Improved Round Robin Scheduling Algorithm for Cloud Computing

Sarah Mohamed Ahmed Lotfy
Cairo University

Improved Round Robin Scheduling Algorithm for Cloud Computing

Sarah Mohamed Ahmed Lotfy

Computer Engineering Department
Faculty of Engineering, Cairo University
Cairo, Egypt

sarah.lotfy98@eng-st.cu.edu.eg

***Abstract*—Cloud computing plays an important role in our daily life. It helps us to share and update data, knowledge, storage, and resources between various regions. Process scheduling is very important in cloud computing, it improves resource utilization, decreases process waiting time and maximizes the number of processes done per time. The Round Robin algorithm is one of the most common scheduling algorithms due to its simplicity and fairness. This paper represents a new round-robin algorithm having proficient quantum time. This paper will introduce an algorithm that makes the round-robin algorithm more efficient and try to improve it by combining Shortest-Job-First (SJF) and Round Robin (RR) with an efficient algorithm for choosing suitable quantum time. The algorithm is experimentally and comparatively better than the mentioned round robin algorithms in this paper. The proposed algorithm has better results in decreasing average turnaround time, average waiting time and number of context switching.**

***Index Terms*—Round Robin, Shortest Job First SJF, scheduling algorithm, quantum time, burst time, average waiting time, turnaround time, context switching.**

I. INTRODUCTION

Cloud computing is considered one of the most important research fields used by a lot of developers and designers. Cloud computing services cover many options, from the basics of storage, networking and processing power. Services don't require you to be physically close to the computer hardware. It is an innovative technology that helps in the sharing of data and information. Cloud computing allows consumers to use resources such as storage or applications, remote servers and internet networks. Job scheduling is very important in cloud computing. Scheduling is the concept that is utilized in terms of managing the proper distribution of resources and handling the execution of processes by the operating system. It improves the performance of cloud computing by improving recourses utilization, decreases process waiting time and maximize the number of processes done per time (throughput).Round Robin (RR) Algorithm is one of the most common scheduling algorithms, simplest, fairest designed especially for time-sharing systems. In this paper, we will try to improve round-robin algorithm. This paper will introduce an algorithm that makes the round-robin algorithm more efficient by combining Shortest-Job-First (SJF) and Round Robin(RR) with an efficient algorithm for choosing suitable quantum time. To increase the throughput of Round Robin we will combine the RR algorithm with the shortest job first algorithm (SJF). So Processes will be arranged first according to their burst time. A large number of round-robin algorithms is presented with different types of quantum time that can be determined dynamically or statically. The performance of RR algorithms depends on the determination of good quantum time. The efficiency of the RR algorithm is relying on quantum time. If the quantum time is too large, the algorithm will tend to become FCFS algorithm. While the quantum time is too small, the algorithm will give a poor performance so to increase the efficiency of round-robin we will need to choose the best quantum time by an efficient algorithm. The proposed algorithm calculate quantum time by getting the summation of a median of differences of adjacent consecutive process burst times with median of burst times. The remainder of this paper is organized as 5 main sections where, in Section 2, the concepts of task scheduling and some related work are presented. In section 3, the proposed algorithm is presented. In section 4, the experimental results and analyzation. section 5 includes conclusions and future work.

II. RELATED WORK

Several scheduling algorithms have been proposed for distributing cloud computing. All these algorithms target to reduce the waiting time of processes in the queue list, increase efficiency and balance between available resources. We have so many existing algorithms which tend to achieve optimal and more efficient round-robin task scheduling.
In MRRA [1] algorithm arranges the jobs according to their burst time and computes dynamic quantum time by

