Modified Round Robin Scheduling Algorithm by Dynamic Time Quantum

Keerthana Baskaran
Student, Department of Computer Science and Engineering,
Rajiv Gandhi College of Engineering and Technology,
Pondicherry University, India

Abstract: CPU scheduling has vital effect on the resources, their utilization and overall performance of the systems. Round Robin (RR) scheduling algorithm is widely used in multitasking. Choosing the time quantum in RR algorithm is very crucial as a small-time slice results in large number of context switches and large time quantum increases the response time and makes it more similar to First Come First Serve (FCFS) scheduling algorithm. A CPU scheduling processor is said to be optimally if it has a higher throughput, low waiting time, lower turnaround time and less number of context switches for the processes coming for execution. The main objective of this paper is to improve the working of round robin scheduling by dynamic time quantum adjustment along with the integrated priority scheduling algorithm. The time quantum is decided based on the number of process. If the number of process is less than or equal to 3 then normal FCFS approach along with priority, else time quantum will be number of process divided by 2. The ready queue has the processes in order of priority. The process that arrives first will have to priority as 1 by default. The scheduler will run same as round robin until the remaining burst time is less than or equal to the actual time quantum. In such case the process is allowed for full execution. The proposed algorithm produces better average turnaround time, average waiting time and fewer number of context switches than existing algorithms which is shown by the experimental analysis.

Keywords: **CPU Scheduling, Round robin scheduling, context switches, dynamic time quantum, average waiting time, average turnaround time.**

I. INTRODUCTION

In a single-processor system, only one process can run in the CPU at a time. Others processes in the ready queue must wait until the CPU becomes free and is rescheduled. In multi-processor system the scheduler will determine which process to run in a pool of multiple processes. The objective of multiprogramming is to have some process running at all times, to maximize CPU utilization. Execution of processes uses number of resources such as Memory, CPU etc. CPU scheduling is important because it can have a big effect on resource utilization and the overall performance of the system [1]. Scheduling algorithms are widely used in communications networks and in operating systems to allocate resources to competing tasks. There exists a variety of process scheduling algorithms and most of the operating systems are deploying more than one process scheduling algorithm for optimizing their processing needs. Most common ones are the First Come First Server Algorithm, Shortest Job First Algorithm, Priority Based Scheduling Algorithm. The round robin algorithm is the most used because of the performance issue in other scheduling algorithms[2]. In Round Robin (RR) every process has equal priority and is given a time quantum or time slice after which the process is preempted. Although RR gives improved response time and uses shared resources efficiently. Its limitations are larger waiting time, undesirable overhead and larger turnaround time for processes with variable CPU bursts due to use of static time quantum. This motivates us to implement RR algorithm with dynamic time quantum concept.

Basic Scheduling criteria

In choosing which algorithm to use in a particular situation, we must consider the different properties of the various algorithms [2]. Some of them are,

1. *Throughput:* Throughput is the number of processes completed per unit time. The scheduling algorithm should be designed in such a way that throughput in a system is maximized. Context switch and throughput and inversely proportional to each other.
2. *CPU Utilization:* This is a measure of how much busy the CPU is. The goal of scheduling is to keep the processor busy all the time. It is measured in percentage.
3. *Context Switch:* Context switching is the procedure of storing the state of an active process and restoring the state of another process (pre-empted process), which the CPU starts executing for the next time quantum. The scheduling algorithm should be designed such that the context switch is minimized.
4. *Turnaround time:* It refers to the total time, which is spent to complete the process, and the time taken to execute that particular process. It is generally measured as the time interval between the time of submission of a process to the time of completion of the process. For a good scheduler a turnaround time should be less.
5. *Waiting time:* Waiting time is the sum of the periods spent waiting in the ready queue.
6. *Response Time:* It is the time difference of the submission of job till the first response is produced.

Hence, a good scheduling algorithm for real time and time-sharing system must possess following characteristics [3].

a. Minimum context switches.

b. Maximum CPU utilization.

c. Maximum throughput.

d. Minimum turnaround time.

e. Minimum waiting time.

