International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 3, March - 2014

Modified RR Algorithm with Dynamic Time Quantum for Externally Prioritized Tasks

Lipika Datta
Computer Science and Engineering Department,
College of Engineering and Management Kolaghat
KTPP Township, West Bengal, India

***Abstract*—The objective of this paper is to modify Round Robin scheduling for processes with user defined external priorities so that the scheduling algorithm can be used to schedule processes of soft real time systems. Simple Round Robin scheduling algorithm and Priority scheduling algorithm, both have their own drawbacks. In the paper a new scheduling algorithm is introduced that will be appropriate for scheduling interactive processes as well as will take into consideration the externally defined priority of a process at the same time. It reduces the average response time of the system and also variance in response time to improve predictability of the system. Experimental analysis reveals that the proposed algorithm gives better response time and less number of context switches than some existing algorithms useful for interactive systems.**

Keywords*— *Operating System; Scheduling; Round Robin Algorithm; Context switch; Response time; Variance in Response time;

I. INTRODUCTION

In a multiprocessing and multitasking environment the scheduler selects from among the processes in memory that are ready to execute, and allocates the CPU to one of them. CPU scheduling algorithm decides which of the processes in the Ready Queue (RQ) is to be allocated to the CPU. First Come First Serve (FCFS), Shortest Job First (SJF), Round Robin (RR), Priority scheduling algorithm and Shortest Remaining Time Next (STRN) are some well known CPU scheduling algorithms. Each of the algorithms has some advantages and disadvantages. The selection of scheduling algorithm depends on the workload and environment.

Ideally we want to maximize the CPU utilization and throughput and minimize Average waiting time, average turnaround time and response time. To guarantee that all users get good service, we try to minimize the maximum response time. But this is not always possible. Instead, we choose a scheduling algorithm based on its ability to satisfy a policy.

- Minimize average response time - provide output to the user as quickly as possible and process their input as soon as it is received.
- Minimize variance of response time - in interactive systems, predictability may be more important than a low average with a high variance.
- Maximize throughput by minimizing overhead (OS overhead, context switching) and by efficient use of system resources (CPU, I/O devices)
- Minimize waiting time by giving each process the same amount of time on the processor. This might actually increase average response time.

In time-sharing (multi-tasking) systems, we try to minimize the variance in the response time. Variance in response time is viewed as more appropriate measure of the quality of service (predictability). Minimum variance implies a higher quality of service. So predictability is a dominant performance metric in real time systems [1].

II. RELATED WORK

RR algorithm performs optimally in timeshared systems, but it is not suitable for soft real time systems. Because it gives more number of context switches, larger waiting time and larger response time. Some researchers have already introduced some variations of RR scheduling algorithm. But these algorithms have some limitations. In [2] authors have proposed an algorithm in which according to the given priority the CPU is allocated to the processes only once in RR fashion for a given time quantum. Then processes are arranged in increasing order of their remaining CPU burst time in the ready queue. New priority is assigned to each process following the rule that lesser the remaining burst time higher the priority. Then processes are allocated CPU according to non-preemptive priority scheduling algorithm. If this algorithm is used after first response from the system user may have to wait long for next response. Fairness criteria is not held. In [3] different time slices are calculated for different processes based on three aspects: user defined priority, average CPU burst, context switch avoidance time. An assumption is made on average CPU burst. In [4] also different time slices are calculated for different processes based on priority, shortest CPU burst time and context switch avoidance time. In different rounds the time quantum for a process goes on changing depending on the parameters i.e. the authors have introduced dynamic time quantum. The authors only have taken into consideration the processes with highest priority. Rest of the processes' priorities is ignored. In [5] the authors have introduced a concept called intelligent time slicing which depends on priority and context switch. The time slice is static. This algorithm is modified to get different time slice values in different rounds for different processes in [6]. It calculates the initial time slice for each process as the previous algorithm [5] and in each round the time slices are modified. In [7] the authors have made the priority and time slice for a process dynamic by calculating the weighted mean values of time quantums and priorities of the processes and considering the burst time of the processes.

