Volume 2, Issue 9, September 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Comparative Performance Analysis of Even Odd Round Robin Scheduling Algorithm (EORR) using Dynamic Time Quantum with Round Robin Scheduling Algorithm using static Time Quantum

Pallab Banerjee[1], Probal Banerjee[2], Shweta Sonali Dhal[3]
[1]*Computer Science and Engineering*
[2]*Electronics and Communication Engineering*
[3]*Electrical and Electronics Engineering*
[123]*Cambridge Institute of Technology, Ranchi University*

Abstract- In the Round-Robin scheduling scheme, the scheduler processes each job, one after another, after giving a preset quantum of time. It is a preemptive CPU scheduling algorithm which switches between the processes when static time quantum expires. Round Robin scheduling algorithm is designed especially for time sharing operating system but it has its disadvantages that are its Longer Average Waiting Time, Higher Context Switches and Higher Turnaround Time. In this scheduling algorithm the main idea is to adjust the time quantum dynamically so that (EORR) perform better performance than Round Robin scheduling algorithm.

Keywords-Operating System, Round Robin, Average Mid Max Round Robin, Turnaround time, Waiting time, Context Switch.

I. INTRODUCTION

Operating system is a system software which makes an interface between end user and computer hardware, so that the user can handle the system in a convenient manner. In a single user environment, there was no need to choose any task because task execution continues one after another, but in multitasking environment, it becomes necessary for the processor to choose a task from the ready queue. Operating system follows a predefined procedure for selecting process among number of processes from the ready queue, known as Scheduling. Scheduler selects the ready processes from memory and allocates resource as per their requirement. Whenever one process waits for some other resource, scheduler selects next process and allocates CPU to it. This process continues till the system request for termination of execution and then the last CPU burst ends up with it [2].

II. PRELIMINARIES

Program is refer to the set of instruction that are executed in pipeline fashion. Program in execution is called process. Process are represented by Process Control Block (PCB).PCB contains many information about process such as process state, process number, program counter, list of open files, registers and CPU scheduling information[4]. When process enter into the main memory and are ready and waiting to execute are kept in the data structure called Ready Queue. When a process assign to the CPU, it execute or while waiting for some event to occur. The processes which are waiting for I/O request are kept in Device Queue. The Long term scheduler or job scheduler select process from job pool and load them into main memory for execution. Short term scheduler or CPU scheduler select from among the processes that are ready to execute and allocates the CPU to one of them. Medium term scheduler is used in time sharing system. The main advantage of Medium term scheduler is sometimes it remove processes from main memory and thus reduce degree of multiprogramming. Later the process can be reintroduced into memory, and its execution can be continued where it left off. This scheme is called as swapping. So, the process is swapped out, and is later swapped in, by the medium term scheduler [1].

III. SCHEDULING ALGORITHMS

In the First-Come-First-Serve (FCFS) algorithm, process that arrives first is immediately allocated to the CPU based on FIFO policy. In Shortest Job First (SJF) algorithm, process having shortest CPU burst time will execute first. If two processes having same burst time and arrive simultaneously, then FCFS procedure is applied. Priority scheduling algorithm, provides priority (internally or externally) to each process and selects the highest priority process from the ready queue. In case of Round Robin (RR) algorithm, time interval of one time quantum is given to each process present in the circular queue emphasizing on the fairness factor.

IV. ROUND ROBIN SCHEDULING ALGORITHM

RR Scheduling Algorithm is the simplest and widely used algorithm as it gives fairness to each process. Newly arrived processes are kept in the rear part of the queue. Scheduler chooses each process from front of the queue and allocates the CPU for one time quantum. The performance of RR algorithm depends heavily on the size of the time quantum . For smaller time quantum, the context switching is more and for larger time quantum, response time is more. Overall performance of RR may decrease for weak time quantum selection. Therefore, choice of an optimal time quantum is necessary[3].