II. RELATED WORK

In the last few years different approaches are used to enhance the performance of Round Robin scheduling. Researchers are made to achieve low scheduling overhead with good fairness in dynamic environment in[1]. In Fair Priority Round Robin with Dynamic Time Quantum [2], the processes have been scheduled by giving importance to both the user priority and shortest burst time priority. In [3], an Improved Round Robin Scheduling Algorithm for CPU Scheduling is described which allocates the time quantum to all process only in the first cycle, later the use the SJF scheduling method to select the other process from the ready queue. In [4] the authors have arranged the processes in ascending order of burst time. Time slice is chosen as the CPU burst of the mid process in case of odd number of processes, otherwise time slice is equal to the average CPU burst of all running processes. For each round of RR algorithm, different Time Slice is used which is calculated based on the remaining burst time of currently running process in[5].The authors in [6] proposed an approach in which the process are arranged in ascending order of the burst time and then an optimal time quantum is calculated using the median concept. If the number of processes in the ready queue is odd, the burst time of the middle process will become the time quantum, otherwise the average of the two middle processes will become the time quantum. Mixed Scheduling (A New Scheduling Policy) [7], uses the job mix order for non-preemptive scheduling FCFS and SJF. According to job mix order, from a list of N processes, the process which needs minimum CPU time is executed first and then the highest from the list and so on till the nth process. The researchers in [8] proposed to increase the time quantum of the process, which does not complete its execution within the allotted time. Most of the abovementioned algorithms do not consider the waiting time of a process while calculating time slice for the next round of the RR scheduling.

III. PROPOSED ALGORITHM

The proposed algorithm focuses on the drawbacks of simple round robin architecture which gives equal priority to all the processes (processes are scheduled in first come first serve manner). Round robin architecture is not efficient for processes with smaller CPU burst because of this FCFS approach. This results in increasing the waiting time and response time of processes, which decreases the system throughput.

In our algorithm, the time quantum will be decided based on the number of processes. A decrease in process would increase the time quantum (FCFS approach), furthermore increase in process would decrease the time quantum (proposed algorithm). The algorithm also requires priority of each process and the default priority of the first entering process into ready queue would be set to 1. If the number of process is less than or equal to 3, then normal FCFS approach is done. Else if the number of process is greater than 3, the time quantum is decided by dividing the number of process by 2. While running for a particular process, and the process does not end before the given time quantum, then checks for the remaining time left to finish the process (remaining burst time). If the remaining burst time is less than the time quantum, then execute the entire job, else preempts the job. When the ready queue becomes less than or equal to 3, then again FCFS approach is used for the remaining number of process left to be scheduled.

Following is the way the algorithm works,

Terminology: TQ- Time Quantum, P- Process, N- Total number of process, BT- total burst time, R- remaining burst time after execution, RQ- Ready Queue, n[RQ]= total number of process in ready queue, BT[P]- Burst time of a particular process, R[P]- remaining burst time of a particular process.

Step 1: Check the total number of process,
If N<=3 then
FCFS approach based on priority
Step 2: Else if N>3 then
{
TQ = N/2
Step 3: Enter the process in the ready queue based on priority
Step 4: For all the process in the RQ, schedule the CPU for the given TQ
{
if BT[P]<TQ then complete execution
Step 5: else if BT>TQ then
{
Check if R[P]<TQ then
Complete the execution

Step 6: else preempt the process to the end of RQ
}
Step 7: Go to step 4 until n[RQ]<=3
}
Step 8: If n[RQ]<=3 then FCFS with priority approach
}

IV. EXPERIMENTAL ANALYSIS

Six processes have been defined with CPU burst time and their priorities, these six processes are scheduled in both round robin fashion and according to the proposed algorithm. The context switch, average waiting time, average turnaround time has been calculated and the results were compared. For doing this we have implemented the CPU scheduling algorithm in C and carried out number of experiments out of which only three experiments are discussed here for varying time quantum and we assure that the results analysis are remain unchanged for others.