A. My contribution

In this paper, I have proposed an improved algorithm as compared to the algorithm defined in [2],[3],[4],[5],[6] in terms of variance in response time and number of context switch.

B. *Organization of paper*

Section III presents the illustration of my proposed algorithm. In section IV, Experimental results and its comparison with existing algorithms is presented.

III. PROPOSED ALGORITHM

In my work the processes are allocated CPU according to their priority defined by user. Static time being a limitation of RR scheduling algorithm, in each round the time quantum is changed so that processes with smaller remaining CPU bursts can have enough time to complete their execution in that round.

The processes are arranged in the ready queue according to the user defined priority, i.e. the process with highest priority will be placed at the head of the ready queue. In the first round each process is allocated the CPU in the order of their increasing priority for a fixed time quantum. The processes which complete their execution within the time quantum are removed from the ready queue. If new process arrives in between it is placed in the ready queue in proper position according to its priority. The average of the remaining burst times of the existing processes is calculated and that is the time quantum for the next round. This procedure continues until the ready queue becomes empty.

1. Input: Process no, Burst time, Process priority, Original time slice.
   Output: Average response time, Variance in Response time, Number of Context Switches
2. Initialize TQ= Original time slice, i=1
3. Sort the processes in the ready queue (RQ) according to their priorities.
4. while (RQ!=NULL)
   {
   if (i==1)
   allocate CPU to processes in the RQ according to RR scheduling algorithm with time quantum TQ.
   else
   {
   from RQ remove processes whose remaining burst time == 0.
   if new process arrives place it in proper position in RQ according to its priority.
   assign RT = sum of remaining burst times of existing processes.
   assign TQ = RT/m where m is number of processes in RQ.
   Increment i by 1
   }
   }
5. Calculate Average response time, variance in response time, number of context switch.
   End

IV. EXPERIMENTAL RESULTS

A. *Assumptions:*

Experiments are performed in single processor environment and on independent processes. Original time slice is not more than maximum burst time. All the parameters like number of processes, and burst time, priority of all the processes are known before submitting the processes to the processor. All processes are CPU bound and none I/O bound. Context switching overhead is zero wile calculating Average response time and variance in response time.

B. *Data Set:*

To compare the performance of the algorithm with the algorithms described in [3] (SARTT) and [4] (PBDRR) 3 different data sets are taken as the processes with burst time in increasing, decreasing and random order respectively. For all the cases arrival time is considered as 0 and original time slice is considered as 4 time unit. Again comparison is done between algorithms introduced in [5] (MRR) and [6] (TSPBRR) with a different set of data.

1) Same data set applied to SARTT, PBDRR AND Proposed Algorithm:

a) *Case 1: Processes with increasing Burst Time:*

TABLE 1. Inputs for case 1

Process id	Priority	Burst time
P1	2	5
P2	3	12
P3	1	16
P4	4	21
P5	5	23

TABLE 2. Calculation of time quantum for proposed algorithm for case 1

Process id	Priority	Burst Time	Remaining Burst Time after each round			
			1st (TQ=4)	2nd (TQ=12)	3rd (TQ=6)	4th (TQ=1)
P3	1	16	12	0	0	0
P1	2	5	1	0	0	0
P2	3	12	8	0	0	0
P4	4	21	17	5	0	0
P5	5	23	19	7	1	0

From Fig 1, Fig 2 and Fig 3 Average Response time, Variance in Response Time and number of context switching can be calculated for SARTT, PBDRR and the proposed algorithm for processes with increasing burst time. Table 3 shows the comparison among the algorithms.

