IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 6, Ver. IV (Nov.-Dec. 2016), PP 35-40
www.iosrjournals.org

An Improved Dynamic Round Robin CPU Scheduling Algorithm Based on Variant Time Quantum

Jayanti Khatri
Department of Computer Engg. Poornima University, Jaipur, India

Abstract: *Scheduling is considered one of the main factors that affect the performance of real-time embedded systems. It plays a vital role by switching the CPU among several processes concentrating on maximizing the CPU utilization and throughput and minimizing the waiting time, turnaround time and number of context switching for a set of processes. Many algorithms are available for CPU scheduling such as First Come First Serve (FCFS), Shortest Job First (SJF), Round Robin (RR), Priority Scheduling etc. Among all these algorithms Round Robin is most popular and widely used algorithm for timesharing systems. This paper presents a new Improved Dynamic Round Robin approach (IDRR) to reduce the average waiting time and turnaround time. The proposed algorithm IDRR is also compared with various variants of Round Robin algorithm. It produces minimal average waiting time and average turnaround time.*

Keywords: *CPU Scheduling, RR Scheduling, Waiting Time, Turnaround Time, Gantt Chart, Burst Time*

I. Introduction

Scheduling is one of the fundamental features and it is responsible about deciding which job is selected and run from ready queue [1]. The main purpose of scheduling algorithm is to ensure completely fairness between different processes in the ready queue, maximizing the throughput, minimizing the average waiting and turnaround times and the overhead occurs from context switches and make sure no starvation happen at all [2]. There are many CPU scheduling algorithms which vary in efficiency according to holding environments.
The selection criteria of a CPU scheduling algorithm depend upon the following [3]:

1) Fairness: All processes must fairly get the CPU and no one gets into starvation.
2) CPU utilization: CPU should remain busy for 100% time.
3) Throughput: Increase the number of processes that have finished their execution within a certain time interval.
4) Response time: It must be kept minimum. It is the time when request is submitted for the process till the first response of the process is produced.
5) Waiting time: It is the time a process spends in ready queue. It must be kept minimum.
6) Turnaround time: It is the time from submission of the request till the time it is completed. It must be kept minimum.
7) Context Switch: When a process is preempted, its context is stored so that it can resume later from the same point. It is totally an overhead because CPU does no useful work during context switch. Also it adds overhead for the scheduler. Hence, context switches should be made minimum.

Multiple algorithms exist for CPU scheduling. The core algorithms are [4,5]: First Come First Serve, Shortest Job First, Round Robin, and Priority Scheduling. In First Come First Serve processes present in ready queue are scheduled in the same order in which they come. In Shortest Job First (SJF) a process with the shortest burst time is given the processor first to execute and then the next shortest and so on until the ready queue become empty. This algorithm requires the pre knowledge of the next process burst time. In preemptive SJF the process is preempted if any process with a shorter burst time arrives. Round Robin Scheduling assigns a fixed time quantum to each process in equal portions in cyclic order. The processes can execute in their time quantum only and when their time quantum expires the process is preempted from the CPU and placed at the end of ready queue. In Fixed Priority Preemptive Scheduling the process which has the highest priority is scheduled first, then the process with second largest priority and so on.

This paper is organized as follows: Section 2 discusses related work on scheduling algorithms followed by detailed discussion of the proposed approach in Section 3. Section 4 describes experimental analysis of proposed approach and provides a comparative analysis result about the fairness of proposed approach. Section 5 concludes the paper.