V. PERFORMANCE METRICS

The proposed algorithm is designed to meet all scheduling criteria such as maximum CPU utilization, maximum throughput, minimum turnaround time, minimum waiting time and context switches. Here we are considering three performance criteria in each case of our experiment.

Turnaround Time (TAT)=Finish Time–Arrival Time.
Average Turnaround Time should be less.
Waiting Time (WT)= Start Time- Arrival Time.
Average Waiting Time should be less.
Context Switch(CS)
The number of context Switch should be less.

VI. RELATED WORK

In the last few years different approaches are used to increase the performance of Round Robin scheduling like Adaptive Round Robin Scheduling using Shortest Burst Approach Based on Smart Time Slice[5], Multi-Dynamic time Quantum Round Robin (MDTQRR)[12].Min-Max Round Robin (MMRR)[8], Self-Adjustment Time Quantum in Round Robin (SARR)[11], Dynamic Quantum with Re-adjusted Round Robin (DQRRR)[7],Average Max Round Robin Algorithm (AMRR)[2]. Mid Average Round Robin (MARR)[6]. Average Mid Max Round Robin Scheduling Algorithm (AMMRR)[4].

VII. PROPOSED APPROACH

Let's assume that the burst time of the processes is taken as unsorted sequence. Generally in Round Robin algorithm the performance depends upon the size of fixed or static Time Quantum (TQ). If TQ is too large then Round Robin algorithm approximate to First Come First Served (FCFS). If the Time Quantum is too small then there will be many context switching between the processes. So, our approach solved this problem by taking a dynamic TQ. Here in this paper we calculate two time quantums i.e. TQ1 and TQ2. Where TQ1 is the average burst time of all the processes at even places in the ready queue and TQ2 is the average burst time of all the processes at odd places in the ready queue. Then we compare the two time quantums, to get the greater Time Quantum i.e.

If TQ1> TQ2
then TQn=TQ1
else TQn=TQ2

VIII. PROPOSED ALGORITHM

In our proposed algorithm, processes are already present in the Ready Queue (RQ). By default, Arrival Time (AT) is assigned to zero. The number of processes 'n' and CPU Burst Time (BT) are accepted as input and Average Turnaround Time (ATT), Average Waiting Time (AWT) and number of Context Switch (CS) are produced as output. Let TQn be the new time quantum. The pseudo code for the algorithm is presented in Figure 1 and the flowchart of the algorithm is presented in Figure 2.

1. While(RQ != NULL)
// N= Number of Processes in the ready queue
// BT= Burst Time of the Processes
// RQ= Ready Queue
//Eavg= Average Burst Time of Even Processes
//Oavg= Average Burst Time of Odd Processes
// E= Counter inside Even for loop counting the number of Even processes.
//O= Counter inside Odd for loop counting the number of Odd processes.
//TQn=New Time Quantum

// TQ1=Average Time Quantum of Even Processes
// TQ2=Average Time Quantum of Odd Processes

2. for i=1 to N loop
{
if(i%2==0)
{
Eavg=Eavg+BTPi
E++
} //End of if
} // End of for
Set TQ1=Eavg/E
//(Use round off function in TQ1)

3. for i=1 to N loop
{
if(i%2==1)
{
Oavg=Oavg+BTPi
O++
} //End of if
} //End of for
Set TQ2=Oavg/O
//(Use round off function in TQ2)

4. if (TQ1>=TQ2)
Set TQn=TQ1
Else Set TQn=TQ2

5. **for i=1 to N loop** // Assign TQ to (1 to n) processes
{
Pi->TQn //Assign TQn to all the available processes
}//End of for

6. If one process is there then after calculation TQn is equal to BT itself
7. Calculate the remaining Burst time of the processes.
8. If (new process arrived and BT!=0 Or new process is arrived and BT==0 Or new process is not arrived and BT!=0)
then go to step 1
else
go to step 9
end of if
end of while
9. Calculate ATT,AWT,CS
//ATT=Average Turnaround time
//AWT=Average waiting time
//CS=Number of context switch
10. End