Table 3. Comparison between algorithms for case 1

Algorithm	Average Response Time	Variance in Response Time	No. of Context Switch
SARTT	9.2	40.56	19
PBDRR	6.8	49.71	16
Proposed Algorithm	8	32	12

b) *Case2: Processes with decreasing burst time:*

TABLE 4. Inputs for case 2

Process id	Priority	Burst time
P1	2	31
P2	1	23
P3	4	16
P4	5	9
P5	3	1

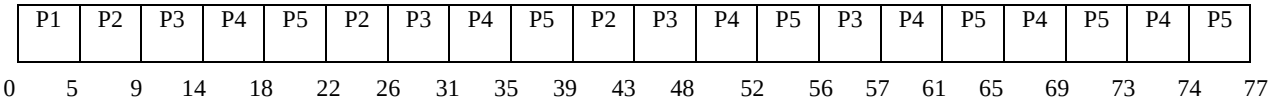

Figure1. Gantt chart for SARTT (case 1)

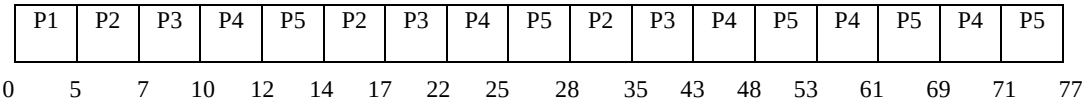

Figure2. Gantt chart for PBDRR (case 1)

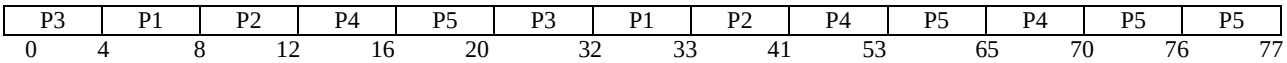

Figure3. Gantt chart for Proposed Algorithm (case 1)

Table 5. Calculation of time quantum for proposed algorithm for case 2

Process id	Priority	Burst Time	Remaining Burst Time after each round			
			1st (TQ=4)	2nd (TQ=16)	3rd (TQ=7)	4th (TQ=4)
P2	1	23	19	3	0	0
P1	2	31	27	11	4	0
P5	3	1	0	0	0	0
P3	4	16	12	0	0	0
P4	5	9	5	0	0	0

From Fig 4, Fig 5 and Fig 6 Average Response time, Variance in Response Time and number of context switching can be calculated for SARTT, PBDRR and the proposed algorithm for processes with decreasing burst time. Table 6 shows the comparison among the algorithms.

TABLE 6. Comparison between algorithms for case 2

Algorithm	Average Response Time	Variance in Response Time	No. of Context Switch
SARTT	9.8	52.16	18
PBDRR	8.2	44.96	11
Proposed Algorithm	6.8	19.76	11

c) Case 3:Processes with Random Burst Time:

TABLE 7. Inputs for case 3

Process id	Priority	Burst time
P1	3	11
P2	1	53
P3	2	8
P4	4	41
P5	5	20

Table 8. Calculation of time quantum for proposed algorithm for case 3

Process id	Priority	Burst Time	Remaining Burst Time after each round			
			1st (TQ=4)	2nd (TQ=23)	3rd (TQ=20)	4th (TQ=6)
P2	1	53	49	26	6	0
P3	2	8	4	0	0	0
P1	3	11	7	0	0	0
P4	4	41	37	14	0	0
P5	5	20	16	0	0	0

From Fig 7, Fig 8 and Fig 9 Average Response time, Variance in Response Time and number of context switching can be calculated for SARTT, PBDRR and the proposed algorithm for processes with random burst time. Table 9 shows the comparison among the algorithms.

TABLE 9. Comparison between algorithms for case 3

Algorithm	Average Response Time	Variance in Response Time	No. of Context Switch
SARTT	10.2	61.31	29
PBDRR	7	35.6	18
Proposed Algorithm	8	32	12

2) Same data set applied to MRR, TSPBRR AND Proposed Algorithm:

TABLE 10. Input data set

Process id	Priority	Burst time
P1	3	11
P2	1	53
P3	2	8
P4	4	41
P5	5	20

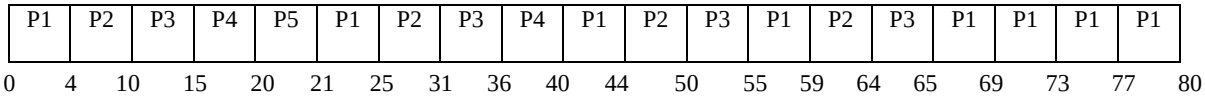

Figure4. Gantt chart for SARTT (case 2)