II. Related Work

Round Robin scheduling algorithm using static time quantum has many drawbacks. In the recent years, many research works has been done to improve the performance of Round Robin algorithm. A Self Adjustment Round Robin [6] scheduling algorithm uses dynamic time quantum repeatedly adjusted according to the burst time of the now-running processes. IMRRSJF [7] algorithm uses mean and highest burst time to calculate the time quantum. Sort the ready queue in ascending order according to processes' burst time. Calculate the time quantum using mean and median. If mean is greater than median then time quantum is equal to ceil(sqrt((mean*highest burst time)+(median*lowest burst time))) and if median is greater than mean then time quantum is equal to ceil(sqrt((median*highest burst time)+(mean*lowest burst time))) otherwise time quantum is equal to mean [8]. Adaptive Round Robin [9] is a novel approach based on Shortest Burst Time using Smart Time Slice. Sort the processes according to their burst time and calculate Smart Time Slice. Smart Time Slice is equal to mid process burst time of all the CPU burst time when number of process given odd and if number of process given even then time quantum is chosen according to average CPU burst time of all running processes. In [10], processes in the ready queue are sorted in ascending order according to processes' burst time and time quantum is computed with the help of median and highest burst time. In [11], Time Slice is equal to the median of all the sorted processes present in ready queue. Time slice is recalculated taking the remaining burst time in account after each cycle. Enhanced Round Robin [12] scheduling algorithm allocates the processor to the first process of the ready queue for a time interval of up to 1 time quantum. Then it checks the remaining burst time of the currently running process and if the remaining burst time is less than or equal to 1 time quantum, the processor again allocated to the same process. After completing the execution, this process is removed from the ready queue. Improved Round Robin scheduling using Dynamic Time Quantum [13] uses median method to find out optimal time quantum. Processes in ready queue are sorted according to their burst time in ascending order. Time quantum is calculated using the median and the highest burst time. IRRVQ [14] combines Round Robin with Shortest Job First to improve the performance. After arranging the processes in ascending order according to their burst time, time quantum is set equal to burst time of the first process.

III. PROPOSED APPROACH

The proposed algorithm IDRR is similar to Self Adjustment Round Robin algorithm with a small improvement. In the proposed algorithm median is considered as optimal time quantum for the set of processes in the ready queue, if the median is less than 25 then its value must be taken as 25 to avoid the overhead of context switch. The proposed algorithm allocates the processor to the first process of the ready queue for a time interval of up to 1 time quantum. Then it checks the remaining burst time of the currently running process and if the remaining burst time is less than or equal to 1 time quantum, the processor again allocated to the same process. After completing the execution, this process is removed from the ready queue. If the remaining burst time of the currently running process is longer than 1 time quantum, the process will be added at the tail of the ready queue

The proposed algorithm performs following steps:

Step 1: Start

Step 2: Make a ready queue of the processes

Step 3: Calculate the median of the CPU burst time of all the processes

Step 4: Set the time quantum equal to median and if the median is less than 25 then its value must be modified to 25.

Step 4: Allocate the CPU to the first process of the ready queue for a time interval of up to 1 time quantum

Step 5: If the remaining burst time of the currently running process is less than 1 time quantum then allocate CPU again to the currently running process for the remaining burst time otherwise remove the currently running process from the ready queue and put it at the tail of the ready queue

Step 6: After completion of execution remove it from the ready queue

Step 7: Repeat steps 2,3,4,5 and 6 WHILE ready queue becomes empty

Step 8: END

IV. EXPERIMENTAL ANALYSIS

4.1 Assumptions

For evaluating the performance it is assumed that all experiments are performed in uniprocessor environment and burst time for all the processes is known before submitting the processes to the scheduler. All processes have equal priority and arrival time of all processes is same. Moreover all processes are CPU bound.

4.2 Experiments

In IDRR I have considered three different cases. In first case CPU burst time has been considered in increasing order. In second case CPU burst time has been considered in random order and in third case CPU burst time has been considered in decreasing order.

Case I: CPU Burst Time in Increasing Order

Consider seven processes in ready queue along with their burst time shown in Table I. The data are taken from [8]. It is assumed that the arrival time and priority of all processes are same. The comparison results of RR, SARR and IDRR is shown in Table II.

Gantt charts for RR, SARR and IDRR are shown in Fig. 1, Fig. 2 and Fig. 3 respectively and the graph is plotted in Fig. 4.

