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
www.iosrjournals.org

An Enhanced Round Robin CPU Scheduling Algorithm

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

Abstract: *CPU scheduling is a process which allows one process to use the CPU while the another process is in waiting state due to unavailability of any resource like I/O etc, thereby making full use of CPU. The aim of CPU scheduling is to make the system efficient, fast and fair. Most CPU scheduling algorithms focus on maximizing CPU utilization, maximum throughput and minimizing waiting time, turnaround time, response time and number of context switches for a set of processes. There are many algorithms available for CPU scheduling and all algorithms have their own advantages and limitations. Round Robin algorithm is designed specifically for time sharing systems. The main aim of this paper is to design a new algorithm that enhances the performance of traditional Round Robin algorithm. The proposed algorithm Enhanced Round Robin (ERR) compared with the traditional Round Robin algorithm and Improved Round Robin algorithm, produces minimal average waiting time (AWT) and average turnaround time (ATT).*

Keywords: *Burst Time, Gantt Chart, Round Robin Scheduling, Time Quantum, Turnaround Time, Waiting Time.*

I. Introduction

In multi-programmed operating systems CPU scheduling plays a fundamental role by switching the CPU among various processes [1]. Scheduling methods affects CPU performance since it determines the CPU and resources utilization [2]. For switching the CPU among various processes there are many CPU scheduling algorithms. 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 [3].

The selection criteria of a CPU scheduling algorithm depend upon the following [4]:

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.

The core algorithms for CPU scheduling are [5]: First Come First Serve, Shortest Job First, Round Robin and Priority Scheduling. The scheduling criteria for all these algorithms are different. In Round Robin scheduling, time quantum is assigned to each process in equal portion and in circular order, dealing with all processes without any priority.

My contribution in this paper is done by developing an Enhanced Round Robin algorithm. The main objective of this algorithm is to improve the performance of CPU by minimizing the average waiting time and turnaround time.

The remainder of this paper presents related work on scheduling algorithms in Section 2, followed by detailed discussion of the proposed approach in Section 3. Section 4 describes experimental analysis of proposed approach along with comparative analysis with different existing Round Robin schemes. Section 5 includes conclusion of the paper.

II. Related Work

In the recent years, many improved Round Robin algorithms have been introduced. An Improved Round Robin CPU Scheduling Algorithm [6] allocates the CPU to the processes for one time quantum and if the remaining burst time is less than 1 time quantum then CPU is again allocated to the currently running process for remaining burst time. It gives better results than traditional Round Robin algorithm. In [7], all the processes are allocated to the CPU for one time quantum. After executing all the processes once, double the initial time quantum and now select the shortest process from the ready queue. Allocate all the processes to the CPU for one time quantum in first cycle and after first cycle select shortest job from the ready queue [8]. Sort the ready queue in ascending order according to processes' burst time and compute time quantum with the help of median and highest burst time [9]. Adaptive Round Robin approach [10] based on Shortest Burst Time using 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. SARR [11] uses dynamic time quantum, in which time quantum is repeatedly adjusted according to burst time of the running processes. In Improved Round Robin scheduling using Dynamic Time Quantum [12], median method is used 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. In [13], Ready queue is sorted according to processes' burst time and median is calculated. Time slice is equal to the median of all the processes present in ready queue. Time slice is recalculated taking the remaining burst time in account after each cycle. IRRVQ [14] combines Round Robin with Shortest Job First. Time quantum is set equal to burst time of the first process in the ready queue after arranging the processes' in ascending order according to their burst time. A new algorithm, called AN, based on new approach called dynamic time quantum [15] is introduced in which the average of the burst time of the processes is calculated after every cycle and allocated as dynamic time quantum.

III. Proposed Approach

The proposed algorithm is similar to traditional Round Robin algorithm with a small improvement. 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: Allocate the CPU to the first process of the ready queue for a time interval of up to 1 time quantum
Step 4: If the remaining burst time of the currently running process is less than or equal to 1 time quantum then allocate CPU again to the currently running process for the remaining burst time.
Step 5: After completion of execution remove it from the ready queue and go to step 3
Step 6: Repeat steps 3,4 and 5 WHILE ready queue becomes empty
Step 7: Remove the currently running process from the ready queue and put it at the tail of the ready queue.
Step 8: END

IV. Experimental Analysis

4.1 Assumptions

It has been assumed that the system where all the experiments are performed is single processor system. All processes have equal priority and number of processes, burst time and time quantum are known before submitting the processes to the processor. Arrival time of all the processes is same. All processes are CPU bound.

4.2 Experiments

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

4.2.1 CASE 1: CPU Burst time in Random order

Consider five processes in ready queue along with their burst time shown in Table I. The data is taken from [6]. It is assumed that the arrival time and priority of all processes are same. The comparison results of RR, IRR and ERR are shown in Table II.

