Volume 7, Issue 5, May 2017 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Generic Round Robin Scheduling for Real Time Systems

Prof. Pallab Banerjee[1], Riya Shree[2], Richa Kumari Verma[3]
[1]Assistant Professor, [2, 3]MCA Scholar
[1, 2, 3]Department of Computer Science and Engineering, Amity University Ranchi, Jharkhand, India

DOI: 10.23956/ijarcsse/SV7I5/0125

Abstract: Round Robin Scheduling algorithm is designed especially for time sharing Operating system (OS).Jobs get the CPU for a fixed time (quantum time or time slice).Similar to FCFS (first come first serve scheduling),but with preemption (that is CPU interrupted at regular intervals).Ready queue is treated as a circular buffer. Process may use less than a full time slice. If process is incomplete at the end of time slice, they join in the end of ready queue. The Round Robin Scheduling algorithm has its disadvantages that is its longer average waiting time, higher context switches, higher turnaround time .In this paper a new algorithm is presented called Generic scheduling algorithm .In this scheduling algorithm the main idea is to adjust the time Quantum dynamically so that generic algorithm perform better performance than simple Round Robin scheduling algorithm.

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

I. INTRODUCTION

An operating system is an interface between computer user and computer hardware. An Operating system is software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers. Modern operating system and time sharing system are more complex, they have evolved from a single task to multitasking environment in which processes run in synchronized manner. Objective of multiprogramming is to maximize resource utilization, not possible to achieve without proper scheduling. All resources are scheduled before use. In a multiprocessing and multitasking environment if several processes are ready to run at the same time, the system must choose among them and assigned to run on the available CPUs, is called CPU scheduling. Allocating CPU to a process requires careful awareness to assure justice and avoid process starvation for CPU. Scheduling decision try to reduce the following: turnaround time, response time and average waiting time for processes and number of context switches. CPU scheduling algorithm decides which of the processes in the Ready Queue(RQ) are to be allocated to the CPU. There are many different CPU scheduling algorithms used like FCFS, SJF, RR, Priority scheduling algorithm and Short Remaining Time Next (STRN) Remaining Time Next (STRN) algorithm. The processes are scheduled according to the given burst time, arrival time, time quantum and priority. Out of those algorithms, Round Robin (RR) is the oldest, simplest and most widely used proportional share scheduling algorithm. It is similar to FCFS scheduling, but preemption is added to switch between processes. In Round Robin algorithm a small unit of time slice are required which is called Time Quantum (TQ). The CPU scheduler goes around Ready Queue and allocates the CPU to each processes by the help of Dispatcher for a time interval of up to 1 Time Quantum(TQ). If new process arrives then it is added to the tail of Circular Queue. The CPU scheduler picks the first process from the Ready Queue sets a timer to interrupt after one Time Quantum and dispatches the process. After TQ is expired, the CPU preempts the process and the process is added to the tail of the Circular Queue. If the process finishes before the end of the TQ, the process itself preempts the CPU willingly[3].In this paper, we tried to solved the Time Quantum problem by adjusting the Time Quantum Dynamically with respect to the existed set of processes in Ready Queue

II. PRELIMINARIES

Program is a specific set of ordered operation for a computer to perform. A process is an instance of a program running in a computer. 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. 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 process 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.

The scheduler is mainly concerned with:
Arrival time:-The time at which process arrives in main memory.
Burst time: - The time for which a process holds the CPU.
Waiting time: - The amount of time, process waiting in the Ready Queue.
Turnaround time: - The total time between arrival of a process and process completion.
Response time: - The time from the submission of a request by a process till its first response