TABLE I: Burst Times of processes in ready queue [8]

Process	Burst time(ms)
P1	20
P2	25
P3	35
P4	50
P5	80
P6	90
P7	120

TABLE II: Comparison of RR, SARR and IDRR

Algorithm	Time Quantum(ms)	Average Waiting Time(ms)	Average Turnaround Time(ms)
RR	40	152.1	212.1
SARR [6]	50,40,30	133.5	193.5
IDRR	50,70	112.1	172.1

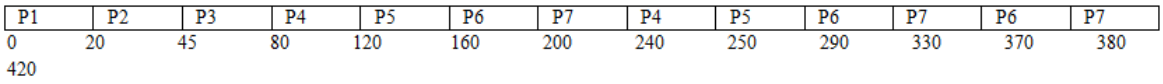

Fig.1 Gantt Chart for RR

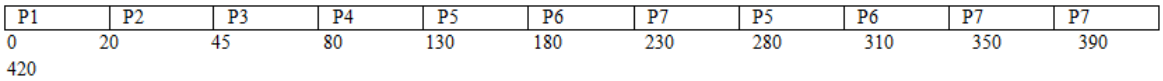

Fig.2 Gantt Chart for SARR

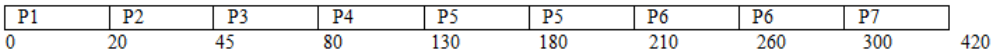

Fig. 3 Gantt Chart for IDRR

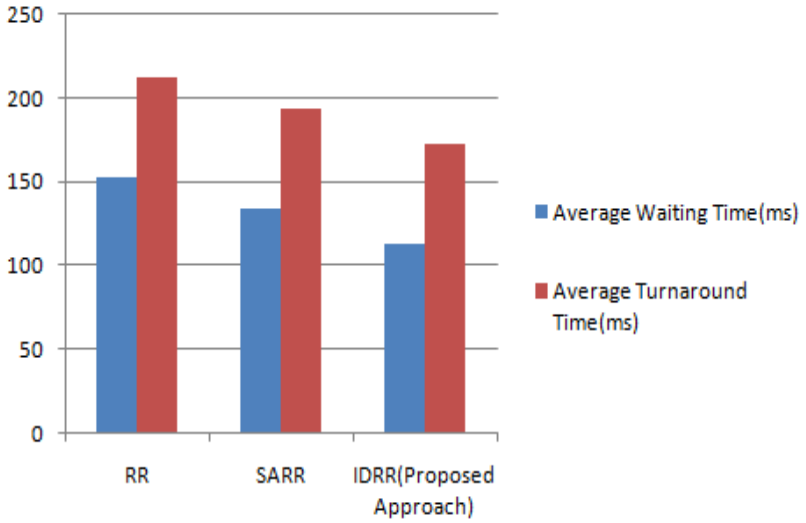

Fig. 4 Comparative Analysis of RR, SARR and IDRR

Case II: CPU Burst Time in Random Order

Consider five processes in ready queue along with their burst time as shown shown in Table III.

The comparison results of RR, SARR and IDRR is shown in Table IV.

Gantt charts for RR, SARR and IDRR are shown in Fig. 5, Fig. 6 and Fig. 7 respectively and the graph is plotted in Fig. 8.

TABLE III: Burst Times of processes in ready queue [8]

Process	Burst time(ms)
P1	80
P2	60
P3	20
P4	10
P5	30

TABLE IV: Comparison of RR, SARR and IDRR

Algorithm	Time Quantum(ms)	Average Waiting Time(ms)	Average Turnaround Time(ms)
RR	25	101	141
SARR [6]	25,35,25	101	141
IDRR	25,45	85	125