calculating the mean of the burst times of all tasks. But it has drawbacks as average turnaround time is large.
In SRBRR [2] median of Burst is calculated and used as a quantum time without sorting the process. Has advantages:- Better in terms of the number of context switching compared to MRRA[1] algorithm. But also has disadvantages as it leads to FCFS in some case.
Some algorithms use two values for a quantum time as in AMBRR [3] there are two quantum time is calculated T.Q1= median of burst time. T.Q2 = (T.Q1- highest Burst time). T.Q1 is used if the number of completed cycles is even and T.Q2 is used if the number of completed cycles is odd. This algorithm has advantages as the computational cost of finding the median in each cycle is reduced compare to SRBRR[2].
In DRR [4] algorithm, The algorithm calculates quantum time by calculating the average of two high Burst time and then subtract from the average of two lowest arrival times. This algorithm has advantages give better result in terms of average turnaround time and the average waiting time and number of Context switching. But has disadvantages as leading to FCFS In some cases.
In [5] Algorithm has two steps, first step: new priorities are allocated to all arrived processes according to their burst times so the lowest value of burst time has the highest priority with the same value of time quantum time. If a new task is arrived Step 2 calculates the burst time of all tasks and re-arranged again with the new priority until the value of a burst time equal to zero. It's an advantage in avoiding frequent context switch.
Some algorithms put conditions for choosing quantum time as in [6] H. S. Behera et al that proposed a new dynamic Round Robin algorithm to determine the quantum time value by considering the number of tasks in the ready queue. If the number of tasks is odd, then the quantum time equals the middle burst time in the ready queue or the quantum will be the average value of the middle burst time plus its successor one for even number of tasks in the ready queue. Their algorithm showed good results in decreasing the performance parameters. There are some algorithms that use two queues as in this algorithm [7] that splits the tasks ready queue into two sub-queues Q1 for short tasks and Q2 for long tasks using the median of burst times as the threshold. If a burst time of a task is less than or equal to the median, insert task into a Q1 otherwise insert task into Q2. This algorithm decreases turnaround time and waiting time but has drawbacks like the number of context switching is large.

The objective of this paper is to propose a new algorithm to enhance the performance of the round- robin algorithm and solve some of the drawbacks of past algorithms via decreasing average waiting time and turnaround average time, establish balance for available resources and increasing the performance of cloud computing system. This enchantment is achieved by combining shortest job first SJF and Round Robin by adding priority to job processes according to their burst time and choosing the best and suitable quantum time by a certain algorithm.

III. Proposed Algorithm

The proposed algorithm focused on increasing the throughput of Round Robin and increase the efficiency of the Round robin algorithm. We will increase throughput by merging between RR algorithm and shortest job first algorithm (SJF), so first all the processes will be sorted ascendancy according to their burst time. The efficiency of the round-robin algorithms relying on the quantum time. So choosing an appropriate quantum time is important. Our algorithm will calculate the quantum time by arranging the processes ascendancy according to their burst times then get the median of burst times and add it to the median of differences of burst times of each two adjacent consecutive processes present into the ready queue.

$$BTDIFF[i] = BT(Pi+1) - BT(Pi) \tag{1}$$

$$TQ = MBT + M(BTDIFF) \tag{2}$$

By this strategy, it is possible to get the efficient and more appropriate quantum time specially that the median of differences of each two consecutive processes burst times is calculated.

In order to make the proposed algorithm more explicit, the steps of the proposed algorithm described in the following diagram.

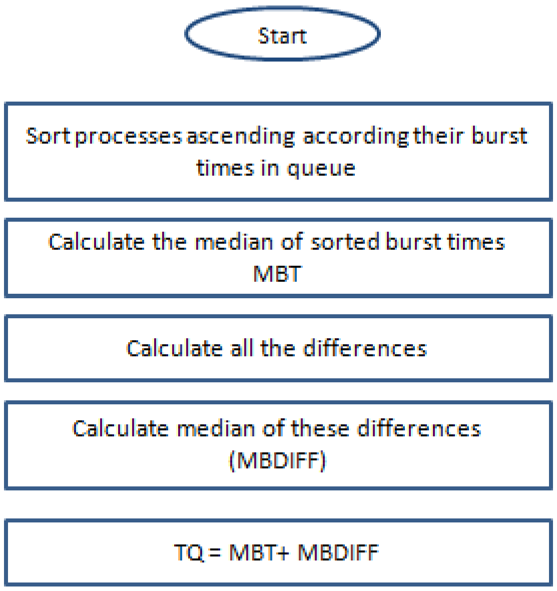
Start
Sort processes ascending according their burst times in queue
Calculate the median of sorted burst times MBT
Calculate all the differences
Calculate median of these differences (MBDIFF)
TQ = MBT+ MBDIFF

Fig. 1. Calculating quantum time.