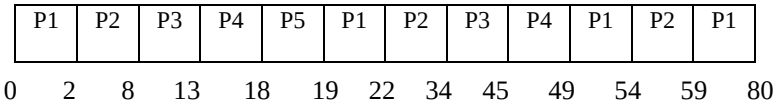

Figure5. Gantt chart for PBDRR (case 2)

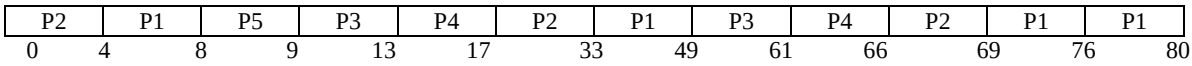

Figure6. Gantt chart for Proposed Algorithm (case 2)

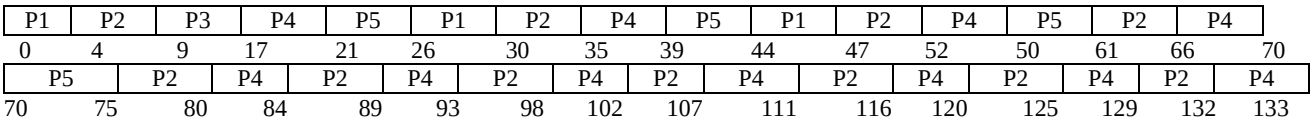

Figure7. Gantt chart for SARTT (case 3)

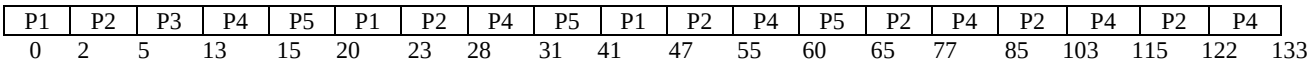

Figure8. Gantt chart for PBDRR (case 3)

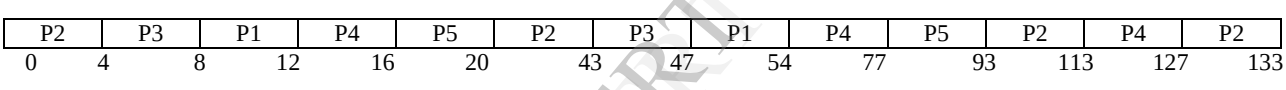

Figure9. Gantt chart for Proposed Algorithm (case 3)

TABLE 11. Comparison between algorithms

Algorithm	Average Response Time	Variance in Response Time	No. of Context Switch
MRR	16	139.6	11
TSPBRR	9.5	59.15	15
Proposed Algorithm	10	50	9

Fig 10,Fig 11,Fig 12 graphically represents the performances of the SARTT, PBDRR and proposed algorithms in terms of Average Response Time, Variance in Response Time and number of context switching. Fig 13 graphically represents the performances of the MRR, TSPBRR and proposed algorithms in terms of the same parameters.

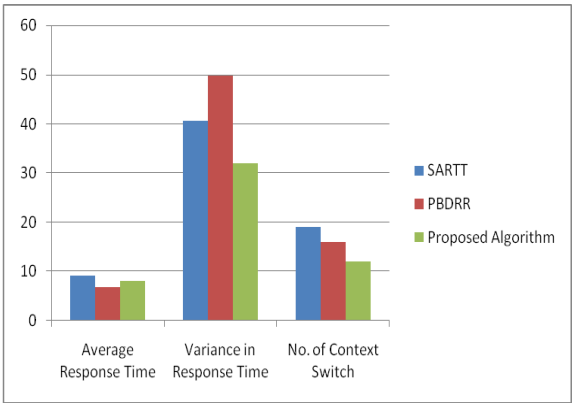

Figure10. Analysis of performance among algorithms (case 1)