Figure 1. Pseudo code for Mid-Max Round Robin (MARR) algorithm

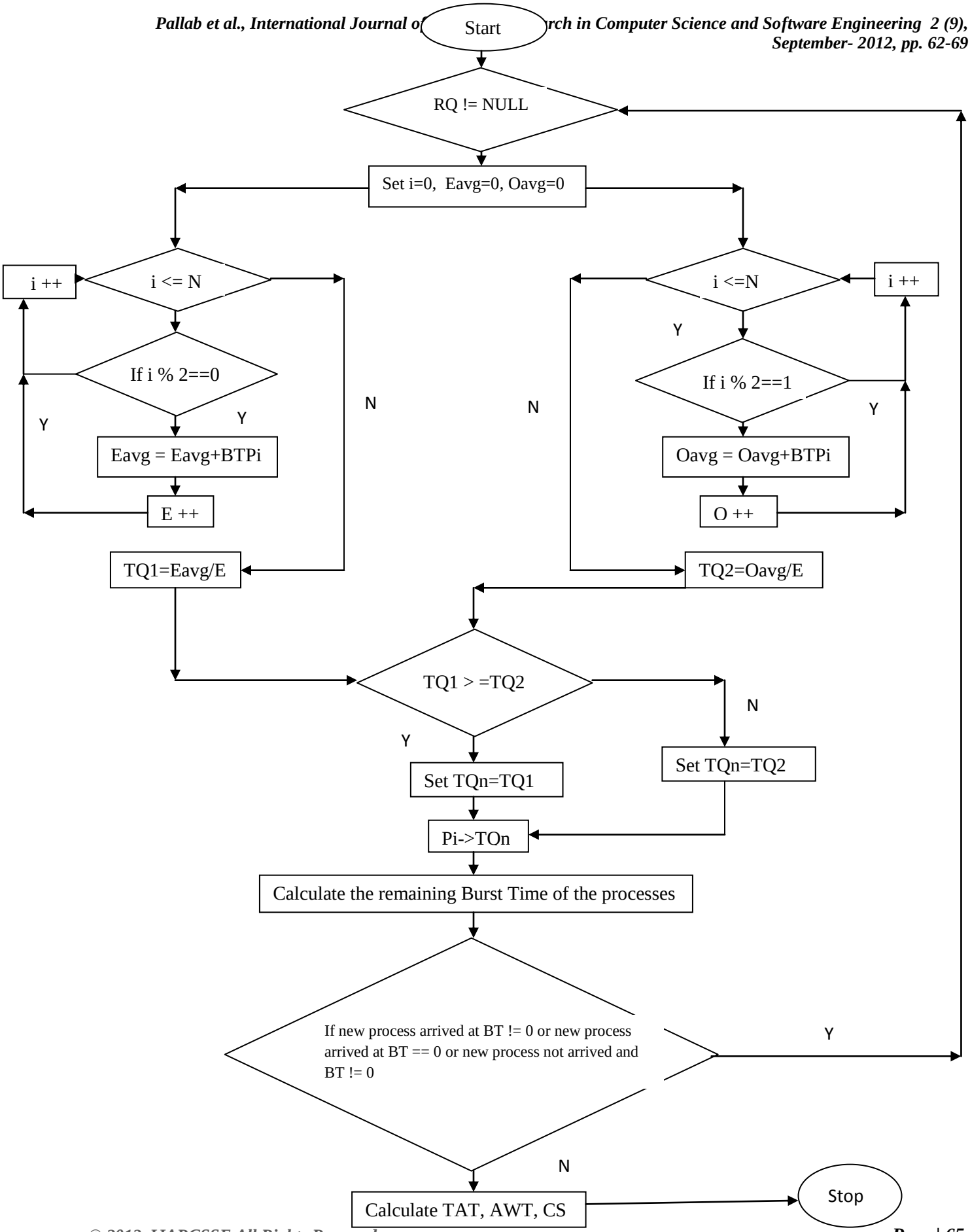

Figure 2. Flowchart of Even Odd Round Robin (EORR) algorithm

IX. ILLUSTRATION

Given the burst time of the process in unsorted sequence:(P1=90,P2=70,P3=20,P4=40,P5=50) taking arrival time=0. Considering the burst time of the processes in Even positions be the average of the summation of all the Even processes. So TQ1=(70+40)/2=55. Then calculatethe burst time of odd processes which is the average of the the summation of all Odd processes. So TQ2=(90+20+50)/3=53.3333,Approx TQ2=54. Then we compare which TQ is bigger one. Is TQ1>TQ2. If true then TQn=TQ1 else TQn=TQ2. So TQn=55. After first iteration the remaining CPU burst time sequence is P1=35,P2=15,P3=0,P4=0 and P5=0. In this case, processes P3,P4,P5 are deleted from the Ready Queue. Now there is only two processes left i.e P4 and P5 with burst time. Now TQn=35. After second iteration P4=0 and P5=0. Now there is no process in the RQ, it completes its execution and TAT, AWT and CS are calculated. In this case, TAT=209, AWT=155 and CS=6.

X. EXPERIMENTAL ANALYSIS

In every case we will compare the result of the proposed Average Mid Max Round Robin (EORR) method with Round Robin (RR) scheduling algorithm. Here we have taken 20 as the static time quantum (TQ) for RR algorithm

