$:

Cloud ERP Data after first Transmission:

Weight Counters WC = 0 : 1 : 2 : 0

Static Weight $W_i = 1$

$C_{1,1}$ = all transmitted

Number of packets for C_1 remaining = 0

$C_{2,1} = 12$

Number of packets for C_2 remaining = 1

$C_{3,1} = 5,30$

Number of packets for C_3 remaining = 2

C_4 = all transmitted

Number of packets for C_4 remaining = 0

It can be seen from above that all Cloud ERP data chunks in cloud $C_{1,1}$ and cloud $C_{4,1}$ are all transmitted. Only three (3) Cloud ERP Data chunk remained at static weight $W_i = 1$.

The remaining data chunk are one(1) and two(2) for clouds $C_{2,1}$ and $C_{3,1}$ respectively and it is subjected to another transmission cycle until static weight $W_i = 0$ as shown below:

Cloud ERP Data after second Transmission:

Weight Counters WC = 0 : 0 : 0 : 0

Static Weight $W_i = 0$

$C_{1,1}$ = all transmitted

Number of packets for C_1 remaining = 0

$C_{2,1}$ = all transmitted

Number of packets for C_2 remaining = 0

$C_{3,1}$ = all transmitted

Number of packets for C_3 remaining = 0

C_4 = all transmitted

Number of packets for C_4 remaining = 0

The Weight Counters W C and the static weights W_i of the various clouds in the multi-cloud environment all reached zero (0) signifying the end of the cycle. As demonstrated above, at the end of the cycle, all

Cloud ERP Data chunk have been transmitted without delays or data loss problem. The proposed algorithm has proven to be very flexible and adjustable based on the chunks of Cloud ERP Data assigned to each cloud in the multi-cloud environment.

RESULTS AND DISCUSSIONS

The outcome of the research findings present analysis of the proposed algorithm as compared with the Weighted Round Robin (WRR) Algorithm. The research work adopted a model of the HyperText Transfer Protocol (HTTP) and the File Transfer Protocol (FTP) as part of the transmission process. Both protocols were modelled at a mean of 0.0277 seconds(s) and at a constant packet size of 150 Bytes (B).

In the multi-cloud environment, the research work used the HTTP to transmit the Cloud ERP Data. Each HTTP session has a number of packet calls with a number of packets. The research work considered components of the proposed algorithm identified as a challenge. Notably among them are the delay in cloud ERP Data transmission, the throughput and data loss.

SIMULATION PARAMETERS

Throughput: We calculated the Throughput when a Cloud ERP Data is transmitted per unit time in kilobits per second as indicated in the formula below:

$$T = \frac{\sum_{i=0}^{n} C_d}{S_t} \qquad (8)$$

Where T is the throughput, C_d is the cloud ERP Data transmitted to each cloud in the multi-cloud environment. S_t is the total time for simulation.

Transmission Delay: The delay of transmitting the cloud ERP data in the multi-cloud environment was also considered in comparing the performance of the proposed algorithm and that of the Weighted Round Robin (WRR). The research defined the delay as the time in milliseconds between the departure of the cloud ERP Data and it's arrival in the multi-cloud environment.

Let t_{di} and t_{ai} be the average time of departure and time of arrival of the cloud ERP Data in the multi-cloud environment respectively.

The delay in the transmission D_t is calculated by using the formula below:

$$T = \frac{\sum_{i=0}^{n}(t_{di} - t_{ai})}{n} \qquad (9)$$

Where n is the number of clouds in the mulit-cloud environment that are scheduled to receive the cloud ERP Data.

Packet Loss: The packet loss in the transmission of the cloud ERP Data was also taking into consideration in comparing the proposed algorithm with the Weighted Round Robin (WRR) algorithm. The packet loss

is considered as the percentage of cloud ERP Data dropped as against the percentage of Cloud ERP transmitted successfully.