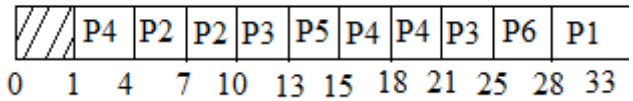
1. Consider the process P1-P6 with following details as shown in table

Process	Priority	Arrival Time	Burst Time
P1	6	5	5
P2	2	4	6
P3	3	3	7
P4	1	1	9
P5	4	2	2
P6	5	6	3

TQ = number of process/2 = 3

(i) Initially the ready queue will have only P4 at arrival time 1.
(ii) By the time one cycle of TQ of P4 completes, 3 more processes are arrived at time 2, 3, 4 and the process are P5, P3, P2 respectively.
(iii) They are arranged in the order of priority. Now the ready queue has P4, P2, P3, P5.
(iv) The remaining 6 ms of P4 are added at the end of Ready queue after P5. And now the ready queue is P4, P2, P3, P5, P4
(v) The next process to be processed is P2 from ready queue. After completion of 3ms of TQ, the remaining time left to be processed in P2 is 3 which is equal to TQ. So Process P2 will be allowed to complete its execution.
(vi) By the end of P2 completion 2 more processes are added to the ready queue at time 5 and 6 which are P1 and P6 respectively.
(vii) Among P1 and P6, P6 has highest priority than P1 so the order is changed. Now the ready queue looks like P4, P2, P3, P5, P4, P6, P1
(viii) After execution of P2, the next process from ready queue is taken. P3 will finish one cycle of TQ and then the process is preempted. The ready queue is P4, P2, P3, P5, P4, P6, P1, P3
(ix) Next process from ready queue is P5 which gets completely processed within the time quantum
(x) P4 is taken next which runs for one TQ cycle and then allowed to run for next cycle since the left over time is less than or equal to the time quantum
(xi) Now there are only 3 processes left over in the ready queue which as P6, P1, P3 with priority 5, 6, 3 respectively.
(xii) Since the number of process is less than or equal to 3, these processes get processed based on priority FCFS scheduling.

GANTT CHART



Process	Completion time	Turn around time	Waiting time
P1	33	28	23
P2	10	6	0

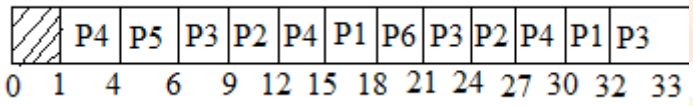
P3	**25**	**22**	**15**
P4	**21**	**20**	**11**
P5	**15**	**18**	**16**
P6	**28**	**22**	**19**
Average		19.3	14

Number of context switches = 10

The same processes with same burst time is calculated by using round robin algorithm,

Process	Arrival time	Burst time	Completion time	Turn around time	Waiting time
P1	**5**	**5**	**32**	**27**	**22**
P2	**4**	**6**	**27**	**23**	**17**
P3	**3**	**7**	**33**	**30**	**23**
P4	**1**	**9**	**30**	**29**	**20**
P5	**2**	**2**	**6**	**4**	**2**
P6	**6**	**3**	**21**	**15**	**12**
Average				21.3	16

GANTT CHART



Number of context switches = 12

2. Consider the process P1-P6 with the following details as shown in table; in this case priority of each process is not given.

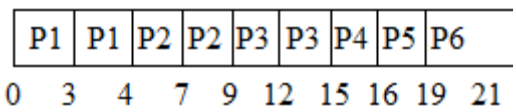
Process	Arrival Time	Burst Time
P1	**0**	**4**
P2	**1**	**5**
P3	**2**	**6**
P4	**4**	**1**
P5	**6**	**3**
P6	**7**	**2**

TQ = N/2 = 3

Ready queue has P1, P2, P3, P4, P5, and P6

Each process is executed for the particular TQ and the remaining burst time is calculated. If it is less than TQ then the process is allowed to complete execution.