III. PROPOSED ALGORITHM

1. Initialize
CS=0, AWT=0, ATT=0, i=0,S1=0,c1=0,j=1,n=100,S2=0,c2=0,p[n], TimeQuantum[5], sum=0
2. Sort the process in the ready Queue in ascending order of their BT.
//n = number of processes and i = loop variable
3. while (RQ != NULL)
//RQ = Ready Queue
a). TQ_{MM} = MAXBT – MINBT
//TQ_{MM} = Time Quantum
//MAXBT = MAXimum Burst Time
//MINBT = MINimum Burst Time (Remaining burst time of the processes)
4. If one process is there then TQ_{MM} is equal to BT of itself
5. TimeQuantum[ ]=TQ_{MM}
6. While(RQ != NULL)
//TQ_H=Hybrid Time Quantum.
// M1= Median of all the available processes in the Ready Queue arranged in Ascending Order.
// M2= median of the processes starting from median M1 to the last process.
7. Find median of the processes(M1).
// use round off function in M1.
8. for(i=M1 to 'n' Loop)
{
//calculate median(M2).
// use round off function in M2.
}
9. TQ_H=(M1+M2)/2.
10. TimeQuantum[]=TQ_H
11. while(i< N loop)
{
a)S1=S1+P[i]
b) i=i+2
c) c1++
}
//End of while.
12. TQ1=S1/c1;
13. while(j<N loop)
{
a) S2=S2+P[j]
b) j=j++;
c) c2++
}
//End of while.
14. TQ2=S2/c;
15. If(TQ1>=TQ2)
16. TQ=TQ1
Else
17. TQ=TQ2
18. TimeQuantum[]=TQ
19. Calculate the average of BT of the last three processes and put them in Avg.
20.For (i=N-2 to N loop)
{
a) sum=sum+ BT(Pi);
}
21. Avg=sum/3;
//Avg=Average of last three process BT.
22. TQ_{BPRR} = Avg;
23. TimeQuantum[]=TQ_{BPRR}

//end of for

24. If two processes are there then the TimeQuantum[ ]= is equal to the average of BT of the two processes.

25. If one process is there then after calculation TQ is equal to BT itself

i.e TimeQuantum[ ]=BT.

26. while(RQ!=NULL)

//MAXBT=Maximum Burst Time

27. AVG= (Sum of BT of all the process)/Number of processes.

//(Use round off function in AVG.)

28. TQ1=(AVG+MAXBT)/2

29. TimeQuantum[ ]=TQ1

//(Use round off function in TQ1) (Remaining Burst time of the process)

30. If one process is there then TimeQuantum= BT itself

Let us take

31. Large_TQ=TimeQuantum[1];

32.Repeat for i=1 to 5-1

{

a)If(TimeQuantum[i]>Large_TQ)

Then

b) Large_TQ=TimeQuantum[i];

}

33. Repeat for i=1 to N loop

// Assign Large_TQ to (1 to N)processes.

{

a) Pi->Large_TQ

//Assign TQ to all the available processes.

}

//End of for.

34. Calculate the remaining Burst time of the processes.

35. If (new process arrived and BT!=0)

then go to step1

else

36. if (new process is not arrived and BT!=0)

then go to step 2

37. if (new process is arrived and BT != 0)

then go to step 1

else

38. Calculate ATT, AWT and CS.

//ATT = Average Turnaround Time /

/AWT = Average Waiting Time

//CS = number of Context Switches

39. End

IV. EXPERIMENTAL ANALYSIS

Case 1: Let's consider five processes with Burst time (P1=15, P2=33, P3=45, P4=60, P5=75) with Arrival Time =0 as shown in the Table 1. Table 2 shows the output using RR , EORR and GRR algorithms. Figure1,2,3 shows the Gantt chart of both RR , EORR and GRR algorithm respectively.

Table1.Process with Burst Time

Processes	Arrival Time	Burst Time
P1	0	15
P2	0	33
P3	0	45
P4	0	60
P5	0	75

Table 2: Comparison between RR algorithm, EORR algorithm and new proposed Generic algorithm (GRR)

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	143.4	97.8	12
EORR	47,28	116.8	71.2	6
GRR	61,14	107.4	61.8	5

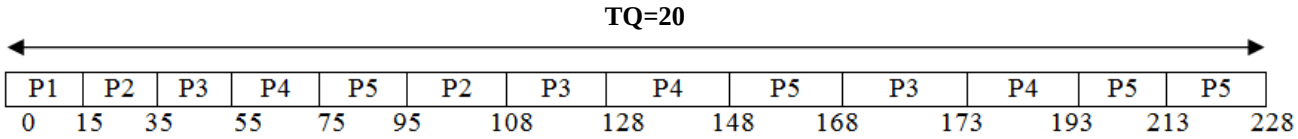

Fig.1: Gantt chart of RR from Table 1 of CASE 1.