Let the percentage of cloud ERP Data dropped to be P_{di} and the percentage of cloud ERP Data transmitted successfully as P_{si} . The total packet loss is calculated using the formula shown in the following equation:

$$P_{loss} = \frac{\sum_{i=0}^{m} P_{di}}{\sum_{i=0}^{n} P_{si}} \qquad (10)$$

Where *m* is the number of clouds that have dropped ERP Data.

Cloud ERP Data Drop Ratio: In the quest to examine how effective the proposed algorithm has performed, the research calculated the Packet Drop Ratio by using the cummulative number of dropped Cloud ERP Data $D_{dropped}$ and the cummulative number of generated cloud ERP Data D_{gen} as follows:

$$D_{ratio} = \frac{D_{dropped}}{D_{gen}} \qquad (11)$$

CLOUD ERP DATA THROUGHPUT ANALYSIS.

The figure 6 below shows the average Cloud ERP Data Throughput analysis for the Round Robin (RR), Weighted Round Robin (WRR) algorithms compared to the proposed algorithms. It can be seen from the simulation results that the proposed algorithm performed below the Round Robin and the Weighted Round Robin Algorithms using Cloud ERP Data chunks below 25 chunks to 150 chunks. From the simulation results, the proposed Algorithm outperformed both the Round Robin Algorithm and the Weighted Round Robin algorithms when the cloud ERP Data chunks are more than 150. It is clear from the simulation results that the proposed algorithm performs better than the Round Robin (RR) and the Weighted Round Robin (WRR) in a high traffic Cloud ERP Data load or when the ERP Data is huge.