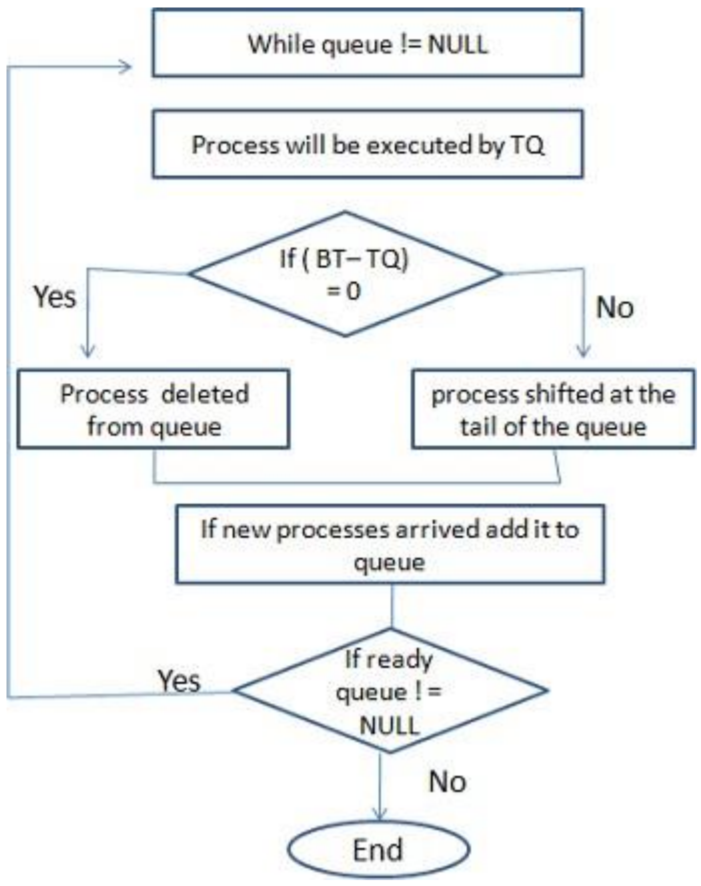
While queue != NULL
Process will be executed by TQ
If (BT– TQ) = 0
Yes
No
Process deleted from queue
process shifted at the tail of the queue
If new processes arrived add it to queue
If ready queue ! = NULL
Yes
No
End

Fig. 2. Algorithm

1. Sorting all processes in ascending according their burst times in queue.
2. Calculate the median of sorted burst times.
3. Calculate all the differences of two adjacent consecutive processes.
4. The median of these differences had been calculated.
5. Calculate quantum time by summation of median differences and median burst time.
6. While queue is not empty.
7. Process will be executed by TQ time quantum, if (process burst time – time quantum) = 0, the process will be deleted from queue.
8. If (process burst time – time quantum) !=0, the process will be shifted at the tail of the ready.
9. If new process arrived go to step 6.
10. Calculate average turnaround time and average waiting time.
11. End while.

Here is an example for our algorithm, suppose that there are 4 processes P1, P2, P3, P4 having burst times such as 32, 18, 20, 42 ms respectively and all processes have arrived at zero milliseconds, processes are arranged in ascending order so it will be P2, P3, P1, P4. Then calculate the median of burst times it will be (18+42) / 2= 30. Then calculate the differences of each two consecutive processes it will be 2, 12, 10 respectively and get the median of these differences by (2+10) / 2 = 6 , so the quantum time will be MBT + M(BTDIFF) = 30 + 6 = 36.

The proposed algorithm also measured for the processes with equal burst times. Suppose that there are 5 processes all having equal burst time such as P1, P2, P3, P4 and P5 all have the same burst time as 22 ms. After sorting in ascending order, the first process will be executed because their burst times are equal.The differences between two consecutive processes will be 0, 0, 0, 0 so median of differences will be 0+0 / 2 =0 and median will be 22 + 22 / 2 = 22. The time quantum will be MBT + M(BTDIFF) = 22 + 0 = 22 which is equal to the burst time of all processes.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

This algorithm target to increase the performance of job scheduling by getting maximum throughput, maximum CPU utilization, reducing age average turnaround time(TAT), minimizing average waiting time (WAT) and the number of context switching (CS). There are two types of processes, first type processes have (0) arrival time and second type processes have different arrival time.

In this experiment we will apply algorithm on two data set having same arrival time and different arrival time. We will start with processes with same arrival time as in table 1. Table 1. represents a data set having 5 processes P1, P2, P3, P4 and P5 having distinct burst time 105, 60, 120, 48 and 75 with zero arrival time.
First we will sort the processes in ascending order according to their burst times so they will be P4 , P2, P5, P1, P3. Then get the median of burst times after sorting median = (120+ 48) / 2 = 84, get the differences which will be 12, 15, 30, 15 and get the median of differences = (12 +15) / 2 = 13.5 so TQ = 84 + 13.5 = 97.5 get the round of TQ so TQ = 98. Process execution has been shown through Gantt-Chart (Table 2) .