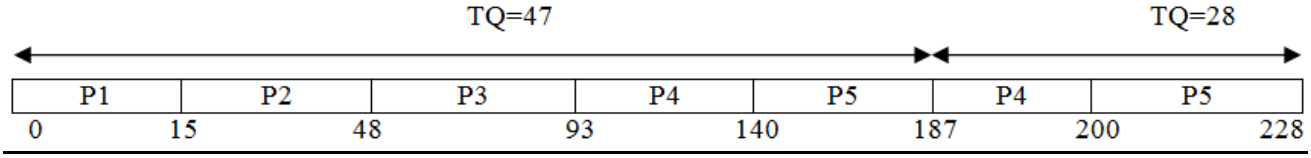

Fig.2: Gantt chart of EORR from Table 1 of CASE 1.

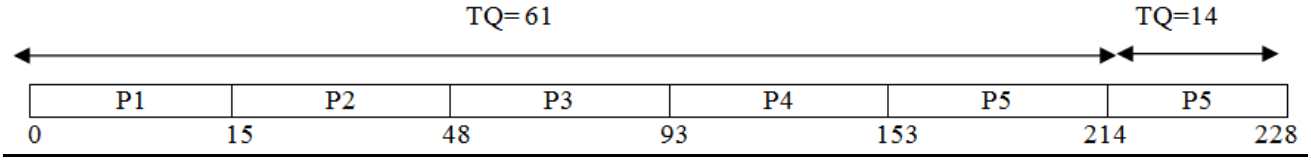

Fig.3: Gantt chart of GRR from Table 1 of CASE 1.

Case2: Let's consider five processes with Burst time (P1=41, P2=42, P3=43, P4=44, P5=45) with Arrival Time =0 as shown in the Table 3 . Table 4 shows the output using RR , EORR and GRR algorithms. Figure 4,5,6 shows the Gantt chart of both RR , EORR and GRR algorithms respectively.

Table3.Process 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, EORR algorithm and new proposed Generic algorithm

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	206.4	164	14
EORR	43	135.6	92.6	6
GRR	44	127	84	5

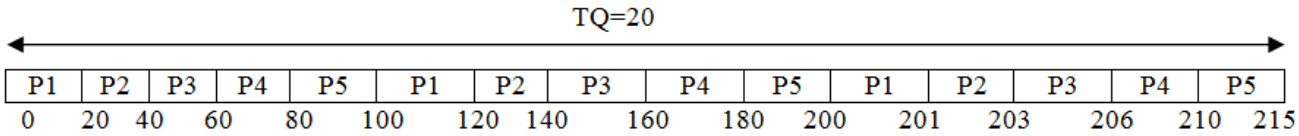

Fig.4: Gantt chart of RR from Table 3 of CASE 2.

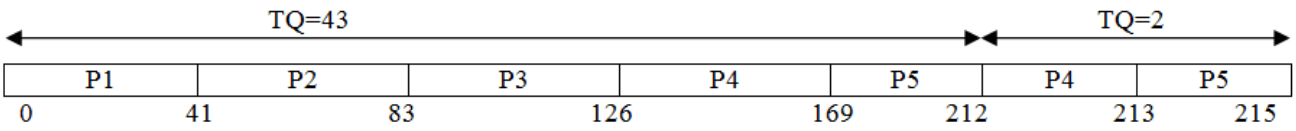

Fig.5: Gantt chart of EORR from Table 3 of CASE 2.