CASE 1:-Let's consider five processes with Burst time (P1=25,P2=45,P3=60,P4=75,P5=90) and Arrival Time =0 as shown in the Table 1. Table 2 shows the output using RR algorithm and EORR algorithm. Figure 3 and Figure 4 shows Gantt chart of both RR and EORR algorithm respectively.

Table 1. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	25
P2	0	45
P3	0	60
P4	0	75
P5	0	90

Table 2: Comparison between RR algorithm and our new proposed EORR algorithm (CASE 1).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	213	154	15
EORR	60,30	157	86	6

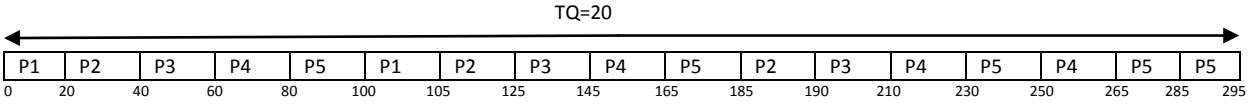

Fig.3: Gantt chart of RR from Table 1 of CASE 1.

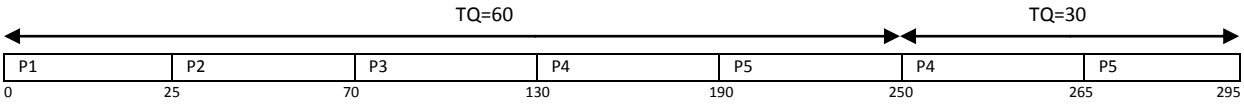

Fig.4: Gantt chart of EORR from Table 1 of CASE 1.

CASE 2:-Let's consider five processes with Burst time (P1=41, P2=42, P3=43, P4=44, P5=45) and Arrival Time =0 as shown in the Table 3. Table 4 shows the output using RR algorithm and EORR algorithm. Figure 5 and Figure 6 shows Gantt chart of both RR and EORR algorithm respectively

Table 3. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	41
P2	0	42
P3	0	43
P4	0	44
P5	0	45

Table 4: Comparison between RR algorithm and our new proposed EORR algorithm (CASE 2).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	207	164	14
EORR	43,2	135.6	84	4

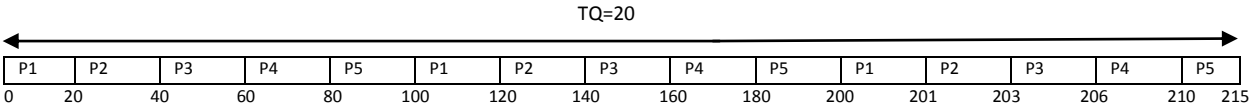

Fig.5: Gantt chart of RR from Table 3 of CASE 2.