TABLE I
PROCESSES WITH ZERO ARRIVAL TIME (CASE STUDY – 1)

Processes	Arrival Time	Burst Time
P1	0	105
P2	0	60
P3	0	120
P4	0	48
P5	0	75

Average turnaround time = (386 + 108 + 379 + 48 + 183) / 5 = 220.8
Average waiting time = (281 + 48 + 259 + 0 + 108) / 5 = 139.2
Context switching = 7

TABLE II
SIMULATION THROUGH GANTT-CHART OF CASE STUDY -1

P4	P2	P5	P1	P3	P1	P3
48	108	183	281	379	386	408

We will apply algorithm on the second type of processes which have different arrival times as in table 3. Table 3 also represents another data set also having 5 processes P1, P2, P3, P4, P5 having distinct burst time 45, 90, 70, 38 and 55 and different arrival time.

TABLE III
PROCESSES WITH DIFFERENT ARRIVAL TIME (CASE STUDY – 2)

Processes	Arrival Time	Burst Time
P1	0	45
P2	5	90
P3	8	70
P4	15	38
P5	20	55

First we will sort the processes in ascending order according to their burst times so they will be P4 , P1, P5, P3, P2. Then get the median of burst times after sorting median = (90 + 38) / 2 = 64, then get the differences which will be 7, 10, 15, 20 and get the median of differences = (20 + 7) / 2 = 13.5.
so TQ = 64 + 13.5 = 77.5 get the round so TQ = 78. Process execution has been shown through Gantt-Chart (Table 4) .

TABLE V
SIMULATION THROUGH GANTT-CHART OF CASE STUDY -2

P1	P4	P5	P3	P2	P2
45	83	138	208	286	298

Average turnaround time = (45 + 68 + 118 + 200 + 293) / 5 = 144/8
Average waiting time = (0 + 30 + 63 + 130 + 203) / 5 = 85.2
Context switching = 6

To evaluate the performance and efficiency of this algorithm there is an experiment applied on two different data sets having zero arrival time and distinct arrival time [8] this literature compares between the proposed algorithms that target to improve the round-robin algorithm. The proposed algorithm is comparatively excellent in getting a smaller average turnaround time and a smaller average waiting time and smaller context switching, table 5 shows that the proposed algorithm has excellent results and more efficient results than past algorithms.

There is a comparison table over the above Table 1 mentioned dataset-1. The comparison occurs on the proposed algorithm against general round robin (RR), DQRRR [9], IRRVQ [10], SARR [11], R.R – 10 [12], MRR [13], DABRR [14]. The comparative terms are number of Context Switching (CS) represented in the third row, average Waiting Time (AWT) presented in forth row and average Turn-Around-Time (ATAT) demonstrated in fifth row in the Table 5. This table applied on random order processes.

Comparatively with the results of the past algorithms the proposed algorithm has better results in smaller average turnaround time and smaller average waiting time.

There is another comparison table for mentioned dataset 2 in table 3 which include processes with different arrival time. In this comparison we will add another algorithm [15] which also improve round robin in 2019 and also our results is better than it. The comparison occurs on the proposed algorithm against general round robin (RR), DQRRR [9], IRRVQ [10], SARR [11], R.R – 10 [12], MRR [13], DABRR [14],RR 2019 . The comparative terms are number of Context Switching (CS) represented in the third row, average Waiting-Time (AWT) presented in forth row and average Turn-Around-Time (TAT)

TABLE IV
COMPARISON ON PROPOSED ALGORITHM (PRR) AGAINST RR, DQRRR, IRRVQ, SARR, R.R-10, MRR, DABRR FOR DATASET-1

Algorithm	RR	DQRRR	IRRVQ	SARR	R.R-10	MRR	DABRR	Propose algorithm
TQ	25	75,37,8	48,12,30,15	120	25,50	72, 45,25	81,31,8	98
CS	17	7	14	4	11	8	7	7
AWT	245.4	192.8	193.2	177.6	237.8	168.6	141.6	139.2
ATAT	327	274.4	274.8	259.2	319.4	250.2	223.2	220.8

demonstrated in fifth row in the Table 6.This comparison on random ordering processes.