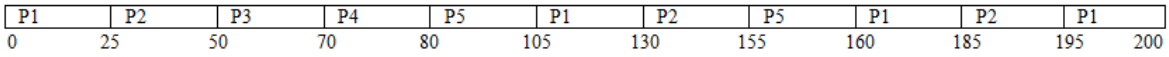

Fig. 5. Gantt Chart for RR

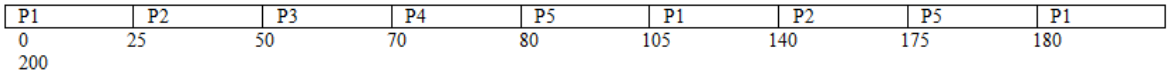

Fig. 6 Gantt Chart for SARR

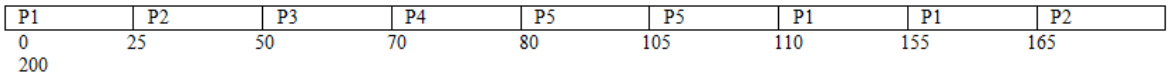

Fig. 7 Gantt Chart for IDRR

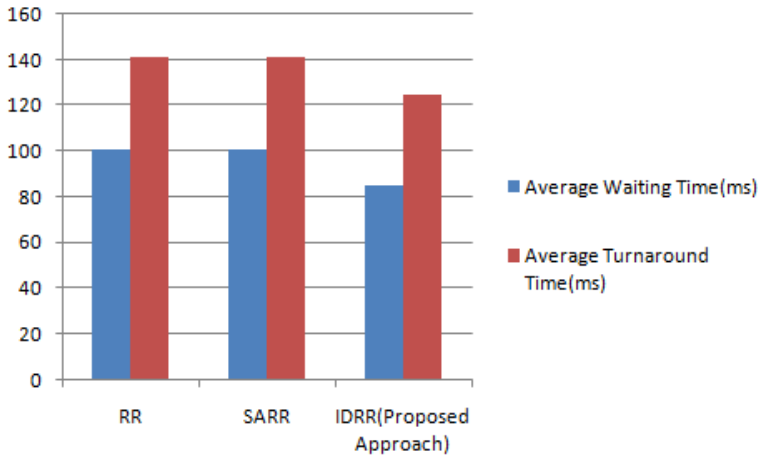

Fig. 8 Comparative Analysis of RR, SARR and IDRR

Case III: CPU Burst Time in Decreasing Order

Consider five processes in ready queue along with their burst time as shown shown in Table V.
The comparison results of RR, SARR and IDRR is shown in Table VI.
Gantt charts for RR, SARR and IDRR are shown in Fig. 9, Fig. 10 and Fig. 11 respectively and the graph is plotted in Fig. 12.

TABLE V: Burst Times of processes in ready queue [8]

Process	**Burst time(ms)**
P1	80
P2	50
P3	40
P4	20
P5	15
P6	10
P7	5

TABLE VI: Comparison of RR, SARR and IDRR

Algorithm	Time Quantum(ms)	Average Waiting Time(ms)	Average Turnaround Time(ms)
RR	20	108.3	140.0
SARR [6]	25,25,30	116.4	147.8
IDRR	25,40	113.5	145