GANTT CHART



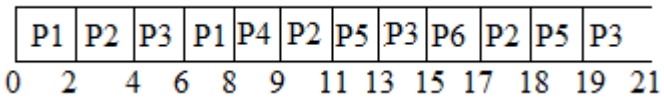
Process	Completion time	Turn around time	Waiting time
P1	4	4	0
P2	9	8	3
P3	15	13	7
P4	16	12	11
P5	19	13	10
P6	21	14	12
Average		10.6	7.1

Number of context switches = 9

Following are the results for round robin algorithm, where TQ=2

Process	Arrival time	Burst time	Completion time	Turn around time	Waiting time
P1	0	4	8	8	4
P2	1	5	18	17	12
P3	2	6	21	19	13
P4	4	1	9	5	4
P5	6	3	19	13	10
P6	7	2	17	10	8
Average				12	8.5

GANTT CHART

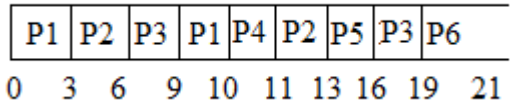


Number of context switches = 12

3. Same processes with same burst time, when TQ is taken as 3, the following results are observed in RR.

Process	Arrival time	Burst time	Completion time	Turn around time	Waiting time
P1	0	4	10	10	6
P2	1	5	13	12	5
P3	2	6	19	17	11
P4	4	1	11	7	6
P5	6	3	16	10	7
P6	7	2	21	14	12
Average				13.3	7.8