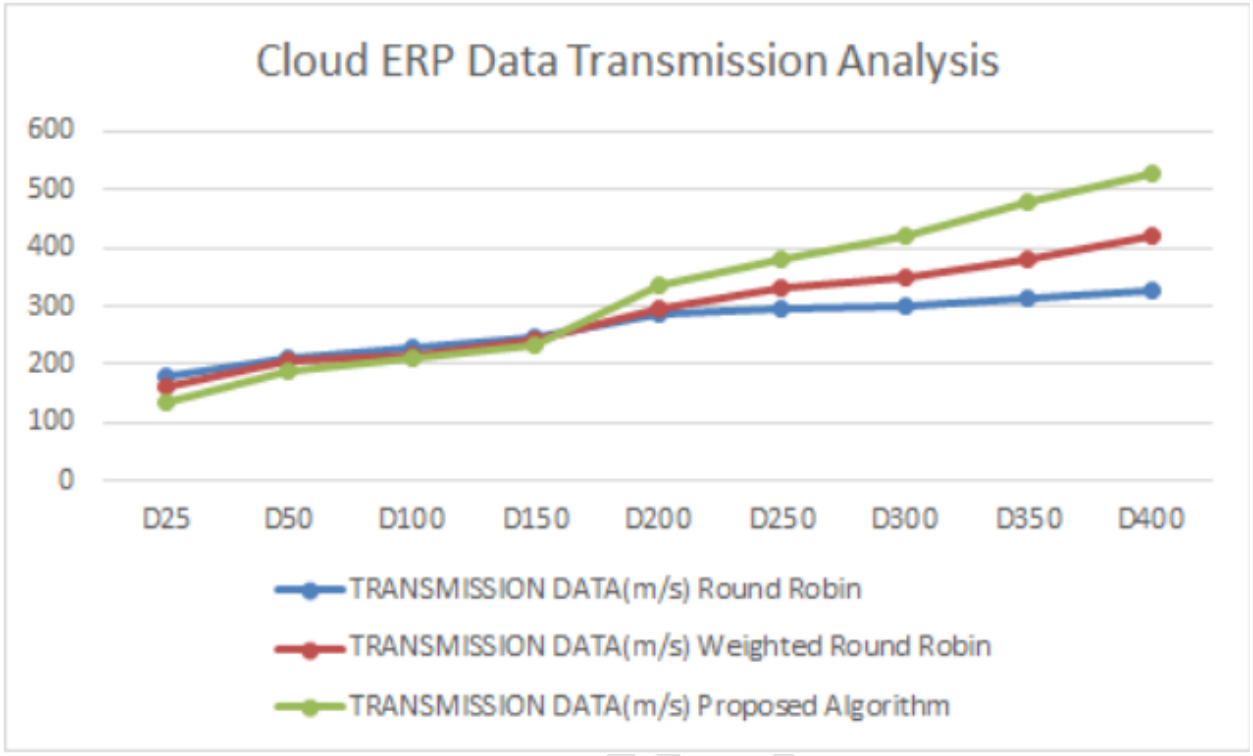

Figure 6: Cloud ERP Data Transmission

TRANSMISSION DELAY ANALYSIS

The transmission delay analysis of the Round Robin, Weighted Round Robin and the proposed Algorithm is presented in the figure 7 below.

The outcome of the simulation indicates the proposed algorithm performed similarly compared to the Round Robin and Weighted Round Robin when the Cloud ERP Data chunks are small. However, the transmission delay of the Round Robin and the Weighted Round Robin increases significantly when the Cloud ERP Data chunk increases. The proposed algorithm registers the lowest transmission delay when the cloud ERP Data chunks increases and hence outperforms both the Round Robin (RR) and the Weighted Round Robin (WRR) algorithms.