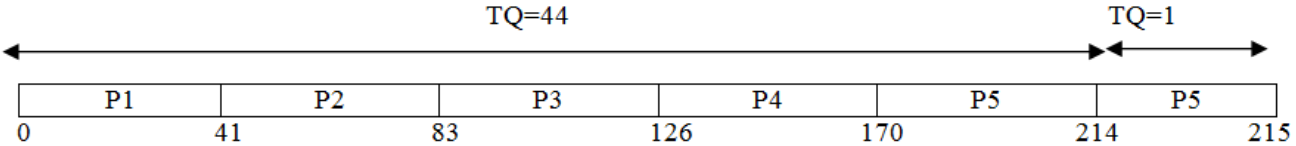

Fig.6: Gantt chart of GRR from Table 3 of CASE 2.

Case3: Let's consider five processes with Burst time (P1=10, P2=20, P3=40, P4=80, P5=160) with Arrival Time =0 as shown in the Table 5 . Table 6 shows the output using RR , EORR and GRR algorithms. Figure 7,8,9 shows the Gantt chart of both RR , EORR and GRR algorithms respectively.

Table5.Process with Burst Time

Processes	Arrival Time	Burst Time
P1	0	10
P2	0	20
P3	0	40
P4	0	80
P5	0	160

Table 6: Comparison between RR algorithm, EORR algorithm and new proposed Generic algorithm

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	134	72	11
EORR	83,77	128	66	7
GRR	111,49	114	52	5

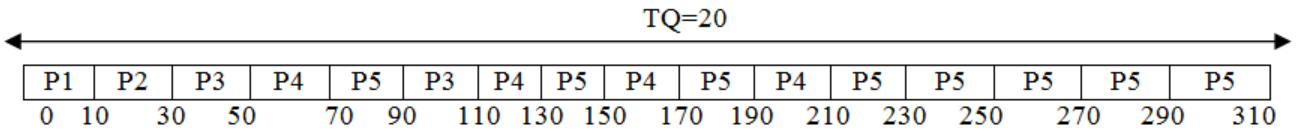

Fig.7: Gantt chart of RR from Table 5 of CASE 3.

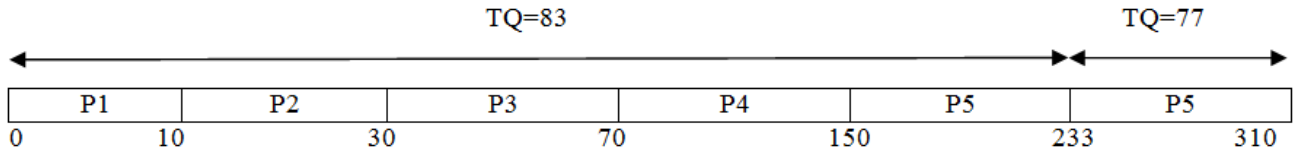

Fig.8: Gantt chart of EORR from Table 5 of CASE 3.

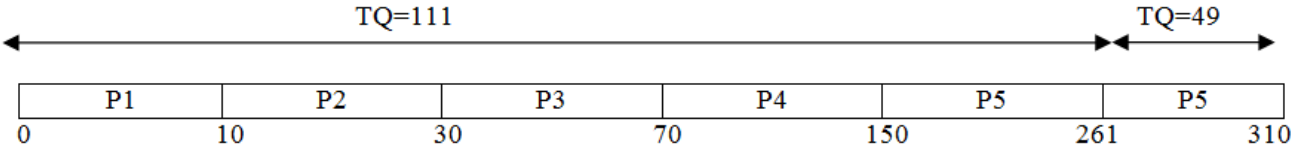

Fig.9: Gantt chart of GRR from Table 5 of CASE 3.

Case 4: Let's consider five processes with Burst time (P1=14, P2=25, P3=35, P4=47, P5=62) with Arrival Time (P1=0,P2=9,P3=11,P4=14,P5=18) as shown in the Table 7 . Table 8 shows the output using RR , EORR and GRR algorithms. Figure 10,11,12 shows the Gantt chart of both RR , EORR and GRR algorithms respectively.