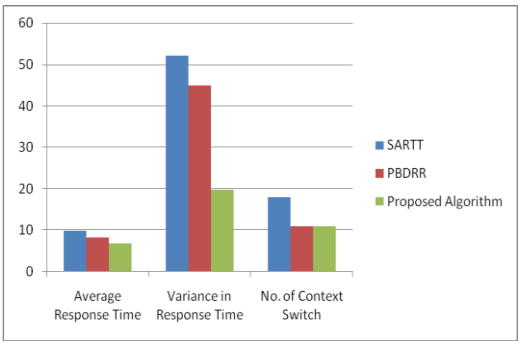

Figure11 .Analysis of performance among algorithms (case 2)

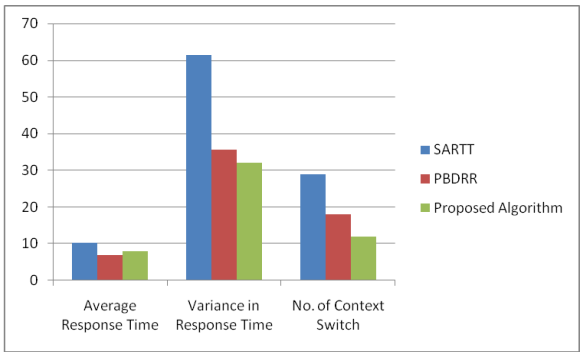

Figure12 .Analysis of performance among algorithms (case 3)

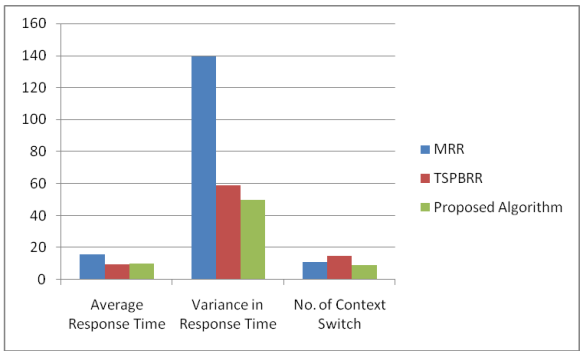

Figure13 .Analysis of performance among algorithms

CONCLUSION

From the above comparisons it can observed that the proposed algorithm is better than some existing algorithms for real time task scheduling in terms of variance in response time and number of context switches. So the quality of service can improved and overhead can be reduced. Thus memory space which is an important constraint for embedded system applications can be saved. Deadlines of tasks can be considered in future as a new parameter while calculating the time quantum in each round.

REFERENCES

1. Shibshankar Halder, Alex Alagarsamy Aravind "Operating Systems", isbn=8131730220
2. Ishwari Singh Rajput, Deepa Gupta "A Priority based Round Robin CPU Scheduling Algorithm for Real Time Systems", International Journal of Innovations in Engineering and Technology (IJIET) Vol. 1 Issue 3 Oct 2012
3. C.Yaashuwanth, Dr.R.Ramesh "A New Scheduling Algorithms for Real Time Tasks", International Journal of Computer Science and Information Security, Vol. 6, No.2, 2009
4. Rakesh Mohanty , H. S. Behera , Khusbu Patwari , Monisha Dash , M. Lakshmi Prasanna "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, February 2011
5. Yaashuwanth .C, Dr.R. Ramesh, " Design of Real Time scheduler simulator and Development of Modified Round Robin architecture" IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.3, March 2010
6. Subasish Mohapatra, Subhadarshini Mohanty, K.Smruti Rekha," Analysis of Different Variants in Round Robin Algorithms for Load Balancing in Cloud Computing, International Journal of Computer Applications (0975 – 8887) Volume 69– No.22, May 2013
7. H.S.Behera, Sabyasachi Sahu, Sourav Kumar Bhoi, "Weighted Mean Priority Based Scheduling for Interactive Systems" Journal of Global Research in Computer Science, Volume 2, No. 5, May 2011