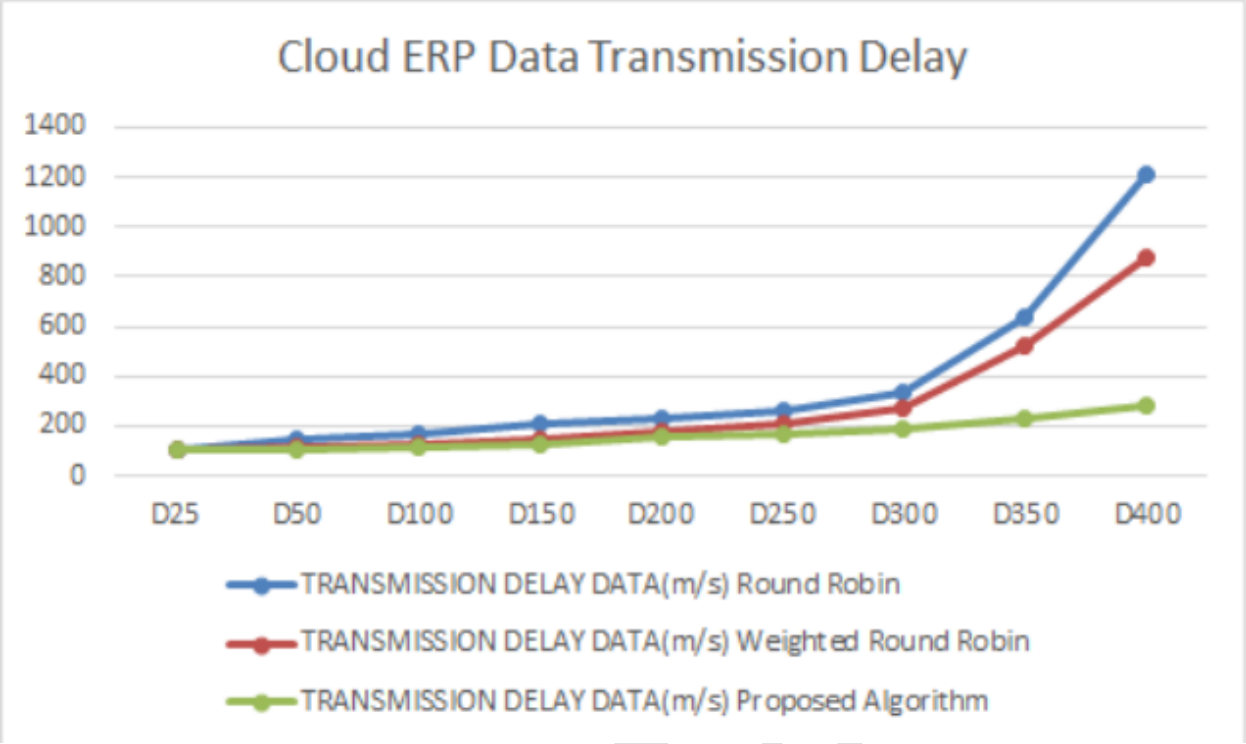

Figure 7: Cloud ERP Data Transmission delay

CLOUD ERP DATA LOSS ANALYSIS

Depending on the cloud ERP Data size, the research computed and simulated the average Cloud ERP Data loss when various sizes of the Cloud ERP Data is used and an average is presented in the graph. Despite the attention being on the performance of the algorithms, other factors might have contributed in producing the simulation and hence were treated and held constant since all the three algorithms were handled under same conditions and state. As indicated in the figure 8 below, round robin (RR) and Weighted Round Robin (WRR) algorithms loss very small amount of Cloud ERP data when the data chunks are relatively smaller in size but drops significant amount of cloud ERP Data as the Cloud ERP Data chunks grows. From the simulation results presented in the graph below, the proposed algorithm has performed far better in handling data loss in the multi-cloud environment compared with the Round Robin (RR) and Weighted Round Robin (WRR) Algorithms. The proposed algorithm registered zero (0) cloud ERP Data loss from one(1) megabyte to over 25 mb of the Cloud ERP Data chunk whiles the other two algorithms recorded some losses.