Table7.Process with Burst Time

Processes	Arrival Time	Burst Time
P1	0	14
P2	9	25
P3	11	35
P4	14	47
P5	18	62

Table 8: Comparison between RR algorithm, EORR algorithm and new proposed Generic algorithm

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	83.8	49.2	9
EORR	37,25	83.2	46.6	7
GRR	50,12	75.8	39.2	6

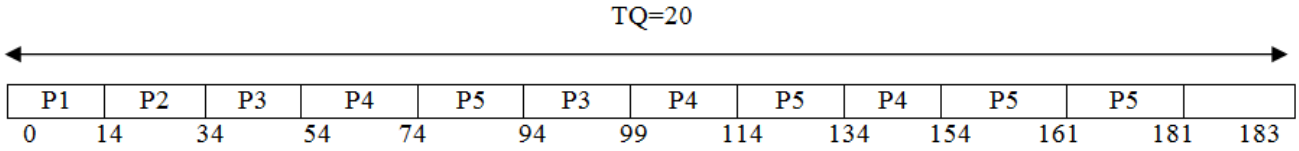

Fig.10: Gantt chart of RR from Table 7 of CASE 4.

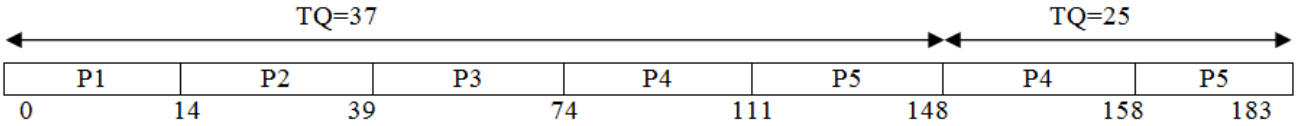

Fig.11: Gantt chart of EORR from Table 7 of CASE 4.

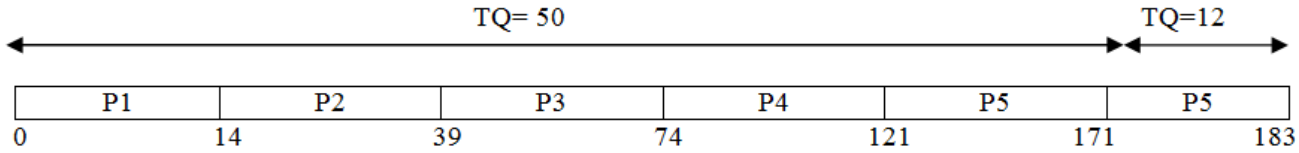

Fig.12: Gantt chart of GRR from Table 7 of CASE 4.

Case 5: Let's consider five processes with Burst time (P1=27, P2=35, P3=39, P4=46, P5=55) with Arrival Time (P1=0,P2=13,P3=15,P4=19,P5=25) as shown in the Table 9 . Table 10 shows the output using RR , EORR and GRR algorithms. Figure 13,14,15 shows the Gantt chart of both RR , EORR and GRR algorithms respectively.

Table9.Process with Burst Time

Processes	Arrival Time	Burst Time
P1	0	27
P2	13	35
P3	15	39
P4	19	46
P5	25	55

Table 10: Comparison between RR algorithm, EORR algorithm and new proposed Generic algorithm

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	137.4	97	11
EORR	41,14	101.6	71.2	6
GRR	48,7	93.4	53	5

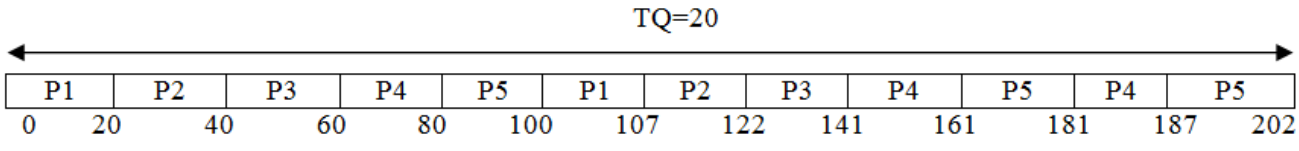

Fig.13: Gantt chart of RR from Table 9 of CASE 5.