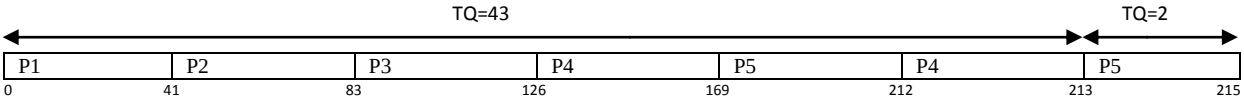

Fig.6: Gantt chart of EORR from Table 3 of CASE 2.

CASE 3:-Let's consider five processes with Burst time (P1=8,P2=16,P3=32,P4=64,P5=128) and Arrival Time =0 as shown in the Table 5. Table 6 shows the output using RR algorithm and EORR algorithm. Figure 7 and Figure 8 shows Gantt chart of both RR and EORR algorithm respectively

Table 5. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	8
P2	0	16
P3	0	32
P4	0	64
P5	0	128

Table 6: Comparison between RR algorithm and our new proposed EORR algorithm (CASE 3).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	111.2	62	11
EORR	56,72	102.4	52.8	6

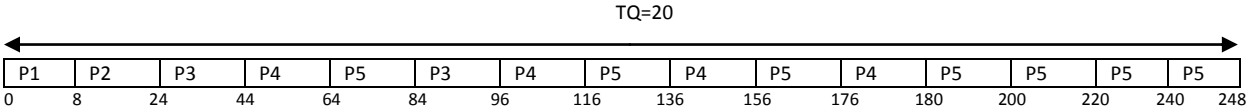

Fig.7: Gantt chart of RR from Table 5 of CASE 3.

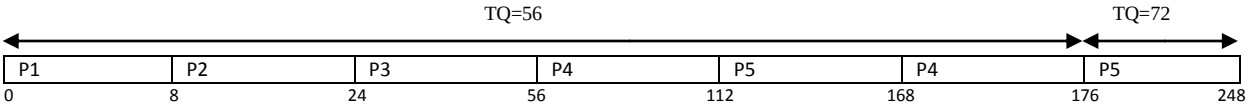

Fig.8: Gantt chart of EORR from Table 5 of CASE 3.

CASE 4:-Let's consider five processes with Burst time (P1=25,P2=30,P3=45,P4=55,P5=70) and Arrival Time as (P1=0,P2=10,P3=15,P4=25,P5=30) shown in the Table 7. Table 8 shows the output using RR algorithm and EORR algorithm. Figure 9 and Figure 10 shows Gantt chart of both RR and EORR algorithm respectively.

Table 7. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	25
P2	10	30
P3	15	45
P4	25	55
P5	30	70

Table 8: Comparison between RR algorithm and our new proposed EORR algorithm (CASE 4).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	131	103	12
EORR	47,23	105.4	60.4	6

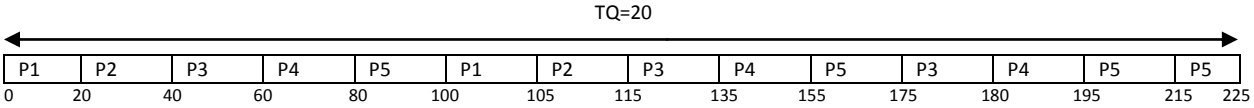

Fig.9: Gantt chart of RR from Table 7 of CASE 4.