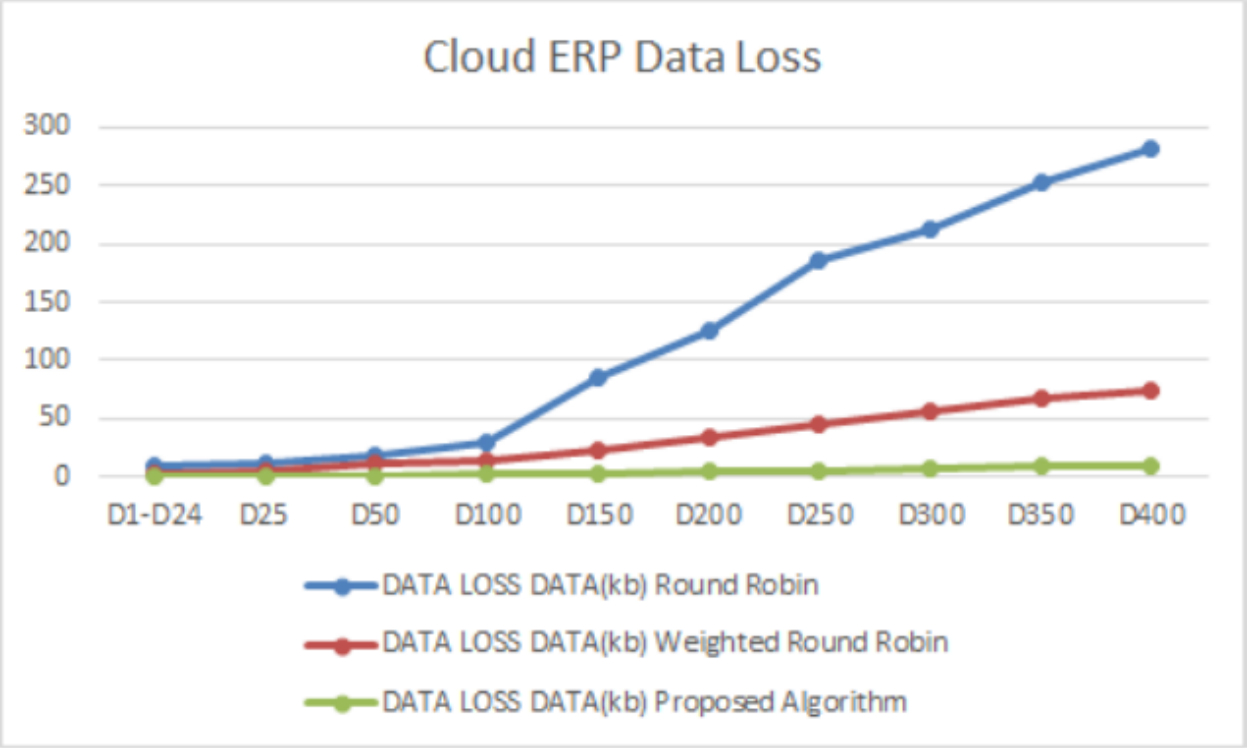

Figure 8: Cloud ERP Data Loss

CLOUD ERP DATA DROP RATIO ANALYSIS

The research work calculated the Cloud ERP Data drop ratio by considering the amount of Cloud ERP Data chunk generated against the amount of data loss during the simulation and the results presented in the figure 9 below. As can be seen the figure 9, the proposed algorithm has the lowest data drop ratio outperforming the Round Robin (RR) and the Weighted Round Robin (WRR) algorithms. Considering about 1849 Cloud ERP Data chunk generated, the Round Robin (RR) dropped about 1212 as the data chunk size increases with a drop ratio of 0.655 whiles the Weighted Round Robin (WRR) recorded about 0.178 with a total of 330 Cloud ERP Data loss. The proposed Algorithm lost only 39 Cloud ERP Data chunks out of 1849 with data drop ratio of 0.021 outperforming the other two algorithms.