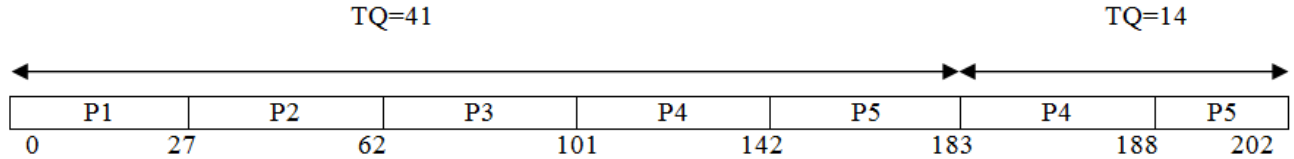

Fig.14: Gantt chart of EORR from Table 9 of CASE 5.

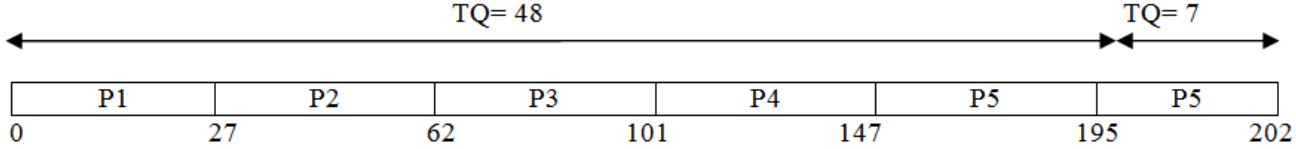

Fig.15: Gantt chart of GRR from Table 9 of CASE 5.

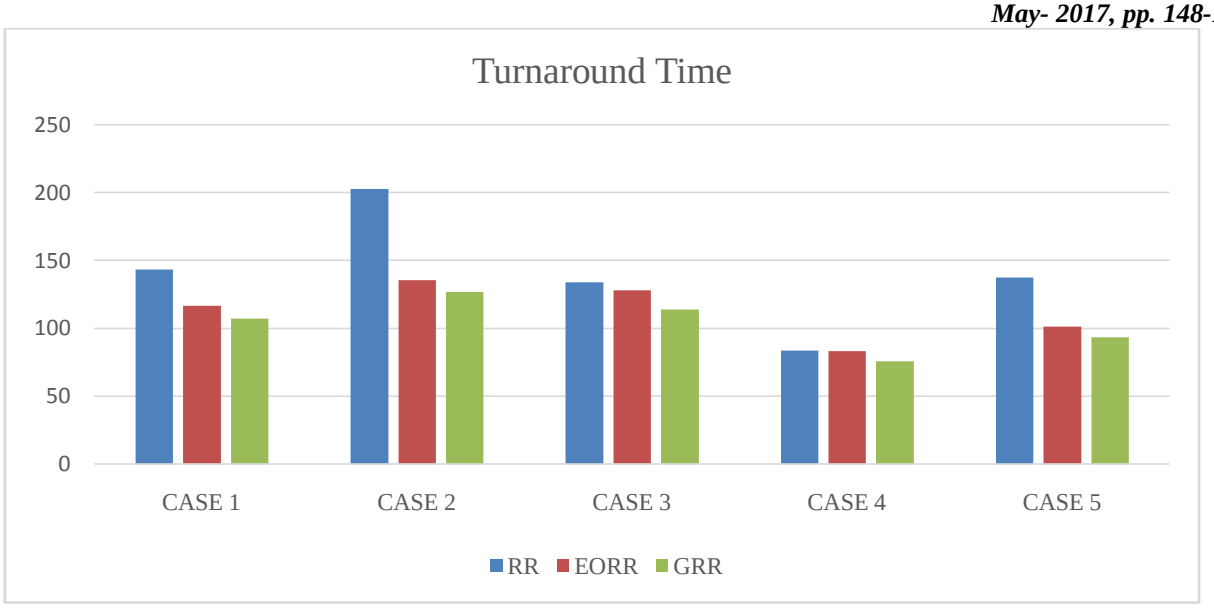

Fig.16: Comparison of average Turnaround Time of RR , EORR and GRRR taking arrival time into consideration.