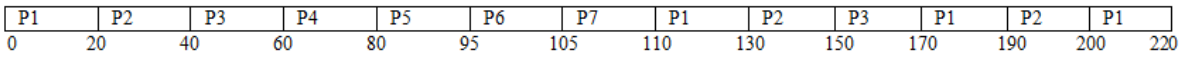

Fig. 9 Gantt Chart for RR

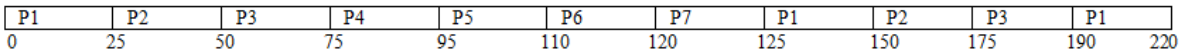

Fig. 10 Gantt Chart for SARR

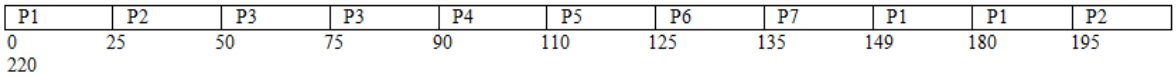

Fig. 11 Gantt Chart for IDRR

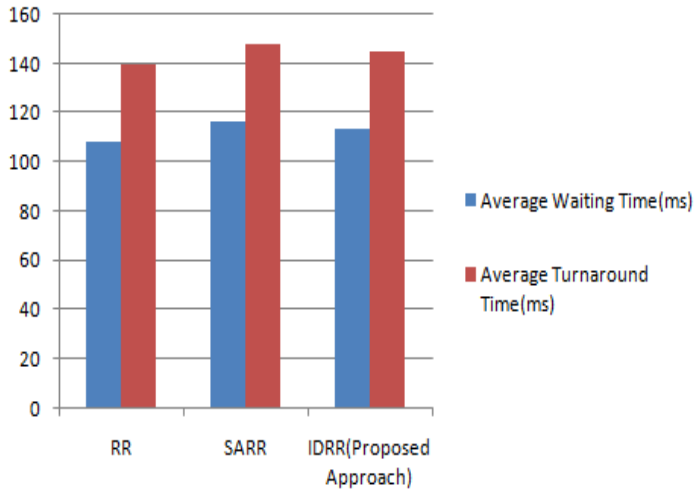

Fig. 12 Comparative Analysis of RR, SARR and IDRR

V. Conclusion

Time Quantum plays a very crucial role in round robin scheduling. This paper presents an improved dynamic round robin algorithm which is based on dynamic time quantum. The proposed algorithm is compared with traditional Round Robin and Self Adjustment Round Robin algorithm. From the experimental results it is proved that the proposed approach is better than Round Robin and Self Adjustment Round Robin because of reduced average waiting time and average turnaround time. But when burst time of processes are in decreasing order, Round Robin and SARR gives better results than the proposed approach. This algorithm can be implemented to improve the CPU performance in time sharing systems.

References

[1]. A. R. Dash. S. K. Sahu and S. K. Samantra, An Optimized Round Robin CPU Scheduling Algorithm with Dynamic Time Quantum *International Journal of Computer Science, Engineering and Information Technology (IJCSEIT), Vol. 5, No. I, February 2015.*

[2]. Ahmed Alsheikhy, Reda Ammar and Raafat Elfouly An Improved Dynamic Round Robin Scheduling Algorithm Based on a Variant Quantum Time, *978-1-5090-0275-7/15/$31.00 ©2015 IEEE*

[3]. Arfa Yasin, Ahmed Faraz, Saad Rehman, Prioritized Fair Round Robin Algorithm with Variable Time Quantum, *13th International Conference on Frontiers of Information Technology 2015 IEEE*

[4]. Silberschatz, Galvin and Gagne, *Operating system concepts*, 8th edition, Wiley, 2009

[5]. Tenanbaum, A.S., *Modern Operating Systems*, 3rd edition, PHI.

[6]. 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*

[7]. Radhe Shyam, Sunil Kumar Nandal, Improved Mean Round Robin with Shortest Job First Scheduling, *International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 7, July 2014.*

[8]. Raman, Dr. Pardeep Kumar Mittal, An Efficient Dynamic Round Robin CPU Scheduling Algorithm, *International Journal of Advanced Research in Computer Science and Software engineering, Volume 4, Issue 5, May 2014 ISSN: 2277 128X.*

[9]. Vishnu Kumar Dhakad1, Lokesh Sharma, Performance Analysis of Round Robin Scheduling using Adaptive Approach based on Smart Time Slice and comparison with SRR, *International Journal of Advances in Engineering & Technology, May 2012, ©IJAET, ISSN: 2231-1963.*

[10]. P.Surendra Varma, A Best possible Time quantum for Improving Shortest Remaining Burst Round Robin (SRBRR) Algorithm, *International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 11, November 2012.*

[11]. H.S.Behera, R. Mohanty, Debashree Nayak, A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis, *International Journal of Computer Applications (0975 – 8887), Vol. 5– No.5, August 2010*

[12]. Jayanti Khatri, An Enhanced Round Robin CPU Scheduling Algorithm, *IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 4, Ver. II (Jul.-Aug. 2016), PP 20-24, DOI: 10.9790/0661-1804022024.*

[13]. Debashree Nayak, Sanjeev Kumar Malla, Debashree Debadarshini, Improved Round Robin Scheduling using Dynamic Time Quantum, *International Journal of Computer Applications (0975 – 8887), Vol. 38– No.5, January 2012.*

[14]. Manish Kumar Mishra1 and Dr. Faizur Rashid, An Improved Round Robin CPU Scheduling Algorithm with Varying Time Quantum, *International Journal of Computer Science, Engineering and Applications, Vol.4, No.4, August 2014*