GANTT CHART



Number of context switches = 9

V. EXPERIMENTAL RESULTS

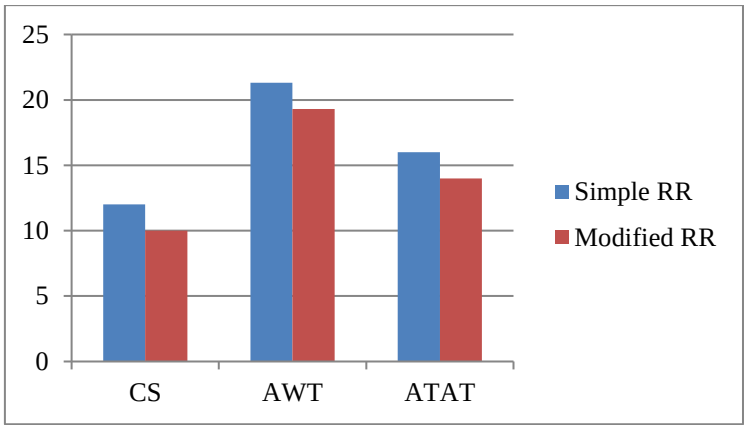


Fig 1: Performance analysis based on experiment 1

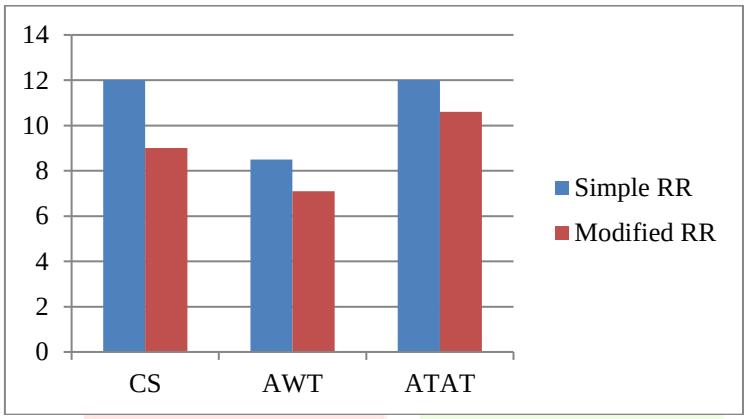


Fig 2: Performance analysis based on experiment 2

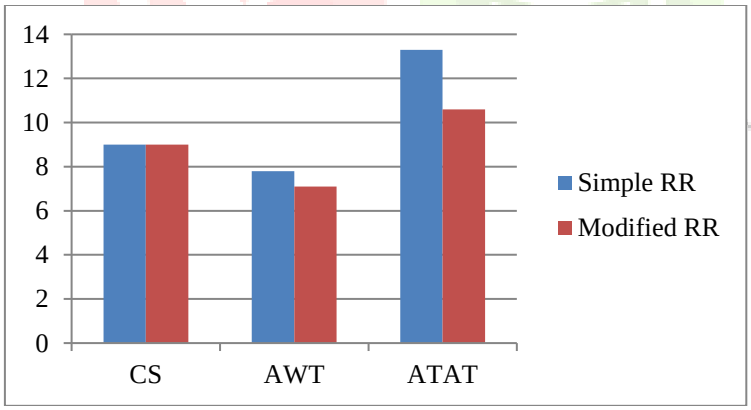


Fig 3: Performance analysis based on experiment 3

VI CONCLUSION

In this paper an efficient and improved RR algorithm is proposed and a comparative study of RR algorithm and the proposed one is made. Experimental results shows that the proposed algorithm shows better performance than the existing RR algorithm as it has less waiting time, less turnaround time and usually less context switching. This reduces overhead and memory space utilization. Starvation of process is also decreased drastically as the priority of process and remaining burst time are also taken into consideration. This algorithm can be implemented to improve the performance of hard real time systems where deadlines are taken into consideration.

REFERENCES

[1] Sabrian, F., C.D. Nguyen, S. Jha, D. Platt and F. Safaei, (2005). Processing resource scheduling in programmable networks. Computer communication, 28:676-687

[2] Silberschatz, A., Peterson, J. L., and Galvin,B.,*Operating System Concepts,* AddisonWesley, 7th Edition, 2006.

[3] Ajit Singh, Priyanka Goyal, Sahil Batra," AnOptimized Round Robin SchedulingAlgorithm for CPU Scheduling", (IJCSE)International Journal on Computer Science andEngineering Vol. 02, No. 07, 2383-2385, 2010

[4] Kevin Jeffay, F. Donelson Smith, Arun Moorthy, James Anderson, "Proportional Share Scheduling of Operating System Services for Real-Time Applications", In Proceedings of the 19th IEEE Real-Time Systems Symposium, Madrid, Spain, December 1998.

[5] M.K. Srivatav, Sanjay Pandey, IndreshGahoi and Neelesh Kumar Namdev, "Fair Priority Round Robin with Dynamic Time Quantum", International Journal of Modern Engineering Research, Vol. 02, Issue. 03, May-June 2012, pp. 876-881.

[6] Rakesh Kumar Yadav, Abhishek K Mishra, Navin Prakash and Himanshu Sharma, "An Improved Round Robin Schedduling Algorithm for CPU Scheduling", International Journal on Computer Science and Engineering, Vol. 02, No. 04, 2010, pp. 1064-1066.

[7] Saroj Hiranwal, Dr. K.C. Roy, "Adaptive Round Robin Scheduling using Shortest Burst Approach Based on Smart Time Slice", International Journal of Computer Science and Communication Vol. 2, No. 2, July-December 2011, pp. 319-323.

[8] Rami J. Matarneh," Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now Running Processes", American Journal of Applied Sciences 6 (10): 1831-1837, 2009 ISSN 1546-9239 © 2009 Science Publications.

[9] H.S.Behera, R. Mohanty, Debashree Nayak, "A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and its Performance Analysis", Volume 5-No. 5, August 2010, International Journal of Computer Application(0975-8887)

[10] Sunita Mohan, "Mixed Scheduling (A New Scheduling Policy)", Proceedings of Insight'09, 25-26 November 2009.

[11] Haidar M. Ali, Kaies Khalid, "An Improvement on Round Robin Scheduling Method" International Conference on Information Technology and Natural Sciences, ICITNS-2003.