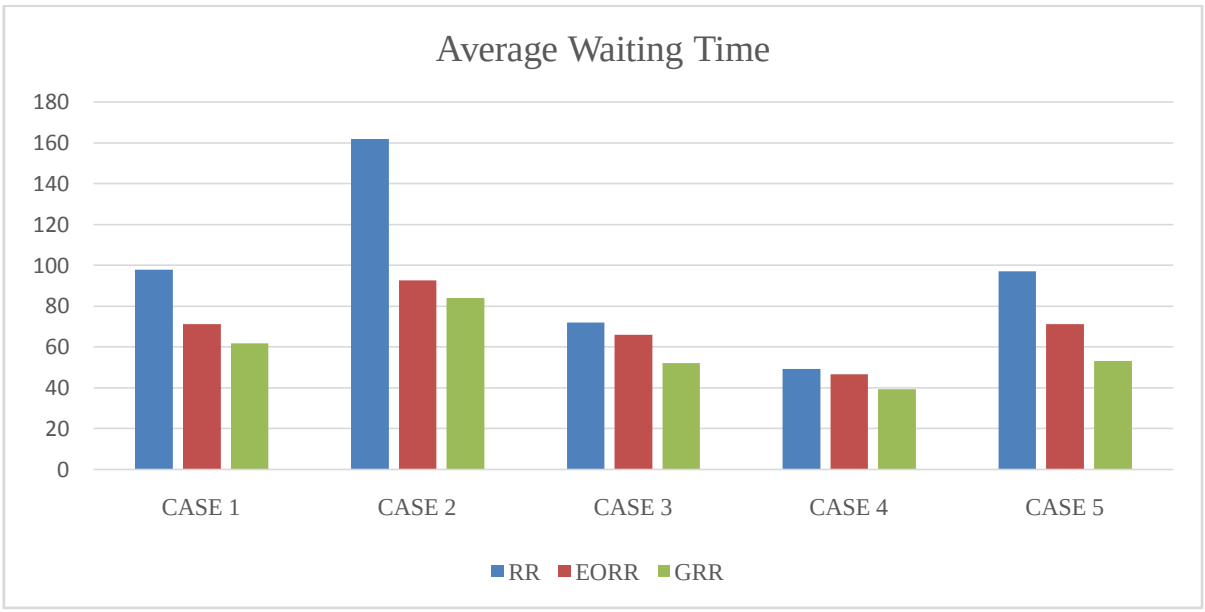

Fig.16: Comparison of average Waiting Time of RR , EORR and GRRR taking arrival time into consideration.