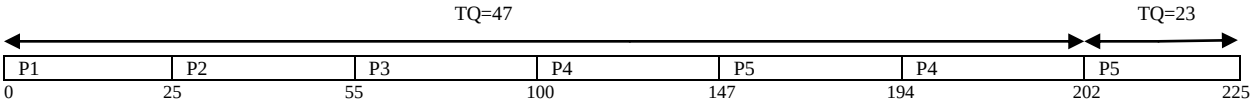

Fig.10: Gantt chart of EORR from Table 7 of CASE 4.

CASE 5:-Let's consider five processes with Burst time (P1=20,P2=30,P3=50,P4=60,P5=70) and Arrival Time as (P1=0,P2=5,P3=20,P4=25,P5=30) shown in the Table 9. Table 10 shows the output using RR algorithm and EORR algorithm. Figure 11 and Figure 12 shows Gantt chart of both RR and EORR algorithm respectively.

Table 9. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	20
P2	5	30
P3	20	50
P4	25	60
P5	30	70

Table 10: Comparison between RR algorithm and our new proposed EORR algorithm (CASE 5).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	132	86	11
EORR	47,13,10	124	78.2	7

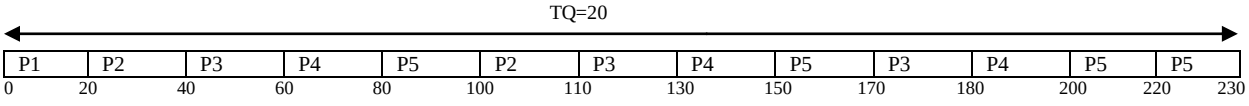

Fig.11: Gantt chart of RR from Table 9 of CASE 4.

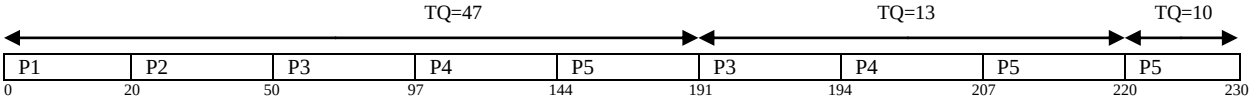

Fig.12: Gantt chart of EORR from Table 9 of CASE

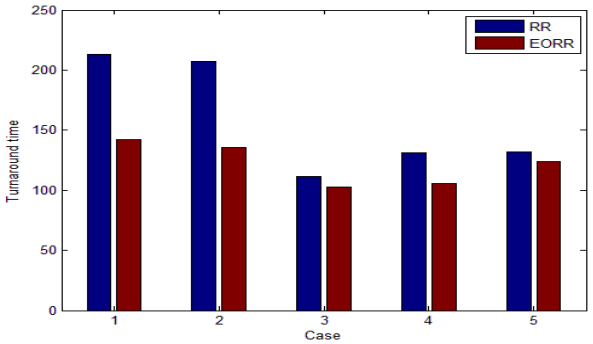

Fig.13: Comparison of average turnaround time of RR and EORR taking arrival time into consideration.

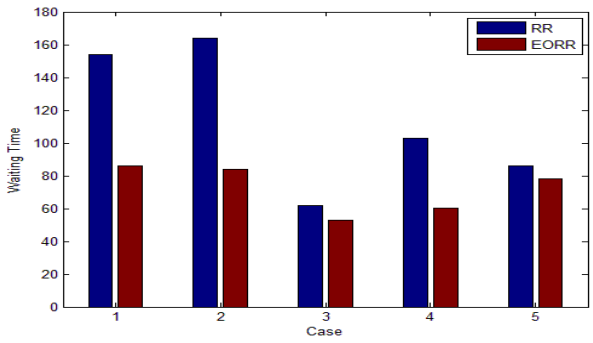

Fig.14. Comparison of average waiting time of RR and EORR taking arrival time into consideration.