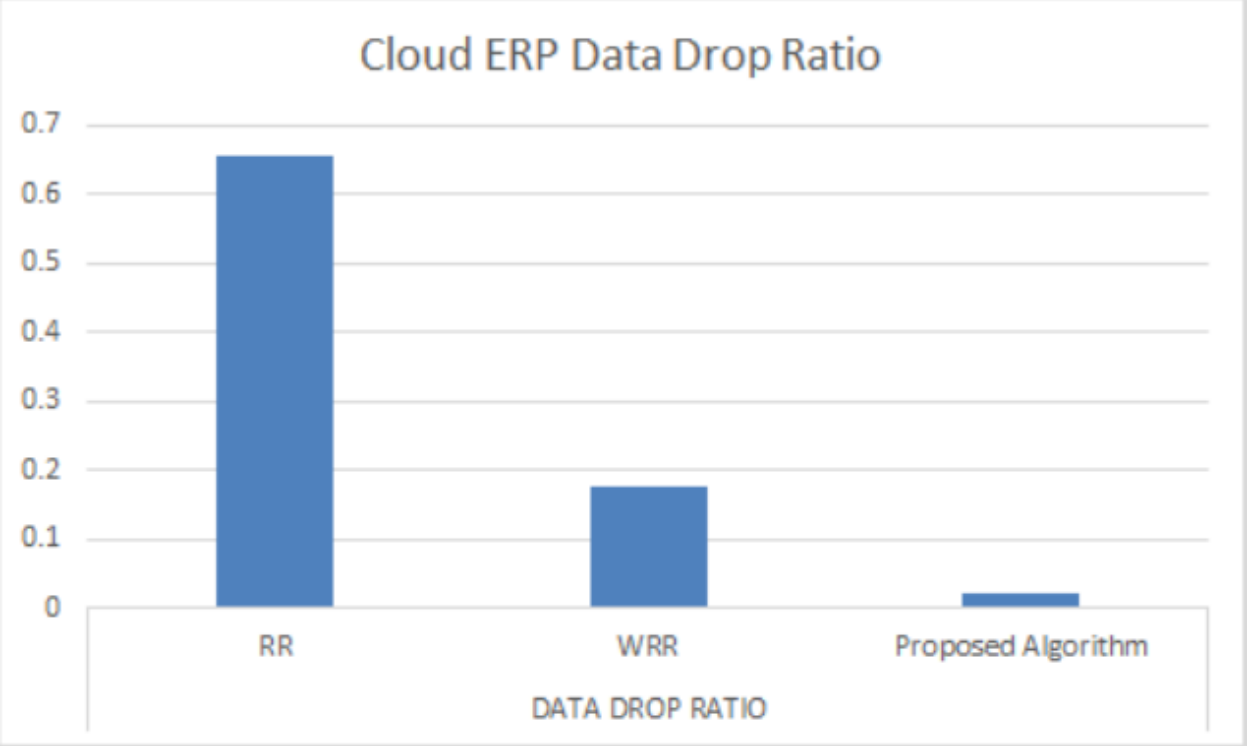

Figure 9: Data Drop Ratio Analysis

CONCLUSIONS

Businesses and individuals have seen the need to adopt the cloud and mulit-cloud environment for their businesses and storage of data. The load balancing concerns especially in the multi-cloud environment was investigated and a new algorithm proposed. The Cloud data transmission rate, transmission delays and data loss were the key concerns under investigation. In this research, a proposed new load balancing algorithm is presented and compared with the Round Robin (RR) and Weighted Round Robin (WRR) algorithms. The proposed scheduling algorithm considered several Cloud ERP Data chunks to analyse the data transmission rate or throughput, the transmission delay, data loss and the Cloud ERP Data drop ratio.

The introduction of the dynamic weight calculated from the cloud ERP Data variation in addressing the

Cloud ERP Data has contributed immensely towards improving the throughput, transmission delay and data loss. Several simulations were conducted and the results indicates that the proposed load balancing algorithm based on the Weighted Round Robin (WRR) outperforms the Round Robin (RR) and the Weighted Round Robin (WRR) algorithms. The effective implementation of our proposed algorithm in the multi-cloud environment will increase the throughput significantly as well as significantly reduce the possibility of data loss during transmission in the multi-cloud environment.

REFERENCES

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

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

Hicham, G.T. and Chaker E. (2017). Smarter Round Robin Scheduling Algorithm for Cloud Computing and Big Data. 2017. hal-01443713

Savitha P., Geetha, J., Reddy.(2013). A Review Work On Task Scheduling In Cloud Computing Using Genetic Algorithm, International Journal of Scientific and Technology Research, Volume 2, Issue 8.

Hicham,G.T., Chaker,E.L.(2016). Cloud Computing CPU Allocation and Scheduling Algorithms using CloudSim Simulator, International Journal of Electrical and Computer Engineering, Vol 6, No 4.

Rajveer,K. and Supriya,K. (2014). Analysis of Job Scheduling Algorithms in Cloud Computing, International Journal of Computer Trends and Technology, volume 9, number 7.

N. Zanoon,N. and Rawshdeh,D. (2015). STASR: A New Task Scheduling Algorithm For Cloud Environment, Network Protocols and Algorithms, Vol. 7, No. 2.

Santhosh, B., Harshitha, A., PrachiKaneria, D. and Manjaiah,H.(2014). Comparative Study of Workflow Scheduling Algorithms in Cloud Computing, International Journal of Innovative Research in Computer and Communication Engineering, Vol.2, Special Issue 5.

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, Vol. 1, Issue 3.

Mohita, G. and Savita,G.(2013). Optimized Processor Scheduling Algorithms using Genetic Algorithm Approach, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 6.

Abdulrazaq,A., Saleh,E. A., Junaidu, B. Sahalu. (2014).A New Improved Round Robin (NIRR) CPU Scheduling Algorithm, International Journal of Computer Applications, Volume 90 – No 4.

Siva Nageswara Rao,G., Srinivasu,N., Srinivasu,S.V.N., and Rama Koteswara Rao,G. (2015). Dynamic Time Slice Calculation for Round Robin Process Scheduling Using NOC, International Journal of Electrical and Computer Engineering, Vol. 5, No. 6.

N. Abbas,N., Ali, K., and Seifedine,K. (2011). A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average, International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1.

Rakesh,M., Behera,H. S., Khusbu,P., Monisha,D. and Lakshmi,P. (2011). Priority Based Dynamic Round Robin (PBDRR) Algorithm with Intelligent Time Slice for Soft Real Time Systems, International Journal of Advanced Computer Science and Applications, Vol. 2, No.2.

Manish Kumar,M. and Abdul Kadir,K. (2012). AN IMPROVED ROUND ROBIN CPU SCHEDULING ALGORITHM, Journal of Global Research in Computer Science, Volume 3, No. 6.