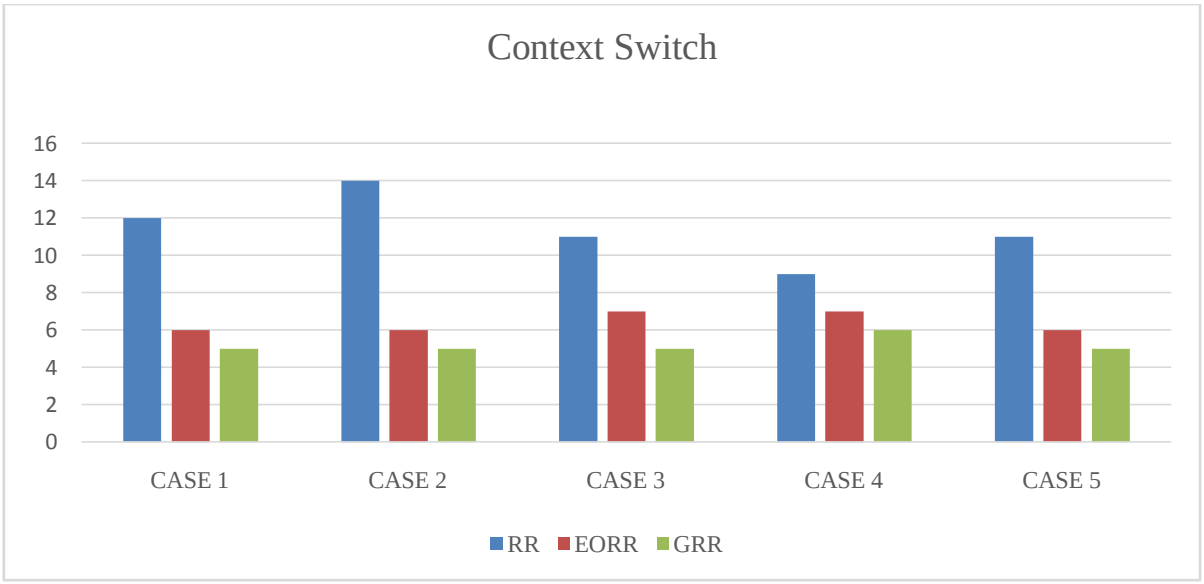

Fig.16: Comparison of Context Switching of RR , EORR and GRRR taking arrival time into consideration.

V. CONCLUSION

In a computer system, the main role is of the CPU. One of the important work of CPU is to perform multitasking. The ability to execute more than one task at the same time is referred as multitasking. In multitasking, only one CPU is involved, but it switches from one program to another so quickly that it gives the appearance of executing all of the programs at the same time. The objective of multitasking is to increase the efficiency of the CPU and to reduce the average waiting time of the system. Many CPU scheduling algorithms have been presented with some advantages and disadvantages. An existing round robin does not provide a better performance, so an improved round robin CPU scheduling algorithm with varying time quantum is proposed called Generic Round Robin. The paper presents a new CPU scheduling algorithm. Comparisons of various algorithms that is round robin, even odd round robin (EORR) and the proposed algorithm that is generic round robin (GRR) has been done. It is concluded that the proposed algorithm is more efficient than round robin and even odd algorithm because of its less average waiting time, less average turnaround time and less number of context switches as compared to round robin, so it reduces the operating system overhead

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] Pallab banerjee, probal banerjee, shweta sonali dhal,"Comparative Performance Analysis of Even Odd Round Robin Scheduling Algorithm (EORR) using Dynamic Time Quantum with Round Robin Scheduling Algorithm using static Time Quantum" IJARCSSE,ISSN: 2277-128X,Volume-2, Issue-9, August 2012.

[6] Pallab banerjee, probal banerjee, shweta sonali dhal,"Improved High Performance Round Robin Scheduling Algorithm(HPRR)using Dynamic Time Quantum" International Journal of Computer Information System,ISSN: 2277-128X,Volume-5, No-3, 2012.

[7] 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.

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

[9] 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.

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

[11] 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.

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

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

[14] H. S. Behera, Rakesh Mohanty, Sabyasachi Sahu and Sourav Kumar Bhoi. "Comparative performance analysis of multidynamic time quantum round robin (mdtqrr) algorithm with arrival time", ISSN : 0976-5166, Vol. 2, No. 2,Apr- May2011,pp.262-271.

[15] Pallab banerjee ,Prof Dr L.N.Padhy. "Comparative analysis of Maximum performance round robin(MPRR) by Dynamic Time Quantum with static time quantum",ISSN:2277- 128X,pg.372-377Vol-4,Issue-11,Nov-2014

[16] Pallab banerjee,Talat Zabin,Shweta Kumari,Pushpa Kumari "Comparative performance analysis of best performance of round robin scheduling algorithm(BPRR) using Dynamic Time quantum with priority based round robin(PBRR) CPU Scheduling algorithm in Real Time System".ISSN:2277- 1956,Vol-4,Number-2, pg. 151-159,May 2015.

[17] R.Nallakumar ,Dr.N.Sengottaiyan ,S.Nithya "A Servey of Task Scheduling Methods in Cloud Computing "IJCSE ,ISSN :2347-2693 ,Volume-2,Issue-1,2014

[18] Abbas Noon,Ali Kalakech,Seifedine Kadry"A New Round Robin Scheduling Algorithm for Operating Systms: Dynamic Quantum Using Mean Average"IJCSI,ISSN:1694- 0814,Vol-8,Issue-3,No-1,May-2011