Gantt charts for RR (Time Quantum is 10ms), IRR and ERR 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 [6]

Process	Burst time(ms)
P1	20
P2	34

P3	5
P4	12
P5	26

Table II: Comparison of RR, IRR and ERR

Algorithm	Average Waiting Time(ms)	Average Turnaround Time(ms)
RR	47	67.4
IRR	40.4	59.4
ERR	37	56.4

P1	P2	P3	P4	P5	P1	P2	P4	P5	P2	P5	P2

0 10 20 25 35 45 55 65 67 77 87 93 97

Fig. 1 Gantt Chart for RR when TQ=10ms (Case I)

P1	P2	P3	P4	P4	P5	P1	P2	P5	P5	P2	P2

0 10 20 25 35 37 47 57 67 77 83 93 97

Fig. 2 Gantt Chart for IRR when TQ=10ms (Case I)

P1	P1	P2	P3	P4	P4	P5	P2	P5	P5	P2	P2

0 10 20 25 35 45 47 57 67 77 83 93 97

Fig. 3 Gantt Chart for ERR when TQ=10ms (Case I)

The following chart shows the result of comparison analysis among RR, IRR and ERR.

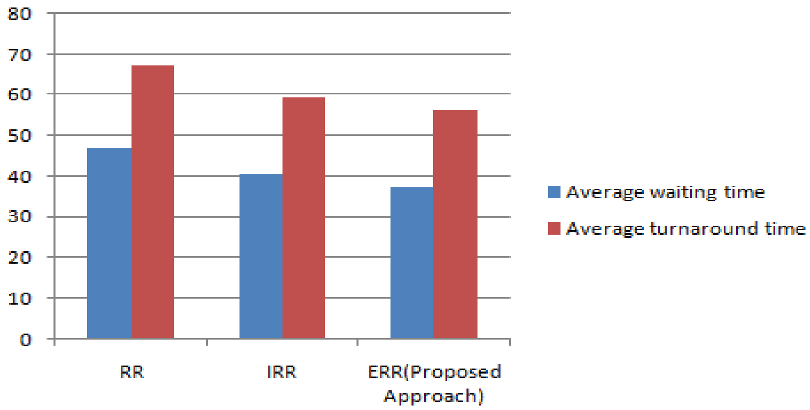

Fig. 4 Comparative Analysis of RR, IRR and ERR(Case I)

4.2.2 CASE 2: CPU Burst Time in Increasing Order

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

Table III: Burst Times of processes in ready queue

Process	Burst time(ms)
P1	5
P2	12
P3	20
P4	26
P5	34

The comparison results of RR, IRR and ERR are shown in Table IV.
Gantt charts for RR (Time Quantum is 10ms), IRR and ERR are shown in Fig. 5, Fig. 6 and Fig. 7 respectively and graph is plotted in Fig. 8.

Table IV: Comparison of RR, IRR and ERR

Algorithm	Average Waiting Time(ms)	Average Turnaround Time(ms)
RR	38.4	57.8
IRR	30.4	49.8
ERR	26.4	45.8

P1	P2	P3	P4	P5	P2	P3	P4	P5	P4	P5	P5	
0	5	15	25	35	45	47	57	67	77	83	93	97

Fig. 5 Gantt Chart for RR when TQ=10ms (Case II)

P1	P2	P2	P3	P4	P5	P3	P4	P4	P5	P5	P5	
0	5	15	17	27	37	47	57	67	73	83	93	97

Fig. 6 Gantt Chart for IRR when TQ=10ms (Case II)

P1	P2	P2	P3	P3	P4	P5	P4	P4	P5	P5	P5	
0	5	15	17	27	37	47	57	67	73	83	93	97

Fig. 7 Gantt Chart for ERR when TQ=10ms (Case II)

The following chart shows the result of comparison analysis among RR, IRR and ERR.

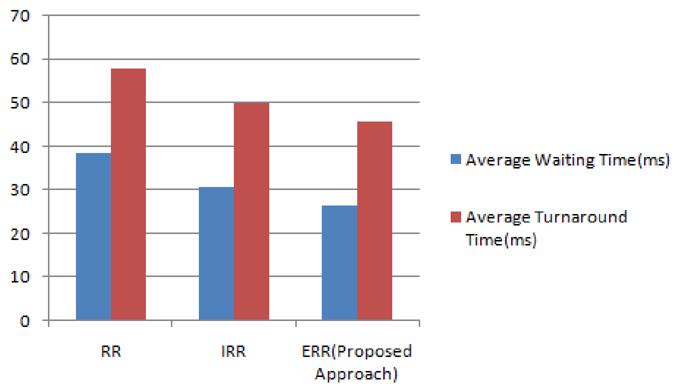

Fig. 8 Comparative Analysis of RR, IRR and ERR (Case II)

4.2.3 CASE 3: CPU Burst Time in decreasing Order

Consider five processes in ready queue along with their burst time in decreasing order shown in Table V.