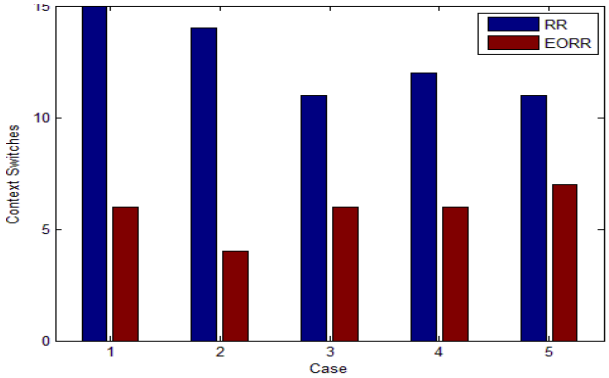

Fig.15. Comparison of Context Switches of RR and EORR taking arrival time into consideration.

XI. CONCLUSION

From the above study we conclude that the proposed Even Odd Round Robin Scheduling algorithm shows better performance as compared to Round Robin Scheduling algorithm by decreasing the Total Turnaround Time, Average Waiting Time and Number of context switching .This is achieved by increasing the Time Quantum Dynamically.

REFERENCE

[1] "Silberschatz, A., P.B. Galvin and G. Gagne, 2008" Operating Systems Concepts. 7th Edn., John Wiley and Sons, USA., ISBN: 13: 978-0471694663, pp: 944.

[2] Pallab banerjee, probal banerjee, shweta sonali dhal, "Comparative Performance Analysis of Average Max Round Robin Scheduling Algorithm (AMRR) using Dynamic Time Quantum with Round Robin Scheduling Algorithm using static Time Quantum",IJITEE,ISSN: 2278-3075, Volume-1, Issue-3, August 2012.

[3] "Tanebaun, A.S., 2008" Modern Operating Systems. 3rd Edn., Prentice Hall, ISBN: 13:9780136006633, pp: 1104.

[4] Pallab banerjee, probal banerjee, shweta sonali dhal, "Performance Evaluation of a New Proposed Average Mid Max Round Robin (AMMRR) Scheduling Algorithm with Round Robin Scheduling Algorithm",IJARCSSE,ISSN: 2277-128X, Volume-2, Issue-8, August 2012.

[5] Sarojhiranwal and D.r. K.C.Roy"Adaptive Round Robin Scheduling using Shortest Burst Approach Based on Smart Time Slice".volume 2,No. 2,July-Dec 2011,pp. 319-32.

[6] Pallab banerjee, probal banerjee, shweta sonali dhal, "Comparative Performance Analysis of Mid Average Round Robin Scheduling Algorithm (MARR) using Dynamic Time Quantum with Round Robin Scheduling Algorithm having static Time Quantum",IJECSE,ISSN: 2277-1956, Volume-1, Issue-4, August 2012.

[7] H. S. Behera, R. Mohanty, and D. Nayak, "A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis," vol. 5, no. 5, pp. 10-15, August 2010.

[8] Sanjay Kumar Panda and Saurav Kumar Bhoi, "An Effective Round Robin Algorithm using Min-Max Dispersion Measure" ISSN : 0975-3397 ,Vol. 4 No. 01, January 2012.

[9] "Tarek Helmy, Abdelkader Dekdouk" Burst Round Robin: As a Proportional-Share Scheduling Algorithm, IEEE Proceedings of the fourth IEEE-GCC Conference on towards Techno-Industrial Innovations, pp. 424-428, 11-14 November,2007.

[10] Yaashuwanth .C & R. Ramesh "Inteligent time slice for round robin in real time operating system", IJRRAS 2 (2), February 2010.

[11] R. J. Matarneh, "Seif-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now RunningcProceses", American Journal of Applied Sciences 6 (10), pp. 1831-1837, 2009.

[12] H. S. Behera, Rakesh Mohanty, Sabyasachi Sahu and Sourav Kumar Bhoi. "Comparative performance analysis of multi-dynamic time quantum round robin (mdtqrr) algorithm with arrival time", ISSN : 0976-5166, Vol. 2, No. 2, Apr-May 2011,pp.262-271.