According to the experiments results, the proposed algorithm is experimentally better shown in comparison Table 5 and Table 6.
The proposed algorithm has better results in reducing average turnaround time and smaller average waiting time and smaller number of context switching.

The experimental results of the two different types of data sets represented in Table 5 and 6 are illustrated more in form of graphs that show that the proposed algorithm is much better than the past algorithms that seek to enhance the Round Robin algorithm.

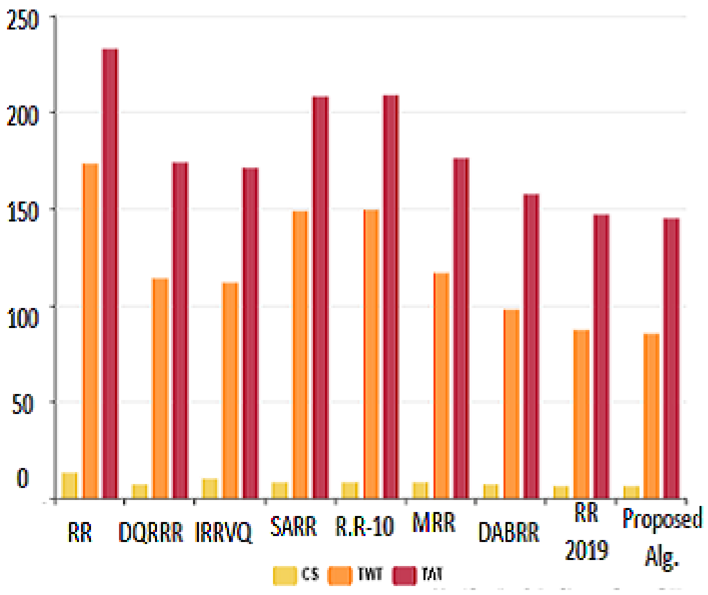

Fig. 4. Graphical representation over dataset-2

TABLE VI
COMPARISON ON PROPOSED ALGORITHM (PRR) AGAINST RR, DQRRR, IRRVQ, SARR, R.R-10, MRR, DABRR, RR 2019 FOR DATASET-2

Algorithm	RR	DQRRR	IRRVQ	SARR	R.R-10	MRR	DABRR	RR 2019	Propose algorithm
TQ	25	45,62,18,10	45,38,17,15,20	45,54,16,20	25, 50,100	45, 52, 35,25	45, 63, 17,10	98	
CS	13	7	10	8	8	8	7	6	6
AWT	173.2	113.6	111.4	148.6	149.2	116.4	97.8	87.2.	85.2
ATAT	232.8	173.2	171	208.2	208.8	176	157.4	146.8	144.8

Here are Graphical representations to illustrate the effieciency of our proposed algorithm. There are three bars the red bar is the average turnaround time, the orange bar is the average waiting time and the yello bar is the number of context switching. The labels on y- axis are the algorithms and the label on x-axis is the time in milli second.