Table V: Burst Times of processes in ready queue

Process	Burst time(ms)
P1	34
P2	26
P3	20
P4	12
P5	5

The comparison results of RR, IRR and ERR are shown in Table VI.
Gantt charts for RR (Time Quantum is 10ms), IRR and ERR are shown in Fig. 9, Fig. 10 and Fig. 11 respectively and the graph is plotted in Fig. 12.

Table VI: Comparison of RR, IRR and ERR

Algorithm	Average Waiting Time(ms)	Average Turnaround Time(ms)
RR	58	77.4
IRR	49	68.4
ERR	46.4	65.8

P1	P2	P3	P4	P5	P1	P2	P3	P4	P1	P2	P1	
0	10	20	30	40	45	55	65	75	77	87	93	97

Fig. 9 Gantt Chart for RR when TQ=10ms (Case III)

P1	P2	P3	P4	P4	P5	P1	P2	P2	P3	P1	P1	
0	10	20	30	40	42	47	57	67	73	83	93	97

Fig. 10 Gantt Chart for IRR when TQ=10ms (Case III)

P1	P2	P3	P3	P4	P4	P5	P1	P2	P2	P1	P1	
0	10	20	30	40	50	52	57	67	77	83	93	97

Fig. 11 Gantt Chart for ERR when TQ=10ms (Case III)

The following chart shows the result of comparison analysis among RR, IRR and ERR.

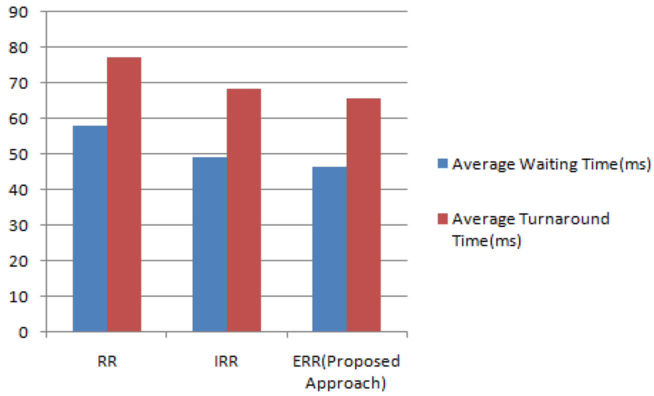

Fig. 12 Comparative Analysis of RR, IRR and ERR (Case III)

V. Conclusion and Future Scope

In this paper an enhanced round robin CPU scheduling algorithm is proposed. The proposed approach is compared with traditional Round Robin and Improved Round Robin algorithm [1]. From the results of comparison it is concluded that the proposed approach is better than Round Robin and Improver Round Robin because of reduced average waiting time and average turnaround time. This algorithm can be implemented to improve the performance of CPU in time sharing systems. In future work, processes at different arrival times and different priorities can be considered for the proposed algorithm.

References

[1]. A. Silberschatz, P. B. Galvin, and G. Gagne, *Operating System Concepts* (7th Edn., John Wiley and Sons Inc, 2005, ISBN 0-471-69466-5).

[2]. 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.

[3]. Ahmed Alsheikhy, Reda Amma 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

[4]. 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.

[5]. Imran Qureshi, "CPU Scheduling Algorithms: A Survey", *Int. J. Advanced Networking and Applications, Vol. 05, Issue 04*, Pages 1968-1973 (2014) ISSN 0975-0290.

[6]. Manish Kumar Mishra, Abdul Kadir Khan, "An Improved Round Robin CPU Scheduling Algorithm", *Journal of Global Research in Computer Science, Vol. 3, No. 6*, June 2012.

[7]. Ajit Singh, Priyanka Goyal, Sahil Batra," An Optimized Round Robin Scheduling Algorithm for CPU Scheduling", *International Journal on Computer Science and Engineering, Vol. 02, No. 07*, 2010, 2383-2385

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

[9]. 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, Vol. 2, Issue 11*, November 2012.

[10]. Vishnu Kumar Dhakad, 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, Vol. 3, Issue 2*, pp. 333-339, May 2012, ISSN 2231-1963.

[11]. 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.

[12]. 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.

[13]. 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

[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

[15]. Abbas Noon1, Ali Kalakech2, Seifedine Kadry, "A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average", *IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1,* May 2011 ISSN (Online) 1694-0814.