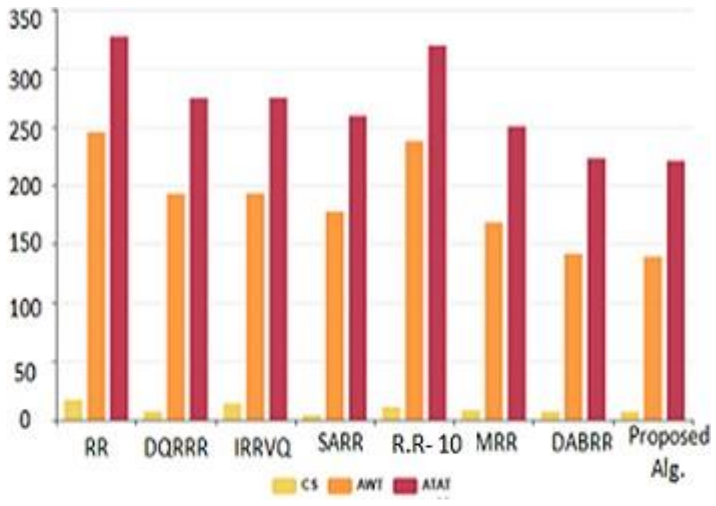

Fig. 3. Graphical representation over dataset-1

V. CONCLUTION AND FUTURE WORK

In this paper a new optimized Round Robin scheduling algorithm for cloud computing is proposed to solve problems in scheduling and to reduce average waiting time, average turnaround time, number of context switching and increasing throughput. It has the ability to create balance load between jobs processes in cloud computing at ready queue by sorting the processes according to their burst time then calculate the quantum time by summation of median differences and median burst times. By applying and deploying the proposed algorithm it is possible to enhance and improve the process allocation through system. Since calculating quantum time is the most important term in increasing the performance of the system. When compare our proposal with the previous algorithm, we can easily notice that the proposed algorithm have a significant effect on performance of cloud computing. Our experimental results and analysis show that the proposed algorithm is more efficient than the past algorithms that seek to enhance Round Robin algorithm. When all processes have same burst time the first processes will be scheduled first according to the proposed algorithm and this is not fair all time. Our future work is to apply priority when all processes have same burst time, priority other than burst time. Also will try to decrease number of context switching.

REFERENCES

[1] Pandaba Pradhan, Prafulla Ku. Behera and B NB Ray, ”Modified Round Robin Algorithm forResource Allocation in Cloud Computing ”,Science Direct, Procedia Computer Science 85 (2016) 878 – 890,2016.

[2] Rami J. Matarneh,”Self-Adjustment TimeQuantum in Round Robin AlgorithmDepending on Burst Time of Now RunningProcesses”, American J. of Applied Sciences6(10): 1831 -1837,2009.

[3] Salman Arif,Naveed Ghaffar,Ali Javed”Implementation of Alternating Median BasedRound Robin Scheduling Algorithm”,IEEE,International Conference on Computer andInformation Technology,2016.

[4] Salman Arif, Saad Rehman and Farhan Riaz”Design of A Modulus Based Round RobinScheduling Algorithm”,IEEE, 9th MalaysianSoftware Engineering Conference, Dec. 2015.

[5] Kumar, D. Praveen, T. Sreenivasula Reddy, and A. Yugandhar Reddy. ”Finding Best Time Quantum for Round Robin Scheduling Algorithm to avoid Frequent Context Switch.” International Journal of Computer Science and Information Technologies 5.5 (2014): 6750-6754.

[6] H. S. Behera, R. Mohanty and D. Nayek, ”A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis”, International Journal of Computer Applications (0097 - 8887), Vol. 5, No. 5, pp. lOIS, August 2010.

[7] S. Elmougy, S. Sarhan, and M. Joundy, —A novel hybrid of Shortest job first and round Robin with dynamic variable quantum time task scheduling technique,‖ 2017.

[8] Dash, Amar Ranjan, and Sanjay Kumar Samantra. ”An optimized round Robin CPU scheduling algorithm with dynamic time quantum.” arXiv preprint arXiv:1605.00362(2016).

[9] Fataniya, Bhavin, and Manoj Patel. ”Dynamic Time Quantum Approach to Improve Round Robin Scheduling Algorithm in Cloud Environment.” (2018).

[10] Yadav, Rakesh Kumar, et al. ”An improved round robin scheduling algorithm for CPU scheduling.” International Journal on Computer Science and Engineering 2.04 (2010):1064-1066.

[11] Matarneh, Rami J. ”Self-adjustment time quantum in round robin algorithm depending on burst time of the now running processes.” American Journal of Applied Sciences 6.10 (2009): 1831.

[12] Singh, Ajit, Priyanka Goyal, and Sahil Batra. ”An optimized round robin scheduling algorithm for CPU scheduling.” International Journal on Computer Science and Engineering 2.07 (2010): 2383-2385.

[13] Verma, Rishi, Sunny Mittal, and Vikram Singh. ”A Round Robin Algorithm using Mode Dispersion for Effective Measure.” International Journal for Research in Applied Science and Engineering Technology (IJRASET) (2014): 166-174.

[14] Dash, Amar Ranjan, and Sanjay Kumar Samantra. ”An optimized round Robin CPU scheduling algorithm with dynamic time quantum.” arXiv preprint arXiv:1605.00362(2016).

[15] Dipto Biswas, Md. Samsuddoha, ” Determining Proficient Time Quantum to Improve the Performance of Round Robin Scheduling Algorithm”, International Journal of Modern Education and Computer Science(IJMECS), Vol.11, No.10, pp. 33-40, 2019.DOI: 10.5815/ijmecs.2019.10.04